Classification of high-dimensional mislabelled data and online algorithms for high-dimensional streaming data

Peiyun Hu

PhD

University of York

Mathematics

June, 2023

Abstract

Motivated by extensive discussions and applications of big data, we delve into the realm of sparse data, specifically high-dimensional data characterised by a larger number of predictors than sample sizes. The advantages and challenges associated with high-dimensional data have been thoroughly discussed (Donoho et al., 2000). Our research primarily focuses on addressing the challenges on two prevalent domains: *Classification with mislabelled data* and *Online algorithms for streaming data*. To overcome these challenges, we incorporate regularisation methods and utilise Sure Independence Screening (SIS) and Iteratively Sure Independence Screening (ISIS) (Fan and Lv, 2008, Fan and Song, 2010).

In Chapter 3, we introduce a two-step estimation method using resampling for classification with mislabelling, offering enhanced cost-effectiveness over conventional data cleansing. Simulations reveal that direct training on corrupted datasets leads classifiers like Logistic Regression (LR) to perform akin to random guessing. Our method greatly enhances LR classifier efficiency, matching the performance of classifiers on perfectly labelled datasets. Notably, our method aligns closely with the performance of the Bayes classifier in diverse contexts. Real data analysis, using a deliberately mislabelled Framingham Heart Study dataset, underscores our classifier's superiority over one trained on raw data with mislabelling, comparable with one trained on impeccable data.

In Chapter 4, we explore incremental algorithms for streaming data, focusing on Generalised Linear Models (GLMs). Our methodologies parallel offline techniques in both low and high-dimensional analyses but excel in computational efficiency. A highlight of our approach is the avoidance of storing specific data, optimising resources and boosting data security. Analysing data from the National Automotive Sampling System Crashworthiness Data System showcases our method's superiority in estimation accuracy, variable selection, and model interpretation. Our technique significantly outperforms those neglecting variable selection and aligns with conventional offline methods.

Contents

Abstract 2

Contents 3

List of Tables 7

List of Figures 15

Acknowledgements 16

Declarations 18

1 Introduction 20

1.1 Research objective 1: Classification with mislabelled data 21

1.1.1 Background and motivation 21

1.1.2 Contributions of our method 23

1.2 Research objective 2: Online algorithms for streaming data 24

1.2.1 Background and motivation 24

1.2.2 Contributions of our method 26

1.3 Structure of the thesis 28

1.4 Preliminary notations and assumption of sparsity 28

2 Literature review 30

2.1 Variable selection 30

2.2 Penalised generalised linear model 42

2.3 Renewable estimation 44

2.4 High-dimensional classification 50

2.5 Classification with imperfect labels 57

3 Classification with mislabelled data 62

3.1 Methodology 63

3.1.1 The proposed classifier 64

3.1.2 Computational algorithm 66

3.2 Evaluation metrics for simulation study 68

3.3 Mislabelled dataset generation and simulation study set-up 71

3.3.1 Resampling for the mislabelled dataset . . . 72
3.3.2 Simulation study . . . 73
3.4 Determining the tuning parameter for the penalty function in the presence of mislabelled data . . . 82
3.4.1 Modified Leave-P-Out Cross-Validation (mLPOCV) . . . 83
3.4.2 Selection of validation data from the unresampled dataset for second-step estimation . . . 83
3.4.3 Selection of validation data from the resampled dataset for second-step estimation . . . 84
3.4.4 Selection of validation data from resampled and unresampled datasets for second-step estimation . . . 84
3.4.5 Simulation study . . . 85
3.5 The order of estimation in the second-step estimation . . . 87
3.5.1 Simulation study . . . 88
3.6 Estimation using oracle information of flipping probabilities . . . 91
3.6.1 η_0 is known . . . 91
3.6.2 η_1 is known . . . 92
3.6.3 Both η_0 and η_1 are known . . . 93
3.6.4 Simulation study . . . 94
3.7 Classifiers from different ways to cope with mislabelling . . . 96
3.7.1 Estimation on datasets with all labels corrected . . . 96
3.7.2 Estimation on raw datasets . . . 97
3.7.3 Estimation on combined datasets having resampled and unresampled data without considering flipping probabilities . . . 98
3.7.4 Simulation study . . . 98
3.8 The mislabelling probabilities are estimated from the mislabelling ratios obtained from the resampled dataset . . . 105
3.8.1 Methodology . . . 106
3.8.2 Simulation study . . . 106
3.9 Estimation with the Independence Screening (IS) method . . . 113
3.9.1 Methodology . . . 114
3.9.2 Simulation study . . . 117
3.10 Estimation with the Iterative Independence Screening (IIS) method . . . 123

3.10.1 Methodology . . . 124
3.10.2 Simulation study . . . 128
3.11 Real data analysis . . . 134
3.11.1 Description of the dataset and how to reclassify the perfect labels with noise . . . 135
3.11.2 Detailed analysis . . . 137

4 Online algorithms for streaming data **152**
4.1 Model specification . . . 153
4.2 Methodology . . . 155
4.2.1 Offline penalised maximum likelihood estimation algorithm . . . 155
4.2.2 Incremental algorithm for penalised maximum likelihood estimation . . . 156
4.2.3 Simulation study setup . . . 161
4.2.4 Simulation study . . . 162
4.3 Determining the tuning parameter for the penalty function in renewable estimation . . . 170
4.3.1 Modified Leave-P-Out Cross-Validation (mLPOCV) for streaming data . . . 170
4.3.2 Two algorithms of search processes . . . 171
4.3.3 Simulation study: Comparison of two search methods for tuning parameter selection . . . 173
4.3.4 Simulation study: Comparing the algorithm with SCAD penalty function to the one without penalty function . . . 180
4.3.5 Simulation study: Tuning parameter selection on different sizes of validation data in mLpOCV . . . 183
4.4 Iteratively updated tuning parameter . . . 187
4.4.1 Incremental algorithm for penalised maximum likelihood estimation with iteratively updated penalty . . . 188
4.4.2 Simulation study . . . 190
4.5 The incremental algorithm with Independence Screening (IS) and its variant approaches . . . 200
4.5.1 The incremental algorithm with IS . . . 200

4.5.2 The incremental algorithm with IS_{V1}: Variant 1 of IS 202
4.5.3 The incremental algorithm with IS_{V2}: Variant 2 of IS 202
4.5.4 Simulation study: Comparative analysis of online algorithms with SCAD, IS-SCAD, IS_{V1}-SCAD and IS_{V2}-SCAD for the case of $p < n_b$ 203
4.6 The incremental algorithm with Iterative Independence Screening (IIS) and its variant approach 207
4.6.1 The incremental algorithm with IIS 208
4.6.2 The incremental algorithm with IIS_{V1}: Variant 1 of IIS 209
4.6.3 Simulation study: Comparative analysis of online algorithms with IS-SCAD, IS_{V1}-SCAD, IIS-SCAD, and IIS_{V1}-SCAD for the cases of $n_b < p < N_b$ and $p > N_b$ 210
4.6.4 Simulation study: Comparative analysis of online algorithms with IIS-SCAD and IIS_{V1}-SCAD, along with the offline algorithm with IIS-SCAD for the case of $p > N_b$ 216
4.7 Real data analysis 224

5 Discussion **238**
5.1 Research objective 1: Classification with mislabelled data 238
5.2 Research objective 2: Online algorithms for streaming data 241

References **253**

A Appendix: Derivation of estimators for *Classification with mislabelled data* **254**
A.1 Two-step estimation method using LQA and Newton-Raphson algorithm 254

B Appendix: Tables for *Classification with mislabelled data* **258**

C Appendix: Tables for *Online algorithms for streaming data* **262**
C.1 Simulation study 262
C.2 Real data analysis 268

List of Tables

3.1 AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\beta}}$s 76

3.2 AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\eta}}_0$s 76

3.3 AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\eta}}_1$s 76

3.4 SDs and ESEs for non-zero $\boldsymbol{\beta}_{15\times 1}$ coefficient estimates 76

3.5 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$ 77

3.6 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\eta}_0$ 77

3.7 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\eta}_1$ 77

3.8 Excess risks of $C_{\hat{\boldsymbol{\beta}}}$s 78

3.9 AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\beta}}$s 79

3.10 AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\eta}}_0$s 79

3.11 AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\eta}}_1$s 80

3.12 SDs and ESEs for non-zero $\boldsymbol{\beta}$ coefficient estimates 80

3.13 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$ 80

3.14 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\eta}_0$ 81

3.15 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\eta}_1$ 81

3.16 Excess risks of $C_{\hat{\boldsymbol{\beta}}}$s 81

3.17 AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\beta}}$s from methods using different validation datasets 87

3.18 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$ 87
3.19 Excess risks of various classifiers and computing time (in brackets) . . 87
3.20 AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\beta}}$ from the methods employing different estimation orders for the three unknown parameters 90
3.21 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$ 90
3.22 Excess risks of $C_{\hat{\boldsymbol{\beta}}}$s 91
3.23 AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\beta}}$s, $\hat{\boldsymbol{\beta}}_{|\boldsymbol{\eta}_0}$s, $\hat{\boldsymbol{\beta}}_{|\boldsymbol{\eta}_1}$s and $\hat{\boldsymbol{\beta}}_{|\boldsymbol{\eta}_0,\boldsymbol{\eta}_1}$s. . . 95
3.24 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$ 96
3.25 Excess risks of various classifiers. 96
3.26 AMRSEs and MMRSEs (in brackets) of estimates of $\boldsymbol{\beta}$ from different methods handling mislabelled data using various approaches 101
3.27 SDs and ESEs for non-zero $\boldsymbol{\beta}_{15\times1}$ coefficient estimates 101
3.28 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$ 102
3.29 Excess risks of various classifiers 102
3.30 AMRSEs and MMRSEs (in brackets) of estimates of $\boldsymbol{\beta}$ from different methods handling mislabelled data using various approaches 104
3.31 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$ 104
3.32 Excess risks of various classifiers 104
3.33 Averaged mislabelling ratios of full-size generated and resampled datasets. 109
3.34 AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\beta}}$s and $\hat{\boldsymbol{\beta}}_{|\hat{\mathbf{M}}_{\mathbf{y}|0},\hat{\mathbf{M}}_{\mathbf{y}|1}}$s 109

3.35 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$ . . . 109
3.36 Excess risks of $C_{\hat{\boldsymbol{\beta}}}$s and $C_{\hat{\boldsymbol{\beta}}_{|\hat{M}_{y|0},\hat{M}_{y|1}}}$s, and computing time (in brackets) . . . 109
3.37 Average mislabelling ratios of full-size generated and resampled datasets 112
3.38 AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\beta}}$s and $\hat{\boldsymbol{\beta}}_{|\hat{M}_{y|0},\hat{M}_{y|1}}$s . . . 112
3.39 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$ . . . 112
3.40 Excess risks of $C_{\hat{\boldsymbol{\beta}}}$s and $C_{\hat{\boldsymbol{\beta}}_{|\hat{M}_{y|0},\hat{M}_{y|1}}}$s . . . 113
3.41 AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\beta}}$s and $\hat{\boldsymbol{\beta}}_{\text{IS}}$s . . . 120
3.42 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$ . . . 120
3.43 Excess risks of $C_{\hat{\boldsymbol{\beta}}}$s and $C_{\hat{\boldsymbol{\beta}}_{\text{IS}}}$s, and computing time and memory utlisations (in brackets) . . . 120
3.44 AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\beta}}_{\text{IS}}$s and $\hat{\boldsymbol{\beta}}^{*}_{\text{IS}}$s . . . 122
3.45 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$ . . . 122
3.46 Excess risks of $C_{\hat{\boldsymbol{\beta}}_{\text{IS}}}$s and $C_{\hat{\boldsymbol{\beta}}^{*}_{\text{IS}}}$s . . . 122
3.47 AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\beta}}_{\text{IS}}$s and $\hat{\boldsymbol{\beta}}_{\text{IIS}}$s . . . 130
3.48 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$ . . . 130
3.49 Excess risks of $C_{\hat{\boldsymbol{\beta}}_{\text{IS}}}$s and $C_{\hat{\boldsymbol{\beta}}_{\text{IIS}}}$s, and computing time and memory utilisations (in brackets) . . . 131
3.50 AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\beta}}_{\text{IIS}}$s and $\hat{\boldsymbol{\beta}}^{*}_{\text{IIS}}$s . . . 132
3.51 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$ . . . 133
3.52 Excess risks of $C_{\hat{\boldsymbol{\beta}}_{\text{IIS}}}$s and $C_{\hat{\boldsymbol{\beta}}^{*}_{\text{IIS}}}$s . . . 133

3.53 Description of the dataset . . . 137
3.54 mislabelling ratios of the resampled datasets with varying sizes. . . . 140
3.55 Misclassification rates of various classifiers (in %). . . . 140
3.56 Comparisons of different estimates under the setting of resampled data with a size of $m = 292$ . . . 141
3.57 Comparisons of different estimates under the setting of resampled data with a size of $m = 585$ . . . 142
3.58 Comparisons of different estimates under the setting of resampled data with a size of $m = 877$ . . . 143
3.59 Comparisons of different estimates under the setting of resampled data with a size of $m = 1170$ . . . 144
3.60 Comparisons of different estimates under the setting of resampled data with a size of $m = 1462$ . . . 145
3.61 Averaged mislabelling ratios of full-size generated and resampled datasets . . . 147
3.62 Misclassification rates of various classifiers (in %) . . . 147
3.63 Comparisons of different estimates of **CASE 1** under the setting of resampled data with a size of $m = 1170$ . . . 148
3.64 Comparisons of different estimates of **CASE 2** under the setting of resampled data with a size of $m = 1170$ . . . 149
3.65 Comparisons of different estimates of **CASE 3** under the setting of resampled data with a size of $m = 1170$ . . . 150
4.1 AMRSEs and MMRSEs (in brackets) of $\tilde{\boldsymbol{\beta}}_b$s and $\hat{\boldsymbol{\beta}}^*$s . . . 165
4.2 SDs and ESEs for non-zero $\boldsymbol{\beta}_{12\times 1}$ coefficient estimates . . . 165
4.3 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$ . . . 165
4.4 Excess risks of $C_{\tilde{\beta}}$s and $C_{\hat{\beta}^*}$s, and computing time and memory utilisations (in brackets) . . . 166
4.5 AMRSEs and MMRSEs (in brackets) of $\tilde{\boldsymbol{\beta}}_b$s and $\hat{\boldsymbol{\beta}}^*$s . . . 168
4.6 SDs and ESEs for non-zero $\boldsymbol{\beta}_{150\times 1}$ coefficient estimates . . . 169

4.7 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for β 169
4.8 Excess risks of $C_{\tilde{\beta}_b}$s and $C_{\hat{\beta}^*_b}$s, and computing time and memory utilisations (in brackets) 169
4.9 AMRSEs and MMRSEs (in brackets) of $\tilde{\beta}_{\text{One-Way}}$s and $\tilde{\beta}_{\text{Two-Way}}$s 175
4.10 SDs and ESEs for non-zero β coefficient estimates 175
4.11 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for β 175
4.12 Excess risks of $C_{\tilde{\beta}_{\text{One-Way}}}$s and $C_{\tilde{\beta}_{\text{Two-Way}}}$s, and computing time (in brackets) 175
4.13 AMRSEs and MMRSEs (in brackets) of $\tilde{\beta}_{\text{One-Way}}$s and $\tilde{\beta}_{\text{Two-Way}}$s 177
4.14 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for β 177
4.15 Excess risks of $C_{\tilde{\beta}_{\text{One-Way}}}$s and $C_{\tilde{\beta}_{\text{Two-Way}}}$s, and computing time (in brackets) 177
4.16 AMRSEs and MMRSEs (in brackets) of $\tilde{\beta}_{\text{One-Way}}$s and $\tilde{\beta}_{\text{Two-Way}}$s 179
4.17 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for β 179
4.18 Excess risks and computing time (in brackets) 179
4.19 AMRSEs and MMRSEs (in brackets) of $\tilde{\beta}$s and $\tilde{\beta}_{\lambda=0}$s 182
4.20 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for β 183
4.21 Excess risks of classifiers $C_{\tilde{\beta}}$s and $C_{\tilde{\beta}_{\lambda=0}}$s 183
4.22 AMRSEs and MMRSEs (in brackets) of $\tilde{\beta}$s 185
4.23 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for β 185
4.24 Excess risks of $C_{\tilde{\beta}}$s and computing time (in brackets) 185
4.25 AMRSEs and MMRSEs (in brackets) of $\tilde{\beta}_2$s and $\tilde{\beta}_{20}$s 187

4.26 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$ 187
4.27 Excess risks of $C_{\tilde{\boldsymbol{\beta}}_b}$s and computing time (in bracket) 187
4.28 AMRSEs and MMRSEs (in brackets) of $\tilde{\boldsymbol{\beta}}_\lambda$s, $\tilde{\boldsymbol{\beta}}_{\lambda^*}$s and $\hat{\boldsymbol{\beta}}^*$s 192
4.29 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$ 193
4.30 Excess risks of various classifiers and computing time (in brackets) . . 193
4.31 AMRSEs and MMRSEs (in brackets) of $\tilde{\boldsymbol{\beta}}_\lambda$s, $\tilde{\boldsymbol{\beta}}_{\lambda^*}$s and $\hat{\boldsymbol{\beta}}^*$s 196
4.32 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$ 196
4.33 Excess risks of $C_{\tilde{\boldsymbol{\beta}}_\lambda}$s, $C_{\tilde{\boldsymbol{\beta}}_{\lambda^*}}$s and $C_{\hat{\boldsymbol{\beta}}^*}$s, and computing time (in brackets) 196
4.34 AMRSEs and MMRSEs (in brackets) of $\tilde{\boldsymbol{\beta}}_\lambda$s, $\tilde{\boldsymbol{\beta}}_{\lambda^*}$s and $\hat{\boldsymbol{\beta}}^*$s 198
4.35 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$ 199
4.36 Excess risks of classifiers $C_{\tilde{\boldsymbol{\beta}}_\lambda}$s, $C_{\tilde{\boldsymbol{\beta}}_{\lambda^*}}$s and $C_{\hat{\boldsymbol{\beta}}^*}$s, and computing time (in brackets) 199
4.37 AMRSEs and MMRSEs (in brackets) of $\tilde{\boldsymbol{\beta}}_{\text{SCAD}}$s, $\tilde{\boldsymbol{\beta}}_{\text{IS}-\text{SCAD}}$s, $\tilde{\boldsymbol{\beta}}_{\text{IS}_{\text{V1}}-\text{SCAD}}$s and $\tilde{\boldsymbol{\beta}}_{\text{IS}_{\text{V2}}-\text{SCAD}}$s 206
4.38 SDs and ESEs for non-zero $\boldsymbol{\beta}_{150\times1}$ coefficient estimates 206
4.39 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$ 207
4.40 Excess risks of various classifiers and computing time (in brackets) . . 207
4.41 AMRSEs and MMRSEs (in brackets) of $\tilde{\boldsymbol{\beta}}_{\text{IS}-\text{SCAD}}$s, $\tilde{\boldsymbol{\beta}}_{\text{IS}_{\text{V1}}-\text{SCAD}}$s and $\tilde{\boldsymbol{\beta}}_{\text{IIS}-\text{SCAD}}$s 212
4.42 SDs and ESEs for non-zero $\boldsymbol{\beta}_{350\times1}$ coefficient estimates 212
4.43 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$ 213

4.44 Excess risks of various classifiers and computing time 213
4.45 AMRSEs and MMRSEs (in brackets) of $\tilde{\boldsymbol{\beta}}_{\mathrm{IS}-\mathrm{SCAD}}$s, $\tilde{\boldsymbol{\beta}}_{\mathrm{IS}_{\mathrm{V1}}-\mathrm{SCAD}}$s, $\tilde{\boldsymbol{\beta}}_{\mathrm{IIS}-\mathrm{SCAD}}$s and $\tilde{\boldsymbol{\beta}}_{\mathrm{IIS}_{\mathrm{V1}}-\mathrm{SCAD}}$s. 215
4.46 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$ 215
4.47 Excess risks of various classifiers and computing time 216
4.48 AMRSEs and MMRSEs (in brackets) of $\tilde{\boldsymbol{\beta}}_{\mathrm{IIS}-\mathrm{SCAD}}$s, $\tilde{\boldsymbol{\beta}}_{\mathrm{IIS}_{\mathrm{V1}}-\mathrm{SCAD}}$s and $\hat{\boldsymbol{\beta}}^{*}_{\mathrm{IIS}-\mathrm{SCAD}}$s 219
4.49 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$ 219
4.50 Excess risks of $\mathrm{C}_{\tilde{\boldsymbol{\beta}}_{\mathrm{IIS}-\mathrm{SCAD}}}$s, $\mathrm{C}_{\tilde{\boldsymbol{\beta}}_{\mathrm{IIS}_{\mathrm{V1}}-\mathrm{SCAD}}}$s and $\mathrm{C}_{\hat{\boldsymbol{\beta}}_{\mathrm{IIS}-\mathrm{SCAD}}}$s, and computing time and memory utilisations (in brackets) 219
4.51 AMRSEs and MMRSEs (in brackets) of estimates 221
4.52 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$ 221
4.53 Excess risks of $\mathrm{C}_{\tilde{\boldsymbol{\beta}}_{\mathrm{IIS}-\mathrm{SCAD}}}$s, $\mathrm{C}_{\tilde{\boldsymbol{\beta}}_{\mathrm{IIS}_{\mathrm{V1}}-\mathrm{SCAD}}}$s and $\mathrm{C}_{\hat{\boldsymbol{\beta}}^{*}_{\mathrm{IIS}-\mathrm{SCAD}}}$s, and computing time and memory utilisations (in brackets) 222
4.54 AMRSEs and MMRSEs (in brackets) of estimates 223
4.55 Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$ 223
4.56 Excess risks of $\mathrm{C}_{\tilde{\boldsymbol{\beta}}_{\mathrm{IIS}-\mathrm{SCAD}}}$s, $\mathrm{C}_{\tilde{\boldsymbol{\beta}}_{\mathrm{IIS}_{\mathrm{V1}}-\mathrm{SCAD}}}$s and $\mathrm{C}_{\hat{\boldsymbol{\beta}}^{*}_{\mathrm{IIS}-\mathrm{SCAD}}}$s, and computing time and memory utilisations (in brackets) 224
4.57 Variable Descriptions 227
4.58 Comparisons of $\hat{\boldsymbol{\beta}}^{*}_{84,\lambda=0}$ and $\tilde{\boldsymbol{\beta}}_{84,\lambda=0}$ with $\hat{\boldsymbol{\beta}}^{*}$, $\tilde{\boldsymbol{\beta}}_{84,\lambda}$ and $\tilde{\boldsymbol{\beta}}_{84,\lambda^{*}}$ 230
4.59 Comparisons of $\hat{\boldsymbol{\beta}}^{*}_{84,\lambda}$, $\tilde{\boldsymbol{\beta}}_{84,\lambda}$ and $\tilde{\boldsymbol{\beta}}_{84,\lambda^{*}}$ 236
B.1 SDs and ESEs for non-zero $\boldsymbol{\beta}_{120\times 1}$ coefficient estimates 258
B.2 SDs and ESEs for non-zero $\boldsymbol{\beta}_{150\times 1}$ coefficient estimates 258
B.3 SDs and ESEs for non-zero $\boldsymbol{\beta}_{15\times 1}$ coefficient estimates 259
B.4 SDs and ESEs for non-zero $\boldsymbol{\beta}_{15\times 1}$ coefficient estimates 259

B.5 SDs and ESEs for non-zero $\boldsymbol{\beta}_{10\times 1}$ coefficient estimates 260
B.6 SDs and ESEs for non-zero $\boldsymbol{\beta}_{10\times 1}$ coefficient estimates 260
B.7 SDs and ESEs for non-zero $\boldsymbol{\beta}_{150\times 1}$ coefficient estimates 260
B.8 SDs and ESEs for non-zero $\boldsymbol{\beta}_{350\times 1}$ coefficient estimates 261
B.9 SDs and ESEs for non-zero $\boldsymbol{\beta}_{3000\times 1}$ coefficient estimates 261
B.10 SDs and ESEs for non-zero $\boldsymbol{\beta}$ coefficient estimates 261
C.1 SDs and ESEs for non-zero $\boldsymbol{\beta}_{15\times 1}$ coefficient estimates 262
C.2 SDs and ESEs for non-zero $\boldsymbol{\beta}_{15\times 1}$ coefficient estimates 262
C.3 SDs and ESEs for non-zero $\boldsymbol{\beta}_{50\times 1}$ coefficient estimates 263
C.4 SDs and ESEs for non-zero $\boldsymbol{\beta}_{p\times 1}$ coefficient estimates 263
C.5 SDs and ESEs for non-zero $\boldsymbol{\beta}_{100\times 1}$ coefficient estimates 264
C.6 SDs and ESEs for non-zero $\boldsymbol{\beta}$ coefficient estimates 264
C.7 SDs and ESEs for non-zero $\boldsymbol{\beta}_{p\times 1}$ coefficient estimates 265
C.8 SDs and ESEs for non-zero $\boldsymbol{\beta}_{150\times 1}$ coefficient estimates 265
C.9 SDs and ESEs for non-zero $\boldsymbol{\beta}_{1900\times 1}$ coefficient estimates 266
C.10 SDs and ESEs for non-zero $\boldsymbol{\beta}_{3000\times 1}$ coefficient estimates 266
C.11 SDs and ESEs for non-zero $\boldsymbol{\beta}_{3000\times 1}$ coefficient estimates 267
C.12 SDs and ESEs for non-zero $\boldsymbol{\beta}_{3000\times 1}$ coefficient estimates 267
C.13 Comparison of misclassification rates (%) of various classifiers 268
C.14 Comparison of misclassification rates (%) for various classifiers 269

List of Figures

4.1 Trace plots for the coefficient estimates of “INTERCEPT”, “YOUNG”, and “OLD”. 231
4.2 Trace plots for the coefficient estimates of “SEX”, “PARUSE” and “LGTCOND”. 231
4.3 Trace plots for the coefficient estimates of “DRINKING”, “SPLIMIT” and “TRCTLFCT”. 232
4.4 Trace plots for the misclassification rates of renewable classifiers and offline classifiers. 232
4.5 Trace plots for the misclassification rates of two penalised renewable classifiers and the penalised offline classifier. 237

Acknowledgements

Looking back on my PhD journey, there are numerous individuals to whom I am deeply grateful.

Firstly, I wish to express my profound gratitude to my supervisor, Prof. Wenyang Zhang, whose invaluable guidance and mentorship have been instrumental. His proficiency, detailed feedback, and continuous support have been crucial in shaping my research and propelling me to reach my fullest potential.

I extend my genuine thanks to my Thesis Advisory Panel members, Prof. Degui Li and Prof. Marina Knight. Engaging in dialogues with them has always been uplifting; their insightful perspectives, proactive suggestions, and encouragement have been key assets on my research journey.

I am grateful to my fellow PhD candidates at the Mathematics Department of the University of York, particularly Álvaro Guinea, Andrew Scoones, David Serrano Blanco, Diego Vidal, Rutvij Bhavsar, and countless others. Your support, assistance, and dedicated spirit have inspired me to overcome many obstacles. To my colleagues in G/N/160, Asma Alalyani, Kuntal Sengupta, and Leiws Wooltorton, I want to express my gratitude for your wonderful companionship and unwavering support. I will genuinely miss our daily afternoon tea rituals.

My sincere appreciation also goes to Dr. Ben Powell, who provided valuable help, engaging discussions, and profound insights while organising the reading group. This platform offered a wealth of inspiration derived from the diverse viewpoints of others.

Special acknowledgement goes to the friends I made in York, including Vilasini Venkatesh, who has not only inspired me but has also provided continuous encouragement and support. I would like to express my gratitude to José Almanza Medina for your kind help and sincere friendship. I am grateful to Sue Anthony and Simon Anthony, who have always welcomed me and treated me like family. To my film night group friends, including Roberta Merli, Patrycja Chaba, Lixin Chen, Anne Williamson, Yajie Gu, and many others, thank you for the delightful evenings and your steadfast support. To Lois Laurine Folkard and Nicholas Sheridan Folkard, my first close friends in York, I am truly thankful for the tremendous help, advice, understanding, and care you have provided. Words cannot adequately express my

deep appreciation for these invaluable friendships.

Most notably, my heartfelt thanks go out to my parents Yuanzhang Hu and Qutao Chen, and to the entire Chen family for their support throughout my life. Their faith in me, understanding, and unconditional love have been my steady source of motivation and resilience.

Finally, I want to honour the courage of my past self, who, five years ago, dared to pursue her dreams. As I embark on the next phase of my journey, I aim to uphold this courage, bravery, and curiosity, backed by all the love and support I have received.

Declarations

I declare that this thesis presents original work and that I am the sole author. The research conducted in this thesis has been carried out under the supervision of Prof. Wenyang Zhang and has not previously been submitted for an award at this or any other university.

The literature review in Chapter 2 provides key ideas related to this thesis, which include:

- Section 2.1 carefully discusses the methods of variable selection, specifically "Variable selection via nonconcave penalized likelihood and its oracle properties" by Fan and Li (2001) and "Sure independence screening for ultrahigh-dimensional feature space" by Fan and Lv (2008).

- Section 2.2 presents Generalised Linear Models (GLMs) based on the book "Generalized linear models" by McCullagh (1983), and various variable selection methods for GLMs in sparse and high-dimensional data, primarily focusing on "Ultrahigh dimensional feature selection: beyond the linear model" by Fan et al. (2009) and "Sure independence screening in generalized linear models with np-dimensionality" by Fan and Song (2010).

- Section 2.3 discusses "Renewable estimation and incremental inference in generalized linear models with streaming data sets", as proposed by Luo and Song (2020).

- In Section 2.4, high-dimensional classification methods are introduced. The key ideas of our study are primarily based on the variable selection methods presented in "Sure independence screening for ultrahigh-dimensional feature space" and by Fan and Lv (2008) and "Sure independence screening in generalized linear models with np-dimensionality" Fan and Song (2010). These methods enhance the performance of traditional penalised likelihood methods.

- Section 2.5 explores the topic of classification with mislabelled data and includes the study on "Classification with imperfect training labels" by Cannings et al. (2020).

I confirm that all the references cited in this declaration are accurate and complete. Any contributions from other sources have been appropriately referenced.

1 Introduction

The advent of advanced technology has revolutionised data collection, giving rise to the era of big data. Big data analysis has paved the way for transformative analyses across various industries. It empowers organisations to make informed decisions, improve operational efficiency, gain insights into consumer behaviour, advance healthcare practices, optimise management strategies, and foster innovation. Yet new challenges arise with the abundance of data.

Frequently, the data collected in real-world exhibit a sparse nature, meaning that they contain a mixture of both relevant and irrelevant features. During the data collection process, these irrelevant features are often intermixed with the pertinent ones, and manually identifying and filtering out unimportant information are challenging and impractical tasks. These insignificant variables can have a non-negligible adverse impact on estimation accuracy, model interpretability, computing burdens, and other factors. Moreover, they can lead to failures in the analysis using traditional methods such as Ordinary Least Squares (OLS) regression (Sirimongkolkasem and Drikvandi, 2019). Accordingly, we investigate this problem by taking into account the assumption of sparsity. Our specific focus is on analysing scenarios where the data comes from sparse models. In this context, a "sparse model" is characterised by the majority of parameters or variables having zero values, indicating their insignificance and little relevance in the analysis.

In our study, we focus on two commonly encountered data types: mislabelled classification datasets and streaming data. Classification problems are extensively studied in various domains, such as spam email detection, sentiment analysis, and disease diagnosis. However, in real-world scenarios, collecting mislabelled data is a common occurrence and can arise due to various factors. Traditional data cleansing methods, which involve verifying each label in the dataset, are often impractical and less feasible in such situations. Streaming data refers to data that is continuously generated and observed over time, and it has become increasingly prevalent in various domains such as sensor networks, social media platforms, financial markets, and online transaction records. Unlike traditional static datasets, streaming data is constantly updated and evolving, reflecting real-time events. As a result, traditional offline methods are less suitable for analysing this type of data due to their limited

ability to handle real-time updates and evolving patterns.

These data types present unique challenges within the realm of big data, primarily stemming from the growing dimensions of datasets. In order to tackle these challenges, we are driven to develop innovative approaches tailored to each specific data type. In this chapter, we provide a comprehensive background and motivation for each topic individually. We aim to highlight the complexity and unique challenges associated with sparse data in each topic. Furthermore, we present our novel approaches to effectively address the challenges that have remained unsolved in previous studies.

1.1 Research objective 1: Classification with mislabelled data

1.1.1 Background and motivation

Classification is pervasive in our daily lives with broad applications such as disease diagnosis, toxic comment detection, and many others. In disease diagnosis, the aim is to forecast the presence or absence of a disease based on the results of medical tests, symptoms, and patient information, thereby facilitating medical decision-making procedures. Toxic comment identification has gained considerable interest and is a hot topic of debate among various companies, including TikTok and Instagram, with the objective of enhancing the moderation of online communities and encouraging more constructive and respectful dialogues.

For this extensively studied topic, we focus specifically on supervised classification, which involves using a training dataset consisting of input samples with known labels. In our study, we use the term "classification" as a simplified reference to "supervised classification."

Large datasets with a substantial number of samples and predictors are now easily obtainable, ensuring that adequate information and key features are included in the study. Researchers can benefit from the abundance of information available in their data, as it enables them to obtain more accurate and interpretable results. However, in the classification problem, we have identified two main challenges that arise in the presence of large datasets, which are the curse of growing dimensions and the presence of mislabelling. These obstacles can adversely affect the analysis. Current research tends to address these two issues separately.

Among the collected features, many may have weak correlations with the response variables, yet these are difficult to identify manually because they often correlate with other important features. With the increase in dimensions, traditional methods may underperform or even fail when handling high-dimensional datasets. For high-dimensional classification, conventional and popular classifiers such as the Naive Bayes classifier, Linear Discriminant Analysis (LDA), the K-th-Nearest-Neighbor (KNN) classifier, Support Vector Machines (SVM), and Logistic Regression (LR) struggle to cope. Fan et al. (2011) provide a comprehensive overview of these commonly used classification methods, explaining in detail the reasons for their shortcomings in high-dimensional classification problems.

In the process of data collection, mislabelling can occur, and it is a common issue that is often unavoidable. Several factors can cause mislabelling, including manual errors, inaccurate data collection methods, noisy or unreliable sources, or incomplete or insufficient labeling guidelines. mislabelling can disrupt the study and influence decision-making. Although Cannings et al. (2020) have identified conditions under which classifiers such as KNN, SVM, and LDA perform consistently when trained on imperfectly labeled datasets compared to perfectly labeled ones, and they also claim that the discovery of imperfect labels can enhance efficiency. However, this does not hold true for all classifiers. If the classifier is trained on the corrupted dataset, as conducted by Cannings et al. (2020), without addressing the mislabelling issue and using the raw collected data directly, LR is noticeably impacted by mislabelled data. In the simulation example presented in the subsequent section, we have observed that training the LR classifier on a corrupted dataset without handling mislabelling leads to a performance that is no better than random guessing. While the prevailing methods for handling mislabelled data involve either correcting them or deleting them. In this scenario, each label in the dataset must be verified. It should be noted that on the one hand, it is very costly to check each label in the dataset, especially for datasets with large sample sizes. On the other hand, the traditional data-cleaning approaches view mislabels as outliers. However, instead of acting like random outliers, noisy labels behave remarkably similarly to true labels (Li et al., 2017), making it more challenging to identify them.

Among the various classification methods available, we specifically focus on studying the logistic regression (LR) classifier for addressing two-class classification

problems. The challenges of increasing dimensions and mislabelled data often occur concurrently, yet little work has simultaneously discussed these issues. Therefore, our research interest is in enhancing the performance of LR classifiers specifically for sparse data with mislabelling.

To make LR functional for sparse data, variable selection must be considered. Regularisation is a widely used method for variable selection, with several methods such as Least Absolute Shrinkage and Selection Operator (LASSO) (Tibshirani, 1996), Smoothly Clipped Absolute Deviation Penalty (SCAD) (Fan and Li, 2001), Minimax Concave Penalty (MCP) (Zhang, 2010), and adaptive LASSO (Zou, 2006) extensively discussed. To achieve estimator properties such as unbiasedness, sparsity, continuity, and oracle property, we incorporate SCAD in our model. To simultaneously select variables and estimate the unknown parameter in the model, we use the Local Quadratic Approximated(LQA) function proposed by Fan and Li (2001) in our penalised likelihood method.

While the SCAD penalty has shown good performance for low-dimensional sparse data (Fan and Li, 2001), its effectiveness may not be consistent for high-dimensional data. Therefore, alternative methods need to be introduced in the context of high-dimensional data. Fan and Lv (2008) propose Sure Independence Screening (SIS) and its extension, Iterative Sure Independence Screening (ISIS). These methods are designed to effectively reduce dimensions and have been successful in handling high-dimensional or ultra-high-dimensional data. The methods have also shown improvements on previous regularisation methods such as SCAD, LASSO, and MCP, particularly in terms of variable selection performance and model interpretability, especially for high-dimensional data.

1.1.2 Contributions of our method

For corrupted datasets, it is less feasible to scrutinise and correct each label, but it is possible to inspect a subset of the dataset. Hence, we consider a resampling method in our study. We explore the problem by considering the estimation of flipping probabilities. Specifically, our proposed methods involve two-step estimations: the first step estimates the parameters associated with the LR classifier and the flipping probabilities using the resampled data. Then, using the estimates obtained from the

first step as initial values, the second step employs iterative algorithms to estimate these unknown parameters, using both resampled and unresampled data.

We employ regularisation methods specifically designed for low-dimensional sparse data and incorporate SIS or ISIS (Fan et al., 2009, Fan and Song, 2010) into the penalised likelihood method for sparser or high-dimensional data. Through simulated examples, we have observed a remarkable improvement in the performance of the LR classifier when utilising our proposed method, compared to the method that ignores mislabelling in the dataset. This improvement is also evident from the results, where the LR classifier achieves performance that is close to Naive Bayes classifiers and demonstrates competitive performance with classifiers trained on fully corrected labeled datasets. We also discuss alternative methods, including one that resamples part of the data without considering the flipping probabilities during the estimation process, one that assumes perfect knowledge of the parameters associated with flipping probabilities, and one that simplifies the estimation of flipping probabilities based on the mislabelling ratios in the resampled datasets. While some of these methods are easier to implement and some are considered ideal but unrealistic, our proposed method still demonstrates competitiveness among them, as demonstrated by the simulation results. Our proposed methods demonstrate robust performance across various scenarios and excel in variable selection for both low and high-dimensional datasets with mislabelling. Additionally, a real-data study on coronary heart disease prediction demonstrates the superior accuracy of our classifiers compared to alternative methods discussed in this context.

1.2 Research objective 2: Online algorithms for streaming data

1.2.1 Background and motivation

Streaming data refers to data that is continuously generated and sequentially observed, a phenomenon that has notably increased in recent years. Examples of streaming data abound, including COVID test data during the pandemic, data from social media platforms, e-commerce data, real-time stock data, and more. Streaming data can provide researchers with real-time information and the ability to update models in real time for making immediate decisions, such as policy adjustments or trading actions. However, this availability of streaming data also presents new

challenges for traditional offline methods. These challenges involve not only computational efficiency but also the significant demands on storage space and computational resources. In traditional offline scenarios, when new observations are continuously collected, these methods analyse the new data together with the entire historical dataset, which leads to increased time consumption that can impact decision-making. Additionally, these methods necessitate significant storage space to retain detailed information about the observations, placing a considerable burden on resources. Therefore, traditional offline algorithms that repeatedly analyse the entire dataset are no longer suitable for streaming data scenarios. As a result, alternative approaches are required to effectively study streaming data.

To overcome these limitations, online methods have been developed for the analysis of streaming data. Unlike offline methods that repeatedly analyse the entire dataset, online methods process the data in a sequential manner. Each new observation or batch of data is analysed as it arrives and then discarded, leading to improved computational efficiency and optimal resource utilisation. Additionally, online methods often rely on summary statistics instead of retaining all the details of the observations, ensuring the privacy and security of sensitive information.

When the statistical properties align with specific linear structures of the data, their decomposition and updating with streaming data can be relatively straightforward. This property is exemplified in the work of Luo and Song (2020), where they demonstrate it using the example of the sample mean. Specifically, linear regression, which is a specific case of generalised linear models (GLMs), takes advantage of these simplifications. Our study specifically focuses on the development of online algorithms for the more general case of Maximum Likelihood Estimation (MLE). Unlike statistics that follow linear patterns with data, updating MLEs with non-linear forms of data can be more challenging, primarily due to the absence of closed-form solutions. Therefore, our goal is to explore and develop innovative methods for renewable MLEs with streaming data. We place emphasis on achieving computational efficiency without compromising the attainment of sufficient statistics for online statistical inference.

For renewable MLE, the commonly used online algorithms include the first-order online algorithm Stochastic Gradient Descent (SGD) and one of its notable variants known as implicit SGD, proposed by Toulis et al. (2014). Despite the appealing

features of SGD and numerous variants, such as computational efficiency and ease of implementation, these first-order estimation methods fail to provide information for online statistical inference. Specifically, they lack useful information to approximate the full Hessian matrix.

Newton-Raphson, a second-order estimation method, is a traditional and extensively utilised offline method for GLM. Like other offline methods, it poses challenges in terms of computational efficiency and resource demands. Several adaptations have been proposed to address the issue of computing redundancy, with Quasi-Newton methods being one of the notable approaches. These methods aim to estimate the Hessian matrix without the need for calculating the entire matrix at each iteration of the estimation process. In the context of analysing streaming data, Schraudolph et al. (2007) have adapted the BFGS algorithm and a variant called limited storage BFGS. They have named their methods online adapted BFGS (oBFGS) and online limited storage BFGS (oLBFGS), respectively. Bordes et al. (2009) introduce the Quasi-Newton SGD (SGD-QN) method, designed to utilise the second-order information while retaining computational efficiency comparable to SGD. However, all these methods lack helpful information to estimate the full Hessian matrix.

In order to achieve computational efficiency while preserving important statistical information, Luo and Song (2020) propose an improved method. They demonstrate that the summary statistics of their approach are sufficient and result in minimal information loss compared to offline methods. Furthermore, their method demonstrates remarkable computational efficiency, positioning it as a competitive alternative to the traditional Newton-Raphson algorithm. However, their work solely addresses low-dimensional dense data, with no provisions for sparsity.

1.2.2 Contributions of our method

Existing methods for high-dimensional streaming data from GLMs are inadequate in achieving both computational efficiency and sufficient statistical attainment. This gap in the existing methodologies prompts our interest in extending the advantageous method of Luo and Song (2020) to a more general case, considering sparse streaming data and even high-dimensional streaming data problems.

In our study, we introduce several unique approaches. By incorporating regular-

isation methods to reduce dimensions and select variables, our proposed method integrates a penalty term into the model. The LQA function (Fan and Li, 2001) is applied to the incremental updating penalised likelihood algorithm, which enables simultaneous online estimation and variable selection. Given the characteristics of the online method, the approximation for the penalty function with respect to the parameter can be based solely on historical statistics or on the statistics trained using the new data currently involved. Based on this consideration, we propose two penalised online methods. Both methods demonstrate excellent performance in simulation studies, exhibiting high estimation accuracy comparable to offline MLEs, effective variable selection capabilities, and the ability to generate more interpretable models than the offline method. In addition to the proposed penalised online methods, we also incorporate SIS and ISIS (Fan et al., 2009, Fan and Song, 2010) into the approaches. Considering the continuous updating nature of streaming data, we have also developed variants of the screening methods that incorporate historical statistics into the proposed online methods. These variants take into account the evolving nature of the data and utilise historical information for improved performance. In general, these screening methods demonstrate competitive performance in terms of variable selection within the framework of penalised estimation for high-dimensional streaming data. They are able to effectively identify relevant variables while achieving accurate estimation simultaneously.

The capabilities of our proposed methods are evidenced through various simulated scenarios and comparisons with existing techniques. Multiple types of streaming data are examined in this study, including datasets with varying training dataset sizes, different correlations among covariates, diverse distributions of observations, and datasets with different dimensions. The practical application of our methods is further illustrated through the analysis of the real-world dataset from the National Automotive Sampling System Crashworthiness Data System. Specifically, when compared with Luo and Song (2020)'s method, our techniques demonstrate enhanced adaptability in processing various types of data, generating a more interpretable model. When contrasted with the offline method, which is viewed as a benchmark in our study due to its capacity to access all detailed information simultaneously with minimal information loss, our method exhibits competitive performance in estimation, and superior capability in generating concise models for high-dimensional

datasets. Most importantly, our approach needs substantially less computational time and resources to analyse identical datasets.

1.3 Structure of the thesis

In Chapter 2, we conduct an extensive review of the existing literature relevant to the two research topics investigated in our study. This includes a comprehensive review of well-known regularisation methods for variable selection, regularisation methods specifically designed for GLMs, renewable estimation techniques, high-dimensional classification approaches, and methods for mislabelling in classification problems.

In Chapter 3, we present our research on classification with mislabelled data, providing a comprehensive explanation of several proposed methods. We showcase a variety of simulation studies and conduct real data analysis to validate the effectiveness of these methods. The initial section of Chapter 3 provides a detailed breakdown of the structure and components of the topic of *Classification with mislabelled data*.

In Chapter 4, we devote ourselves to an extensive discussion of renewable estimation. This chapter covers detailed methodological descriptions, a wide range of simulation studies, and real-data analyses. The opening section of Chapter 4 provides an expanded explanation of the frameworks and structures that are explored throughout the chapter.

In the concluding chapter, Chapter 5, we reflect on the central findings from our intensive study across both research domains. Furthermore, we delve into potential directions for future research within each of these areas.

1.4 Preliminary notations and assumption of sparsity

Notations: In the subsequent discussions, for any p-dimensional parameter $\boldsymbol{a}$:

- We employ $\|\boldsymbol{a}\|_0$ to denote the L_0 norm, which counts the number of non-zero components in $\boldsymbol{a}$ and has the form as $\|\boldsymbol{a}\|_0 = \sum_{j=1}^{p} 1\{a_j \neq 0\}$;
- $\|\boldsymbol{a}\|_1$ represents the L_1 norm, defined as the sum of the absolute values of the components of $\boldsymbol{a}$: $\|\boldsymbol{a}\|_1 = \sum_{j=1}^{p} |a_j|$;
- $\|\boldsymbol{a}\|_2$ signifies the L_2 norm, which is the square root of the sum of the squared components of $\boldsymbol{a}$: $\|\boldsymbol{a}\|_2 = \sqrt{\sum_{j=1}^{p} a_i^2}$;

- $\|\mathbf{a}\|_\infty$ denotes the L_∞ norm, which is the maximum absolute value among the components of $\mathbf{a}$: $\|\mathbf{a}\|_\infty = \max_{1 \leqslant j \leqslant p} |a_j|$.

Assumption of sparsity: Our study primarily revolves around the concept of sparsity. Drawing from Fan and Li (2001), the term "sparsity" denotes a model characteristic wherein only a small subset of the total variables significantly influence the prediction outcome. This inherent property has given rise to the term "sparse data", which is a focal point of our study. Specifically, we term a parameter vector as "sparse" when only a subset of its components are non-zero. To illustrate, consider p-dimensional vector $\mathbf{a}$ represented as

$$\mathbf{a} = \begin{pmatrix} \mathbf{a_1} \\ \\ \mathbf{a_2} \end{pmatrix}.$$

Here, $\mathbf{a_1}$ is a d-dimensional vector with $1 \leqslant d < p$, containing the non-zero coefficients. Conversely, $\mathbf{a_2}$ is a $(p-d)$-dimensional vector, where every element is zero, $\mathbf{a_2} = \mathbf{0}$.

In this thesis, we produce "sparse data" for our simulation studies. In the context of low-dimensional sparse data settings, the count of zero coefficients of parameters in the model exceeds that of non-zero coefficients by a factor of more than 5. In high-dimensional settings, this ratio escalates to over 50 times or even 1000 times the count of non-zero coefficients. More details can be found in the subsequent sections of simulation study.

2 Literature review

In this section, we provide a comprehensive review of the literature related to our research topic. The references provided below serve as a foundation for the subsequent sections, where we delve into the details of each study and their contributions to our research.

Specifically, in Section 2.1, we explore various regularisation methods for variable selection. We discuss the extensions of regularisation methods for generalised linear models (GLMs) in Section 2.2. In Section 2.3, we delve into the online algorithms used for studying streaming data. Section 2.4 provides a list of references related to high-dimensional classification. Finally, in Section 2.5, we explore the methods used for classification with mislabelling.

2.1 Variable selection

Variable selection has been increasingly widely discussed in regression analysis. Thanks to rapid advances in technology, in areas such as finance, medicine, and genomics, we can collect and store data with larger sample sizes than ever before. In this section, we start with a simple model and consider the linear regression model

$$\mathbf{y} = X^T\boldsymbol{\beta} + \boldsymbol{\epsilon}, \tag{2.1}$$

where $\mathbf{y} = (y_1, \cdots, y_N)^T$ is an N-vector of responses, and X is an $p \times N$ random design matrix and $X_i = (x_{1i}, \cdots, x_{pi})^T$ is the predictor variable also referred to as the covariates in our study, which is a p dimensional vector. $X_i, i = 1, \cdots, N$, are independent and identically distributed (i.i.d.). $\boldsymbol{\beta} = (\beta_1, \cdots, \beta_p)^T$ is a p-vector of the parameters to be estimated, $\boldsymbol{\epsilon} = (\epsilon_1, \cdots, \epsilon_N)^T$ is an N-vector of i.i.d. random errors. Typically, there are only a few predictor variables in a dataset that contribute to the response, which we name as significant or important variables. In this way, it corresponds to the sparsity assumption that $\boldsymbol{\beta} = \begin{pmatrix} \boldsymbol{\beta_1} \\ \boldsymbol{\beta_2} \end{pmatrix}$, where $\boldsymbol{\beta_1}$, a d-vector for $1 \leqslant d < p$, consists of non-zero coefficients and $(\boldsymbol{\beta_2})_{(p-d)\times 1} = \mathbf{0}$. In our study, we refer to predictor variables that have little or no significant impact on the response as "insignificant" or "unimportant" variables (features). We use the term "sparse

data" or "data from a sparse model" to describe this type of data, where the dataset or model contains a substantial number of such variables with little influence on the response variables.

The ordinary least square (OLS) estimator is one of the classical regression methods and is known as the best linear unbiased estimator (BLUE) of β of the following form:

$$\hat{\beta}^{\mathrm{OLS}} = \arg\min_{\beta} \sum_{i=1}^{N} (y_i - X_i^T \beta)^2.$$

While Ordinary Least Squares (OLS) estimates are typically associated with low bias, they can be prone to high variance. Specifically, the OLS method is known for its inability to distinguish between important and non-important predictors, which can lead to less informative and less explanatory models (Tibshirani, 1996). This limitation of OLS has also been demonstrated in a simulation study conducted by Fan and Li (2001).

Collinearity, also known as multicollinearity, is a frequent issue encountered in regression models. It occurs when two or more predictor variables in a statistical model are linearly related to each other (Dormann et al., 2013). The problem of collinearity often accompanies sparse models, especially for high-dimensional datasets where the number of predictor variables is larger than the size of the data. Collinearity has several negative implications for regression models. Firstly, it reduces the accuracy of the parameter estimates, as the presence of collinearity makes it difficult to determine the individual effects of the correlated variables. Secondly, collinearity hampers the interpretability of the model, as it becomes challenging to isolate the specific contributions of each predictor variable and causes an overfitting problem. Additionally, collinearity can lead to unstable and unreliable estimates, and it can cause the classical model, such as OLS, to fail. It is widely acknowledged that the OLS method is not suitable when dealing with highly correlated variables or high-dimensional datasets. In such scenarios, ill-posed problems can occur, specifically non-full-rank matrices, which can result in convergence issues and the failure of the OLS method (Sirimongkolkasem and Drikvandi, 2019). Therefore, alternative approaches are often employed to overcome these challenges and obtain more reliable estimates.

Indeed, the limitations of classical methods such as the OLS method underscore the importance of employing alternative approaches that incorporate variable se-

lection techniques. Variable selection methods play a crucial role in mitigating collinearity issues, reducing noise, mitigating overfitting, and constructing more interpretable models. By identifying and retaining the most relevant predictors, these methods improve model accuracy and enhance our understanding of the underlying relationships in the data. The most commonly studied methods for variable selection include dimension reduction methods such as Principal Component Analysis (PCA), subset selection methods like best subset selection, and regularisation methods that incorporate penalty terms in the model.

PCA is a widely used dimension reduction method that aims to identify linear combinations of the original variables, known as principal components (PCs), to reduce the dimensions of the data. By projecting the data onto a lower-dimensional space spanned by the PCs, PCA can effectively capture the most important patterns and reduce the complexity of the dataset (Lever et al., 2017). However, PCA has certain limitations. It does not perform well for non-linear data, struggles with highly correlated variables, and can result in less interpretable selected models (Lever et al., 2017, Jolliffe and Cadima, 2016).

Best subset selection stands as a classical approach in variable selection. This method systematically examines all conceivable combinations of predictors. Within the framework of best subset selection, the primary objective is to pinpoint the predictor subset that minimises the subsequent equation:

$$\|\mathbf{y} - \mathbf{X}^T\boldsymbol{\beta}\|_2^2 \text{ subject to } \|\boldsymbol{\beta}\|_0 \leqslant p_0, \tag{2.2}$$

where $\|\boldsymbol{\beta}\|_0 = \sum_{j=1}^{p} 1\{\beta_j \neq 0\}$. The goal is to find the subset of predictors, denoted as p_0, where $1 \leqslant p_0 < p$, that minimises the residual sum of squares (Hastie et al., 2017). In contrast, the forward stepwise method starts with an empty model and iteratively adds variables that best improve the fit (Hastie et al., 2017), resulting in a subset of size p_0, where $1 \leqslant p_0 < p$, that minimises the residual sum of squares $\|\mathbf{y} - \mathbf{X}^T\boldsymbol{\beta}\|_2^2$ in (2.2).

The major drawback of the two methods is that they are computationally expensive and less accurate because stochastic errors are usually ignored in the process, which makes their theoretical properties difficult to understand Fan and Li (2001). In addition, although the best subset selection can select the variables and zero the

coefficients, the process is discrete which is unstable even with small variations in the data (Breiman, 1995, Tibshirani, 1996).

A family of thresholds known as penalised likelihood functions has been proposed, which addresses the limitations of methods such as PCA, best subset selection, and stepwise deletion. These penalised likelihood functions offer the advantages of interpretability and facilitate automatic simultaneous variable selection (Fan and Li, 1999). By applying regularisation methods, these functions generate models that are both easier to interpret and more effective in variable selection. In the following context, we outline the most well-known regularisation methods.

Breiman (1995) proposes the non-negative garrote method. The method starts with OLS and reduces some of the coefficients of the OLS estimates to zero by using a non-negative garrote shrinking factors $\mathbf{c} = (c_1, \cdots, c_p)$. The estimate of the non-negative garrote shrinking factors can be found by minimising (2.3) and denoted as $\hat{\mathbf{c}} = (\hat{c}_1, \cdots, \hat{c}_p)$,

$$\frac{1}{2}\|\mathbf{y} - X^T\hat{\boldsymbol{\beta}}^{OLS}\|_2^2 + \lambda \sum_{j=1}^{p} c_j, \text{ s.t. } 0 \leqslant c_j, j = 1, \cdots, p, \tag{2.3}$$

where the given $\lambda > 0$ is a tuning parameter. The non-negative garrote estimate has the form of

$$\hat{\boldsymbol{\beta}}^{NNG} = \hat{\mathbf{c}}\hat{\boldsymbol{\beta}}^{OLS}.$$

Shrinking the dataset, the non-negative garrote method provides more accurate estimates than the OLS method, and it can also compete with ridge regression (discussed below) if there are not too many non-zero coefficients in the model (Tibshirani, 1996). However, when analysing datasets with highly correlated variables or high-dimensional datasets, the non-negative garrote method has the same limitations as the OLS method due to its reliance on OLS.

Frank and Friedman (1993) introduce bridge regression, one of the classical regularisation methods, also named as the L_q norm penalty function of the form $\sum_{j=1}^{p} |\beta_j|^q$. Ridge regression proposed by Hoerl and Kennard (1970) and Least Absolute Shrinkage and Selection Operator (LASSO) proposed by Tibshirani (1996) are both special cases of L_q norm penalty functions. Ridge regression is a L_2 regularisation function and LASSO is a L_1 regularisation function.

The ridge regression method has the following form for estimating β:

$$\hat{\beta}^{\text{Ridge}} = \arg\min_{\beta} \sum_{i=1}^{N} (y_i - X_i^T \beta)^2 + \lambda \|\beta\|_2^2,$$

where $\lambda > 0$ is a tuning parameter. The main disadvantages of ridge regression are that the complexity of the regression equation is the same as OLS (Breiman, 1995), and that ridge regression cannot select variables.

Using LASSO, the estimate of β has the following form:

$$\hat{\beta}^{\text{LASSO}} = \arg\min_{\beta} \sum_{i=1}^{N} (y_i - X_i^T \beta)^2 + \lambda \|\beta\|_1,$$

where $\lambda > 0$ is a tuning parameter. As can be seen from its form, LASSO does not explicitly rely on OLS and therefore improves on the shortcomings of the non-negative garrote method. LASSO also has the beneficial properties of subset selection and ridge regression (Tibshirani, 1996), which ensure that it finds interpretable models and produces consistent estimates. Although LASSO has many advantages over previous methods in terms of computation time, estimation accuracy, and model interpretations, it cannot analyse highly correlated variables and high-dimensional datasets (Meinshausen, 2007, Wang et al., 2011).

Since its introduction, LASSO has been widely discussed and has inspired a great deal of further research. The elastic net approach proposed by Zou and Hastie (2005) combines the L_1 and L_2 norm penalty functions, from which the good properties are derived. Meinshausen (2007) introduces relaxed LASSO with two regularisation parameters that take care of the variable selection and the shrinkage level respectively and the results show that relaxed LASSO performs better than LASSO when the ratio of signal to noise is high, achieving more accurate predictions and sparser estimates with a less complicated model. Wang et al. (2011) introduces random LASSO which eliminates highly correlated variables more efficiently and has a more flexible estimation process than the elastic net, but it is a computationally intensive method.

Fan and Li (2001) introduce the following three properties for a desirable estimator derived from a good penalty function:

- Unbiasedness: When the true unknown parameters are large, the estimates are almost unbiased to avoid excessive estimation bias.
- Sparsity: When the true unknown parameters are small, the estimated coefficients are set to zero to reduce the complexity of the model.
- Continuity: The resulting estimator is continuous to achieve model stability.

Considering the above properties, for bridge regression, the solution is continuous only if $q \geqslant 1$. Furthermore, bridge regression cannot select variables when $q > 1$, which can cause problems if there are more predictor variables than observations (Fan and Li, 2001). When $q = 1$, L_1 penalty function known as LASSO produces biased estimates of the large parameters (Fan and Li, 2001, Zou, 2006). Therefore, no L_q penalty function can satisfy all three conditions at the same time.

The oracle property, a desirable feature for estimators in high-dimensional settings, has been studied in the context of penalised likelihood methods by Fan and Li (2001). An oracle estimator is defined as one derived from a process where the indices of the non-zero coefficients of the parameter are known a priori. Consider a sparse parameter vector

$$\boldsymbol{\beta} = \begin{pmatrix} \boldsymbol{\beta_1} \\ \boldsymbol{\beta_2} \end{pmatrix}_{p \times 1},$$

where $\boldsymbol{\beta_1}$ represents the non-zero coefficients of $\boldsymbol{\beta}$ with a fixed, finite dimension d_1 such that $1 \leqslant d_1 \ll p$, and $\boldsymbol{\beta}_2 = \mathbf{0}$.

If $\hat{\boldsymbol{\beta}}$ is an estimator obtained without prior knowledge of the non-zero coefficient indices of $\boldsymbol{\beta}$, partitioned as $\hat{\boldsymbol{\beta}} = \begin{pmatrix} \hat{\boldsymbol{\beta}}_1 \\ \hat{\boldsymbol{\beta}}_2 \end{pmatrix}_{p \times 1}$, then it possesses the following properties:

- Sparsity: The estimated coefficients for irrelevant predictors are zero, which is $\hat{\boldsymbol{\beta}}_2 = \mathbf{0}$;
- Asymptotic normality: As the sample size N grows, $\sqrt{N}(\hat{\boldsymbol{\beta}}_1 - \boldsymbol{\beta}_1) \xrightarrow{\mathcal{D}} \mathcal{N}(0, \Sigma)$, where Σ is the covariance matrix of the true nonzero coefficients in the subset model.

Zou (2006) and Zhao and Yu (2006) confirm Fan and Li (2001)'s conjecture that LASSO does not have the oracle property. Zou (2006) gives a necessary condition

for the consistency of variable selection of LASSO and Zhao and Yu (2006) prove an almost necessary and sufficient condition named Irrepresentable Condition which shows that LASSO has the consistency of model selection when there are less correlated variables.

A nonconcave penalised likelihood method named Smoothly Clipped Absolute Deviation Penalty (SCAD) proposed by Fan and Li (2001) has unbiased, sparse, and continuous properties at the same time and it performs as well as oracle estimators.

The estimator by SCAD of the nonzero coefficients of $\boldsymbol{\beta}$ in (2.1) is

$$\hat{\boldsymbol{\beta}}_1^{\mathrm{SCAD}} = \arg\min_{\boldsymbol{\beta}} \sum_{i=1}^{N} (y_i - X_i^T \boldsymbol{\beta}_1)^2 + \sum_{j=1}^{d} p_\lambda^{\mathrm{SCAD}}(|\beta_j|),$$

where the penalty function is

$$p^{\mathrm{SCAD}}(|\beta_j|) = \begin{cases} \lambda|\beta| & \text{if} \quad |\beta| < \lambda, \\ \frac{2a\lambda|\beta| - \beta^2 - \lambda^2}{2(a-1)} & \text{if} \quad \lambda < |\beta| < a\lambda, \\ \frac{\lambda^2(a+1)}{2} & \text{otherwise,} \end{cases} \qquad (2.4)$$

$$\text{for some } a > 2,$$

and both a and λ are tuning parameters. For small coefficients, the SCAD penalty has the same penalisation as LASSO, and for large coefficients, the SCAD penalty reduces the penalisation continuously and shrinks the estimates less heavily. In this way, SCAD improves LASSO's problem of causing biased estimates.

Zou (2006) proposes an adaptive LASSO that is an unbiased, sparse, and continuous penalty function. The estimator of nonzero coefficients of $\boldsymbol{\beta}$ is

$$\hat{\boldsymbol{\beta}}_1^{\mathrm{AdaLASSO}} = \arg\min_{\boldsymbol{\beta}} \sum_{i=1}^{N} (y_i - X_i^T \boldsymbol{\beta}_1)^2 + \lambda \sum_{j=1}^{d} \hat{w}_j |\beta_j|,$$

where λ is the tuning parameter, $\|\boldsymbol{\beta}_1\|_1 = d$ and $\hat{\boldsymbol{w}} = (\hat{w}_1, \cdots, \hat{w}_d)$, is a known weight vector. Zou (2006) uses the OLS estimator to define the weight vector and other estimates obtained by LASSO, ridge regression, or other methods can also be used to define the weight vector. To improve the bias of estimates from LASSO, adaptive

LASSO reduces the bias of estimates by introducing the weight vector. It has also been proved that adaptive LASSO has the oracle property (Zou, 2006). As with SCAD penalty, Adaptive LASSO has the problem of zero absorbing state (Fan and Lv, 2010) meaning that once the estimate has been shrunk to zero, it cannot be changed, which may cause the problem of misjudging the nonzero coefficients to be zeros during the process and this cannot be corrected later.

Zhang (2010) proposes a nearly unbiased and continuous penalised variable method that consists of a Minimax Concave Penalty (MCP) and a penalised linear unbiased selection (PLUS) algorithm and has the advantage of efficient computation. With a similar approach to the SCAD penalty, the MCP penalty function is

$$p^{MCP}(|\beta_j|) = \begin{cases} \lambda(|\beta_j| - \frac{\beta_j^2}{2\gamma\lambda}) & \text{if} \quad |\beta_j| < \lambda\gamma, \\ \frac{\lambda^2\gamma}{2} & \text{otherwise}, \end{cases}$$

where $j = 1, \cdots, p$, and $\gamma > 0$ and $\lambda > 0$ are tuning parameters. Zhang (2010) proves that MCP provides sparse convexity to the broadest extent by minimising the maximum concavity and the estimator achieved by the MCP function has the oracle property.

A solution named Dantzig Selector (DS), L_1 regularisation, which minimises the maximum component of the gradient of the squared error function, is proposed by Candes and Tao (2007).

$$\hat{\boldsymbol{\beta}}^{DS} = \arg\min_{\boldsymbol{\beta}} \|\boldsymbol{\beta}\|_1 \text{ subject to } \|X(\mathbf{Y} - X^T\boldsymbol{\beta})\|_\infty \leqslant \lambda,$$

where λ is a tuning parameter. Uniform uncertainty principles (UUP) is described in Candes and Tao (2007)'s work. If covariate matrix X meets UUP, X is with unit-normed columns, and the estimator from the DS method can achieve the ideal risk which is a loss within a logarithmic factor $\log(p)$ of the ideal mean squared error compared to the oracle estimator.

There are algorithms for variable selections which are frequently used such as Local Quadratic Approximation (LQA) (Fan and Li, 2001), least angle regression (LARS) (Efron et al., 2004) and minorize–maximize (MM) algorithm (Hunter and Li, 2005). As for DS, it uses a primal-dual interior point algorithm which is competitive

in computing efficiency with other existing methods when the dimension is not ultra-high (Cai and Lv, 2007).

In addition to its computational advantage, as DS has the property of L_1- minimisation and compared with all the previous feasible solutions, it can obtain the "sparsest" solution over the space (Cai and Lv, 2007). Efron et al. (2007) compare the performance of the regularisation paths, solution paths, and prediction accuracy of LASSO and DS, and the results show the two methods have identical performance while LASSO outperforms DS slightly in prediction accuracy. However, Cai and Lv (2007) mention that the algorithm for DS is sensitive to its initial values and it cannot work effectively for high-dimensional or ultra-high-dimensional data. The ideal risk of DS is within a logarithmic factor $\log(p)$ but the exact minimax factor was unknown. Like other L_1 regularisations, DS creates biased estimators while adaptive LASSO adjusts the weights of the penalties. SCAD and MCP, which have flat tail functions, can achieve unbiased estimators (Fan and Li, 2001, Zou, 2006, Zhang, 2010).

The collinearity and cumulative error problems become more severe with the addition of dimensions. The previous methods cannot perform as well when dealing with low-dimensional regression problems and can fail to select variables. In addition, the process can become computationally expensive as the increase of dimensions. For example, LASSO and stepwise deletion cannot work well for high-dimensional or ultra-high-dimensional data. Motivated by Candes and Tao (2007)'s work, Fan and Lv (2008) highlight some concerns about the flaws of DS and mentioned four noteworthy problems related to high dimensional data and ultra-high dimensional data.

- When $p > N$ or $p >> N$, the computation issue of existing algorithms is always a problem;
- The ideal loss of the estimator from the DS method is within $\log(p)$, which cannot be ignored and becomes less accurate with the number of dimensions increasing;
- UUP is hard to verify and guarantee, which is presented in a simulation study by Fan and Lv (2008);
- The estimator from DS has not been proven to have the oracle property.

In consequence, a new variable selection method is needed to solve the problems for high-dimensional or ultra-high-dimensional data.

In a high dimensional linear regression setting and when considering prediction, variable selection, and variable ranking, Wang et al. (2020) design a large-scale simulation study to compare the performance of the commonly used variable selection methods including LASSO, adaptive LASSO, elastic net, ridge regression, SCAD, and DS. Their conclusion is that there is no outright winner from these discussed methods in all scenarios and pointed out that the relative performance of each method depends on many factors and the interests of researchers. However, they do not recommend DS because its performance is usually inferior or similar to LASSO and it is more computationally costly for high-dimensional and ultra-high-dimensional data.

Fan and Lv (2008) introduce Sure independence screening (SIS) which is an extension of the two-sample t-statistic. SIS is a model-free method that does not need the model structure of the regression function, and it finds the important variables by correlation learning and only uses the ranking information. SIS is a simple but efficient method that can shrink the data set from ultra-high dimensions to moderate models efficiently and accurately. For the linear model (2.1), when all the covariates are standardised to have a mean of 0 and a standard deviation of 1, we can employ componentwise regression to obtain a vector $\boldsymbol{z} = (z_1, \cdots, z_p)^T$ and rank the marginal correlations, where

$$z_j = X_j \mathbf{y}, \text{ for } j = 1, \cdots, p,$$

and $X_j = (x_{j1}, \cdots, x_{jN})$. Then we have a submodel

$$\mathcal{M}_\gamma = \{1 \leqslant j \leqslant p : |z_j| \text{ is among the first } [\gamma N] \text{ largest of all}\}, \quad \text{for any } \gamma \in (0, 1),$$

where $\mathcal{M}_* = \{1 \leqslant j \leqslant p : |\beta_j| \neq 0\}$, is the true sparse model with the size of $|\mathcal{M}_*| = s$. Sure screening property, a desirable property for variable selections, means that the important features of the model are contained in the subset with probability tending to 1 and it is defined as

$$P(\mathcal{M}_* \in \mathcal{M}_\gamma) \to 1, \text{ as } N \to \infty. \tag{2.5}$$

Fan and Lv (2008) offer a comprehensive proof asserting that their proposed SIS method, based on correlation learning, possesses the sure screening property as defined in (2.5). In contrast, the stepwise deletion method lacks this property (Fan and Lv, 2008). Saldana and Feng (2018) show that SCAD has the sure screening property for the designed relatively easy examples. However, SCAD does not perform as well as SIS or iterative sure independence screening (ISIS) (Fan and Lv, 2008) introduce below, in estimation errors and model sizes. In both Fan et al. (2009) and Saldana and Feng (2018)'s simulated examples, LASSO misses the important feature and performs the worst among other variable selection methods. Zhang et al. (2019) prove that under reasonable assumptions, even though without the assumption of the marginal correlation, ISIS still has the sure screening properties.

SIS usually works with other variable selection methods and it can significantly improve other methods in computational speed and estimation accuracy significantly, especially for ultra-high dimensional data problems. This process is carried out in a two-stage process. At first, SIS finds the important features and reduces the dimensions. Secondly, a penalised likelihood method such as LASSO, SCAD, DS, and so on is used for the new data set consisting of the selected variables from the first stage. In Fan and Lv (2008)'s simulated examples, compared with the performance of LASSO and DS, after using SIS to reduce the dimension at first, both LASSO and DS's performance get noticeably improved. The two-stage variable selection process is named after SIS and the penalised likelihood method. SIS-SCAD outperforms SIS-DS, SIS-DS-SCAD, and SIS-DS-Adaptive LASSO in Fan and Lv (2008)'s study. Fan and Lv (2008) also pointed out that when considering the accuracy of estimates, SCAD has fewer estimation errors than adaptive LASSO. It is worth mentioning that SIS can not only be applied to analyse high-dimensional and ultra-high-dimensional data but also can be used for solving low-dimensional data problems.

SIS may fail to find important features when these important features are correlated with responses but not marginally related, when less important features are highly marginally related to some important features, or when the correlation between predictors is high (Fan and Lv, 2008). ISIS is introduced by Fan and Lv (2008) to improve the defects of SIS. ISIS uses the joint covariates information and selects features through a penalised pseudo-likelihood method in the iterative process. Introducing the iterative process, ISIS also improves the zero absorbing state

problem of SIS to some extent (Fan et al., 2009). Fan et al. (2009) propose variants of ISIS, that delete features during the iterative process and these variants achieved lower false selection rates, based on how many unimportant features are selected when compared with vanilla ISIS.

Fan and Lv (2018) give a comprehensive overview of SIS and ISIS in the context of the linear model and the generalised linear model. Fan et al. (2009) and Fan and Song (2010) also extend SIS and ISIS for generalised linear models, which we give discussion in more detail in the following section.

Since the introduction of SIS and ISIS, numerous methods emerge for variable screening. Wang (2009) explores the classical method, Forward Regression (FR), assessing its suitability for variable screening in linear regression contexts. FR consistently identifies variables for screening, yet its predictive accuracy lags behind that of SIS in real data analysis study (Wang, 2009).

Wang (2012) presents the Factor Profiled Sure Independence Screening (PIS). This method integrates factor modelling into SIS. By integrating factor modelling, which accounts for insignificant variables, PIS overcomes a notable limitation of SIS: overfitting. Through factor profiling, PIS enhances the consistency of SIS in model selection. Building on this, Naifei et al. (2020) develop a variant of PIS, having a double decorrelation process, named as preconditioned PIS (PPIS). This procedure ensures that profiled predictors remain asymptotically uncorrelated. Though both PIS and PPIS prove effective for high-dimensional datasets with highly correlated predictors, their sole reliance on predictors for factor profiling and preconditioning obscures the clarity in interpreting the response variables (Wang, 2012).

He et al. (2013) address nonlinear, high-dimensional, heterogeneous data with censored responses. They unveil the Quantile-adaptive model-free feature screening method. After further refining it to address situations where predictors, though marginally insignificant individually, have a collective influence on the response variables, they name it QaSIS.

Shao and Zhang (2014) introduce the Martingale difference correlation, an natural extension of the distance correlation measurement originally presented by Székely et al. (2007). This metric is designed to identify significant variables in high-dimensional data. The consistency of its screening properties has been confirmed (Shao and Zhang, 2014).

Kong et al. (2017) critique the SIS methodology, spotlighting its shortcoming in detecting variables that, though marginally uncorrelated, show joint correlation with the response. This gap arises from SIS's dependence on marginal correlation learning, which neglects the joint information between the response and the set of covariates. In response, they propose a technique centred on ranking canonical correlations, which signify correlations between chosen covariates and the response. While this method boosts feature selection precision with a more extensive covariate set, it intensifies computational challenges, given the inherent intricacy of canonical correlation calculations.

2.2 Penalised generalised linear model

McCullagh (1983) extends the classical linear model to a more general case where the response variables are assumed to follow an exponential family distribution and the name of the model is generalised linear model (GLM). A classical linear model has the assumption of residuals that follow conditional normal distributions while GLM loosens this restriction and studies distributions from an exponential family where there are various distributions of residuals and the normal distribution is one special case. The assumption of response variables of a classical linear model is continuous while GLM can study continuous data as well as discrete data.

We assume that we have i.i.d. observations $(X, \mathbf{y})$, $X = (X_1, \cdots, X_N)$, where $X_i = (x_{1i}, \cdots, x_{pi})^T$, $i = 1, \cdots, N$, and with a canonical link, the density function of each component of $\mathbf{y}$, $\mathbf{y} = (y_1, \cdots, y_N)^T$, taking the form

$$f_Y(y_i; X_i, \boldsymbol{\beta}) = \exp\{(y_i X_i^T \boldsymbol{\beta} - b(X_i^T \boldsymbol{\beta}))/a(\phi) + c(y_i, \phi)\}, \tag{2.6}$$

where $a(\cdot)$, $b(\cdot)$ and $c(\cdot)$ are known functions, ϕ is a dispersion parameter and $\boldsymbol{\beta}$ is the unknown parameter we are interested in. We are looking for the maximum likelihood estimator (MLE) of $\boldsymbol{\beta}$ denoted as $\hat{\boldsymbol{\beta}}$.

Sparse generalised linear models are widely studied and used. Where the database has correlated variables or the number of features is larger than the number of observations, then the classical maximum likelihood estimator (MLE) fails to find the estimator of $\boldsymbol{\beta}$ in these cases. In our study, we use penalised likelihood methods

to select the important features for generalised linear regressions. And we have the following form of the penalised MLE,

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \left\{ \ell(\mathbf{y}, \boldsymbol{\beta}) - N \sum_{j=1}^{p} p_\lambda(|\beta_j|) \right\},$$

where $p_\lambda(\cdot)$ is a penalty function. The log-likelihood function $\ell(\mathbf{y}, \boldsymbol{\beta})$ takes the following form

$$\ell(\mathbf{y}, \boldsymbol{\beta}) = \sum_{i=1}^{N} \ell(y_i, \boldsymbol{\beta}), \tag{2.7}$$

where

$$\ell(y_i, \boldsymbol{\beta}) = y_i X_i^T \boldsymbol{\beta} - b(X_i^T \boldsymbol{\beta}).$$

LQA is used for the penalty functions that are singular at the origin (Fan and Li, 2001). When $\beta_j \neq 0$, $j = 1, \cdots, p$, we have

$$[p_\lambda(|\beta_j|)]' = p_\lambda(|\beta_j|)\mathrm{sgn}(\beta_j) \approx \{p_\lambda(|\beta_{j0}|)/|\beta_{j0}|\}\beta_j, \quad \text{for } \beta_j \approx \beta_{j0}.$$

When $\beta_{j0} \approx 0$, we have $\hat{\beta}_j = 0$, $j = 1, \cdots, p$.

Using LQA, the iterative processes can be carried out for the penalised log-likelihood estimations. LQA has the advantage of reducing the computational burden. The disadvantage associated with this method is that it has zero absorbing state (Fan and Li, 2001, Fan and Lv, 2010).

We also consider the GLM with ultra-high dimensional data, where $p >> N$ and SIS and its extension ISIS are applied to study the problem. Based on Fan and Lv (2008)'s work within the context of linear regressions shown in the previous section, Fan et al. (2009) and Fan and Song (2010) extend SIS and ISIS for generalised linear regressions. An independence learning method by ranking the maximum marginal likelihood estimator or maximum marginal likelihood itself for GLMs is introduced. The application of the SIS method to GLMs has been demonstrated to retain the sure screening property (2.5) (Fan and Song, 2010). It also has been proved that although there exists bias between the marginal models and the joint model,

under a mild condition, the marginal model can deliver the non-sparse information of the joint model (Fan and Song, 2010). In the context of high-dimensional and ultra-high-dimensional GLMs data analysis, Fan et al. (2009), Fan and Song (2010) have shown that SIS or ISIS greatly enhance the efficacy of classical regularisation methods, notably the widely used LASSO and SCAD.

2.3 Renewable estimation

The data that can be observed in a sequence has been extensively studied in recent years, for example, traffic information, stock market data, e-commerce purchases, and so on. This type of data which comes in a sequence is called the streaming data and the estimators renewed with the online learning method are called renewable estimators in our study. The traditional batch learning or offline learning method is not well-suited for analysing streaming data. This is due to several reasons. Firstly, it requires waiting for the entire data to be collected before analysis can begin, which can be time-consuming and may result in delays in decision-making. Secondly, the storage requirements for storing the entire dataset can be substantial, especially when dealing with large volumes of streaming data. Thirdly, analysing the entire training dataset at once can be computationally expensive, requiring significant computational resources. Lastly, storing all the details of the collected observations can consume large amounts of storage space.

Online learning is highly advantageous for analysing streaming data as it effectively tackles the computational and storage burdens associated with offline learning. Through online learning, the summary statistics are utilised without revisiting previous data, which allows for the discarding of historical data. This approach enables more efficient use of computational resources. This not only improves the computational speed compared to offline methods but also results in notable storage savings. Additionally, online learning mitigates privacy and confidentiality concerns by leveraging historical statistics instead of retaining all detailed data, thus ensuring data security and protecting sensitive information.

For online algorithms, if the statistics take certain linear functions of data, the updates can be easily transferred to a function consisting of the summary statistics of previous data and the new data rather than the historical raw data (Luo and Song,

2020). A more general case where the response variables come from the exponential family distribution and the MLE updated by online learning is more challenging. In this section, we make the same assumption about the data as the one described in the model (2.6).

Gradient Descent (GD), one of the traditional offline optimisation methods, has its estimator at k-th time iteration, where $k = 1, \cdots$, as follows,

$$\hat{\boldsymbol{\beta}}^{\mathrm{GD}(k)} = \hat{\boldsymbol{\beta}}^{\mathrm{GD}(k-1)} - \alpha \nabla \ell(\mathbf{y}, \hat{\boldsymbol{\beta}}^{\mathrm{GD}(k-1)}),$$

where α is the learning rate and $\ell(\cdot)$ is the log-likelihood function. The estimate of $\boldsymbol{\beta}$ is obtained when the iteration reaches convergence.

Sakrison (1965) studies the iterative likelihood estimation process using the idea of stochastic approximation proposed by Robbins and Monro (1951). The method is named as explicit stochastic gradient descent and is also named as Stochastic Gradient Descent (SGD) for short. SGD has become one of the most frequently used methods in machine learning since it was invented. Compared with GD, SGD, an online method, analyses one or a subset of the data set and has the obvious advantages of computational simplicity and asymptotic efficiency.

Using SGD, when new observations are collected at the t-th time, where $t = 1, \ldots,$ the estimator is updated as follows:

$$\hat{\boldsymbol{\beta}}_t^{\mathrm{SGD}} = \hat{\boldsymbol{\beta}}_{t-1}^{\mathrm{SGD}} - \alpha_{t-1} \nabla \ell(\mathbf{y}_t, \hat{\boldsymbol{\beta}}_{t-1}^{\mathrm{SGD}}), \tag{2.8}$$

where α_{t-1} is the learning rate and $\nabla \ell(y, \boldsymbol{\beta})$ is the first gradient of (2.7) with respect to $\boldsymbol{\beta}$.

In Toulis et al. (2014)'s work, implicit SGD is studied. When new observations are collected at the t-th time, the estimator is updated according to the following formula, which involves the estimator at the t-th time on both sides of the equation.

$$\hat{\boldsymbol{\beta}}_t^{\mathrm{im}} = \hat{\boldsymbol{\beta}}_{t-1}^{\mathrm{im}} - \alpha_t \nabla \ell(\mathbf{y}_t, \hat{\boldsymbol{\beta}}_t^{\mathrm{im}}), \tag{2.9}$$

where α_t is the learning rate.

In comparison to the implicit stochastic gradient descent (SGD) estimator, the explicit SGD estimator has the advantage of faster convergence. However, the explicit

SGD estimator is often less stable when applied to small-scale data and is more sensitive to the misspecification of the learning rate. As a result, Toulis et al. (2014) recommend using the implicit SGD estimator over the explicit SGD estimator.

From (2.8) and (2.9), it should be noted that both SGD and implicit SGD update the model parameters based on the gradient of the log-likelihood from the newest observation alone. While these approaches offer computational efficiency, challenges arise, especially in non-stationary data streams. In such scenarios, where the underlying data distribution is dynamic and can shift over time, relying solely on the most recent observations can be inadequately encapsulate the intricacies of past data.

Many variants of the first-order SGD has emerged. Konečnỳ and Richtárik (2013) propose Semi-Stochastic Gradient Descent (S2GD) which gains the properties of stability and the fast convergence from GD and holds the computational efficiency of SGD (Konečnỳ and Richtárik, 2013). It should be noted that the estimator from S2GD is unbiased. Polyak and Juditsky (1992) introduce Averaged stochastic gradient descent (ASGD) and Tran et al. (2015) extend ASGD to averaged implicit stochastic gradient descent (AISGD) that is more stable than explicit SGD. Fang (2019) introduces the perturbation-based resampling procedure to fill the gap of Tran et al. (2015)'s work on the construction of interval estimations for implicit SGD and AISGD. While work by Luo and Song (2020) that uses simulated examples to show this perturbation-based resampling procedure fails to achieve statistical efficiency to construct online inference, which generates less accurate estimated standard errors (ESEs) and shows 0% coverage probability.

First-order optimisation is simpler to compute because it avoids needing to calculate the inverse of the Hessian matrix at each iteration. However, second-order methods have faster convergence than first-order methods because of the adaptation of curvature information. In addition, first-order online methods, such as SGD and its variants mentioned above, cannot achieve the construction of online inferences due to the limited information provided by the statistics.

To illustrate online second-order algorithms, we first provide an overview of the second-order offline method. Newton-Raphson is a classical and widely used second-order method for offline estimation, known for its advantageous feature of converging at a quadratic rate (Ypma, 1995). At the k-th iteration, where $k = 1, \ldots,$

the estimate takes the following form:

$$\hat{\boldsymbol{\beta}}^{\mathrm{NR(k)}} = \hat{\boldsymbol{\beta}}^{\mathrm{NR(k-1)}} - [\mathbf{H}(\hat{\boldsymbol{\beta}}^{\mathrm{NR(k-1)}})]^{-1}\nabla\ell(\mathbf{y}, \hat{\boldsymbol{\beta}}^{\mathrm{NR(k-1)}}), \tag{2.10}$$

where $\mathbf{H}(\boldsymbol{\beta}) = \nabla^2\ell(\mathbf{y}, \boldsymbol{\beta})$, is known as Hessian matrix. The MLE of $\boldsymbol{\beta}$ is achieved when the iteration reaches convergence.

In addition to limitations shared by other offline methods, such as the requirement for large storage spaces and repeated computation of historical data, the Newton-Raphson method also has a notable drawback. This method involves computing the inverse of the full Hessian matrix at each iteration, which can lead to computational inefficiency, particularly when dealing with datasets that have a large number of predictors.

To mitigate this computational redundancy, Quasi-Newton methods have been introduced. Quasi-Newton methods are variants of the Newton-Raphson method that do not compute the full Hessian matrix. Instead, they approximate the inverse of the Hessian matrix with some other positive definite matrix that ensures a good approximation to the Hessian matrix (Mishra et al., 2019). This approximation improves computational efficiency. The most well-known Quasi-Newton algorithms are Broyden's algorithm and the Broyden–Fletcher–Goldfarb–Shanno (BFGS) algorithm, which avoids computing the second derivative of the loss function during the iteration of estimation and improve the computational ability of the Newton-Raphson method. Although BFGS reduces the computational cost of the Newton-Raphson method, the need for a large memory is still a significant drawback. To address this, the limited-memory BFGS algorithm is proposed by Liu and Nocedal (1989).

These Quasi-Newton offline methods have been adapted for online algorithms due to their computational efficiency. Schraudolph et al. (2007) adapt BFGS and limited storage BFGS for analysing streaming data, naming their methods online adapted BFGS (oBFGS) and online limited storage BFGS (oLBFGS). Inspired by oLBFGS, Bordes et al. (2009) propose the Quasi-Newton SGD (SGD-QN) method, which is designed to use second-order information and is as computationally efficient as SGD. It should be noted that these methods do not fully explore the exact information of the Hessian matrix. Specifically, SGD-QN only estimates and updates the diagonal matrix of the Hessian matrix, and it does not provide useful information

for online statistical inference.

In addition, there are other online second-order methods, such as the natural gradient algorithm (Amari et al., 2000) and the online Newton step (Hazan et al., 2007), which are efficient in statistical inference as they retain the full information of the Hessian matrix. However, a drawback of both methods is that they require large memory spaces to store all the detailed information of matrices during the update process, which can be a non-negligible disadvantage.

In order to achieve computational efficiency and enable statistical inference in regression analysis of streaming data, Luo and Song (2020) propose an incremental updating algorithm within the framework of GLMs to refine existing methods. They implement this algorithm using the powerful computing platform "Apache Spark" (Bifet et al., 2015) and introduce a new computing platform called the "rho architecture." This architecture allows for the accommodation of inference-related statistics and the renewal of both estimation and inference statistics.

Specifically, Luo and Song (2020) approximate the first derivative of the loss function by employing a first-order Taylor series expansion centred around the historical MLEs obtained from the previous batch of streaming data. This approximation helps in reducing computational complexity while still preserving the necessary information for estimation. Additionally, they update the Hessian matrix using only the new data and the historical Hessian matrix, further enhancing computational efficiency. In detail, for the method proposed by Luo and Song (2020), in the case of streaming data generated from GLMs, the estimation process at the k-th iteration, where $k = 1, \cdots$, when the b-th data batch is collected, where $b = 2, \cdots$, can be expressed as (2.11). It is worth mentioning that when the first data batch is observed ($b = 1$), the estimation process is the same as the offline method.

$$\tilde{\boldsymbol{\beta}}_b^{(k)} = \tilde{\boldsymbol{\beta}}_b^{(k-1)} - [\tilde{\mathbf{H}}_b(\tilde{\boldsymbol{\beta}}_{b-1})]^{-1}\left\{\left[\tilde{\mathbf{H}}_{b-1}(\tilde{\boldsymbol{\beta}}_{b-1})\right](\tilde{\boldsymbol{\beta}}_b^{(k-1)} - \tilde{\boldsymbol{\beta}}_{b-1}) + \mathbf{S}_b(\tilde{\boldsymbol{\beta}}_b^{(k-1)})\right\}, \quad (2.11)$$

where $\mathbf{S}_b(\boldsymbol{\beta}) = \nabla\ell(\mathbf{y}_b, \boldsymbol{\beta})$, $\tilde{\mathbf{H}}_b(\boldsymbol{\beta}) = \sum_{l=1}^{b} \mathbf{H}_l(\boldsymbol{\beta}_l)$, and $\tilde{(\cdot)}$ indicates the estimate is derived from the incremental algorithm. The renewable estimate of $\boldsymbol{\beta}$, denoted as $\tilde{\boldsymbol{\beta}}_b$, is obtained after the algorithm converges, and the subscript b corresponds to the number of the collected data batch.

According to Luo and Song (2020), the estimate $\tilde{\boldsymbol{\beta}}_b$ obtained from their renewable

estimation method approximates the MLE of β obtained from the offline Newton-Raphson method, with second-order asymptotic errors. They have also demonstrated that the estimated variance of $\tilde{\beta}_b$ yields the same estimated standard error (ESE) as the MLE obtained from the offline method that analyses all the collective data at once. Importantly, the renewable estimation method proposed by Luo and Song (2020) is evidently more time- and resource-efficient compared to the traditional Newton-Raphson method.

Luo and Song (2020)'s renewable estimation method provides an efficient way to process data by only maintaining summary statistics from historical data. This significantly expedites calculations compared to the traditional Newton-Raphson method. Importantly, they demonstrate that the compromise in estimation efficiency compared to MLE obtained by the offline method is negligible. Furthermore, the summary statistics of historical data act as sufficient statistics and can be used for tasks such as the construction of online confidence intervals. Although the method proposed by Luo and Song (2020) has shown competitiveness compared to previous methods, Luo and Song (2020)'s study is confined to the renewable estimation method for low-dimensional dense data, which is less practical for handling sparse high-dimensional data.

High-dimensional streaming data has garnered increasing attention and is more and more prevalent in various fields, presenting unique computational and storage challenges. Vijayakumar and Schaal (2000) incorporated Locally Weighted Projection Regression (LWPR) and assert that their proposed method is the first to merge localised learning with an incremental algorithm for sparse models, thereby facilitating effective handling of high-dimensional data. Rosasco and Villa (2015) introduced an iterative regularisation algorithm for least squares, which does not rely on explicit penalisation. Their method predefines all parameters, with the number of algorithm iterations controlling the amount of regularisation applied to the model. However, the methods do not have explicitly addressed the issue of providing enough information to obtain sufficient statistics.

Losing et al. (2018) offer a comprehensive review of prevalent online algorithms for studying high-dimensional data. The reviewed algorithms include the Incremental Support Vector Machine (ISVM), an online approximate SVM known as LASVM, Online Random Forest (ORF), Incremental Learning Vector Quantisation (ILVQ),

an incremental learning algorithm for supervised neural networks proposed by Fuangkhon and Tanprasert (2009) known as Learn++, as well as adaptations of Naive Bayes and SGD for online algorithms. The authors illustrate the strengths and weaknesses of each method through simulated examples.

However, in the context of our research on online algorithms for GLMs, the existing methods for MLE still have deficiencies in terms of computational redundancy or inadequate retention of statistical information during updates.

2.4 High-dimensional classification

Classification is one of the trending topics in machine learning being a class of regression problems with discrete predicted variables (Kotsiantis et al., 2006). According to the different types of training data, the classification algorithms are categorised as unsupervised, semi-supervised, and supervised. In our study, we discuss supervised learning for classification, where the training data set consists of both the input and output variables. For simplicity, we use "classification" for "supervised classification" from here on.

In this section, we have a collection of N observations, represented as $\{(X_i, Y_i)\}$ where $i = 1, \ldots, N$. In this set, X_i represents the covariates, and Y_i denotes the corresponding response variable. Each of these covariates, X_i, is a p-dimensional vector. We primarily address a two-class classification problem, with $Y_i \in \{0, 1\}$. For new observations, the data point is denoted as x, and $x \in \mathbb{R}^p$.

It is often more practical to choose a classifier that performs well in specific conditions, rather than trying to find a single classifier that outperforms others in all classification problems. In recent years, numerous classification methods have been studied, each based on different assumptions and goals. While it is not feasible to list all of them here, we focus on discussing some of the best-known supervised techniques in this section. These include the Naive Bayes classifier, Fisher Linear Discriminant Analysis (also known as Linear Discriminant Analysis or LDA), the K-th Nearest-Neighbor (KNN) classifier, Support Vector Machines (SVM), and Logistic Regression (LR).

Rish et al. (2001) studies how the characters of data influence the Naive Bayes classifier. They find that Naive Bayes works the best when the features are completely

independent or when the features are functionally dependent. However, in most of the real data analysis, the assumption of independence is hard to meet and Naive Bayes has the worst performance between the two extremes.

LDA is a cornerstone in classification methodologies, renowned for its long history and widespread application. LDA operates by maximising the ratio of between-class variance to within-class variance in a dataset, aiming to achieve optimal class separation (Balakrishnama and Ganapathiraju, 1998). The method identifies a decision boundary that emphasises the difference between classes. Given the matrices for between-class scatter S_B and within-class scatter S_W, LDA determines the direction $\mathbf{w}$ optimising the ratio of the determinants of S_B and S_W. Bhattacharyya and Rahul (2013) delve into face recognition employing LDA. Specifically, for any new observation x, LDA classifies it according to the following rule:

$$C_{LDA}(x) = \begin{cases} 1 & \text{if } \mathbf{w}^T x > 0.5 \\ 0 & \text{otherwise.} \end{cases}$$

PCA is another classical classification method and the details of the history and applications of PCA can be found in Wold et al. (1987)'s work. Similar to LDA, PCA finds the linear combinations of the features and explains the datasets. The main difference between LDA and PCA is that LDA focuses more on data classification while PCA focuses more on feature classification. In detail, LDA keeps the location of the original data sets while PCA changes the shape and location of the data sets (Balakrishnama and Ganapathiraju, 1998). As a result, the data set analysed by PCA is less interpretable than the one from LDA.

KNN is a distance-based classification method, which is one of the most widely used methods due to its simplicity and effectiveness. The central assumption of KNN is that the observations from the same class are close to each other. The parameter K in KNN is the number of the neighbouring data to be checked. Based on pre-given distance measurement, KNN classifies a new observation into the class which has the most members among the K tested observations close to the new data. In detail, given x in $\mathbb{R}^p$, reorder the training data pairs to produce a sequence $(X_1, Y_1), \cdots, (X_N, Y_N)$

based on the increasing distance from x, such that:

$$\|X_1 - x\| \leqslant ... \leqslant \|X_N - x\|.$$

If there are any ties in distance, they are resolved by maintaining the original sequence of the indices. For any k in the range $\{1, \cdots, n\}$, the KNN classifier, denoted as $C_{KNN}(x)$ is defined as:

$$C_{KNN}(x) = \begin{cases} 1 & \text{if } \frac{1}{k}\sum_{i=1}^{k} 1\{Y_i = 1\} > 0.5 \\ 0 & \text{otherwise.} \end{cases}$$

One advantage of KNN is its universal consistency. The asymptotic misclassification rate of KNN approaches the optimal Bayes misclassification rate when $K/N \to 0$, where K is the number of nearest neighbours and N is the size of the training dataset (Stone, 1977). However, KNN also has several limitations and drawbacks. These include sensitivity to the choice of K (Hassanat et al., 2014); the choice of distance measurements, which has been studied by Abu Alfeilat et al. (2019); unstable performance with outliers (Parvin et al., 2010); high computational demands, especially for large datasets due to the necessity of calculating distances between new observations and each training data point.

Since Cortes and Vapnik (1995) introduced SVM, it has become one of the most extensively used and discussed methods in machine learning. SVM can be used for both classification and regression problems by maximising the margin between the training data and the class boundary. Referring to Cannings et al. (2020), SVM typically denotes classifiers represented as:

$$C_{SVM}(x) = \begin{cases} 1 & \text{if } \hat{f}(x) \geqslant 0 \\ 0 & \text{otherwise,} \end{cases}$$

where $x \in \mathbb{R}^p$ and $\hat{f}$ is the decision function and determined by:

$$\hat{f} \in \arg\min_{f \in H} \left[\frac{1}{N} \sum_{i=1}^{N} L(Y_i, f(X_i)) + \Omega(\lambda, \|f\|_H) \right],$$

here $L : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ is a loss function, $\Omega : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ is a regularisation function with a tuning parameter $\lambda > 0$, and H is a reproducing kernel Hilbert space with norm $\|\cdot\|_H$.

There are four main reasons why SVM is so popular, which are the strong theoretical foundation of SVM's algorithm, the good performance of dealing with normal-sized data sets, the flexibility to use, and the accuracy it achieves (Pavlidis et al., 2004). Cervantes et al. (2020) give a comprehensive review of SVM with its applications, challenges, and trends. Yet, according to Cervantes et al. (2020)'s research, SVM has limitations in terms of parameter selection, algorithm complexity, multi-class datasets, and imbalanced data sets. Furthermore, SVM has a considerable problem of analysing the data sets of very large sizes. Specifically, training time grows as the size of the dataset increases, and training and usage become unfeasible due to its computational power limitations.

There are several implementations of SVMs listed in Cervantes et al. (2020)'s work. These methods employ the ideas of data reduction, decomposition, sequential minimal optimisation, shrinking, and working selections to improve the traditional SVMs' performance, and reduce the computational complexity of the traditional SVM classifier for large-scale data sets analysis. However, traditional SVMs cannot study non-linear or high-dimensional data, so kernel SVMs have been proposed. The most frequently used kernel SVMs are linear, polynomial, radial, and tangential SVMs (Salazar et al., 2012).

LR, also known as the logistic model or logit model, is a widely utilised method for classification problems. LR adopts a maximum likelihood approach, leveraging the log-odds as its link function. It facilitates the analysis of relationships between discrete dependent variables and independent variables, which can be either continuous or categorical. Specifically, the model assumes:

$$\text{logit}\{\pi(X_i)\} = X_i^T \boldsymbol{\beta}, \text{ for } i = 1, \cdots, N,$$

where $\boldsymbol{\beta}$ represents a p-dimensional parameter. For a observed data x, if the condition

$$\pi(x) = \frac{\exp(x^T \boldsymbol{\beta})}{1 + \exp(x^T \boldsymbol{\beta})} \geqslant \frac{1}{2}, \tag{2.12}$$

is satisfied, it is classified into the group labelled 1. Thus, the LR classifier can be defined as:

$$C_{LR}(x) = \begin{cases} 1 & \text{if } (2.12) \text{ holds} \\ 0 & \text{otherwise.} \end{cases} \tag{2.13}$$

Due to the nature of LR, it is possible to study two or more classes of classification problems (McCullagh, 1983). There are a large number of references that discuss LR and compare it with other methods:

- Morgan and Teachman (1988) show the reasons why LR is superior to OLS regression. Tu (1996) compare LR with Neural Networks (NN) in predictions of medical outcomes. They conclude that although NN may be useful when the goal is for prediction outcomes and important interactions or when the data sets have complex nonlinearities, LR can look for possible causal relationships between the independent and dependent variables and the effects of the predictor variables on the outcome can be easier to understand.

- Ng and Jordan (2001) discuss the performance of LR and Naive Bayes by doing the repeated experiments, and found that Naive Bayes converged faster than LR, but the asymptotic error of the Naive Bayes was higher than LR. With the increase of the training data size, the performance of LR is closer to Naive Bayes (Ng and Jordan, 2001).

- Pohar et al. (2004) design simulated examples to investigate and compare the performance of LR and LDA. When the assumption of the normality of the covariates is satisfied, LDA works well and performs better than LR on small-size datasets, although the difference in performance between the two classifiers can be ignored on datasets of size 50 or larger (Pohar et al., 2004). One must note that when this assumption is not met, LDA clearly does not perform as well as LR, which in this case is better suited to analysing a larger number of training data sets and is independent of the distribution.

- Salazar et al. (2012) compare LR with the most used kernel SVMs, which are linear, polynomial, radial, and tangential SVMs. They discussed the two-class problem and generated data sets distributed from the Normal, Poisson, and Exponential distributions. It shows that LR performed better than all SVMs

when the sample sizes of the two groups are not the same. Besides, Polynomial SVM has the highest misclassification rate in Normal and Poisson cases.

- Tsangaratos and Ilia (2016) compare the performance of LR and Naive Bayes classifiers; Kalantar et al. (2018) compare the performance of LR, SVM, and NN classifiers; Nhu et al. (2020) study LR, Logistic Model Tree, Naive Bayes, NN, and SVM classifiers with landslide susceptibility mapping problems.

- Liu (2018) studies LR with real data analysis on breast cancer diagnosis rates, which shows good performance.

The aforementioned classifiers are widely discussed and applied in the analysis of high-dimensional data. High-dimensional classification problems are commonly encountered in various fields such as finance, medicine, genomics, healthcare, image categorisation, and more. The challenges posed by high-dimensional data, including the curse of dimensions and the need for effective feature selection, make these classifiers particularly relevant in addressing these problems. Extensive research has been conducted to adapt and optimise these classifiers for high-dimensional settings, leading to valuable insights and advancements in various domains.

Before delving into the solutions and advantages of analysing high-dimensional data, it is crucial to clarify certain assumptions that guide us in selecting appropriate methods from the plethora of available approaches. Dobriban and Wager (2018) outline three key hypotheses: the sparsity hypothesis, neighborhood-based methods under the manifold hypothesis, and independent features. In our study, we primarily focus on the sparsity hypothesis and independent learning assumption, which assumes that the features are independent. These assumptions play a significant role in guiding the choice of methods and algorithms for high-dimensional data analysis, allowing us to leverage the unique characteristics of the data and tailor our approaches accordingly.

Donoho et al. (2000) review the curses and blessings of high dimensional data and explain the theoretical benefits and aspects of issues because of the increase of dimensions. Bickel and Levina (2004) study the Independence Rule (IR, which is also known as the Naive Bayes classifier) for high-dimensional classification problems, and compared the performance of IR with LDA. When dealing with high-dimensional

data, LDA affected by diverging spectra performs much worse than IR (Bickel and Levina, 2004).

The performance of PCA, the most well-known dimension reduction method, and Random Projection (RP) are discussed and compared by Fan et al. (2014). Compared to RP, PCA is computationally more complex and not feasible for large data sets. Additionally, it shows that with the increase of the dimension, RPs have more advantages over PCA. While RP is competitive in terms of computational efficiency, the new space it generates does not take into account the intrinsic structure of the collected data set and the new data set is less interpretable. A number of methods have been proposed to improve the high distortion of RP, but some problems remain (Xie et al., 2017).

According to Fan and Fan (2008), an important finding suggests that even with the assumption of feature independence, classifiers can perform poorly when all features are used. This poor performance is attributed to the accumulation of noise in estimating population centroids within high-dimensional feature spaces. The study reveals that both the Independence Rule (IR), which utilises all features assuming independence, and LDA perform as poorly as random guessing due to noise accumulation. By employing the two-sample t-test, Fan and Fan (2008) identify the conditions under which the t-statistic can correctly identify all the important features with a probability of 1. However, it is important to note that the t-statistic is not always the optimal choice for feature selection, particularly in high-dimensional settings with increased noise (Fan and Fan, 2008).

Additionally, Fan et al. (2011) provide a comprehensive overview of the most commonly used classification methods, including distance-based and loss-based classifiers. They discuss the performance of these methods in the context of high-dimensional data, specifically LR, LDA, KNN, SVMs, and Naive Bayes classifiers (also known as IR). The authors provide detailed explanations for why classical classifiers struggle with high-dimensional data. Furthermore, Fan et al. (2014) review the challenges posed by big data in statistical and computational aspects. This challenge highlights the necessity for developing new approaches to overcome the limitations of traditional classification methods in effectively handling high-dimensional data.

For high-dimensional classification, Fan and Fan (2008) proposed the Features Annealed Independence Rules (FAIR) method, which effectively selects a subset of

important features and reduces the dimensions to solve the problem. This method takes advantage of both the ideas of IR and the t-statistic. IR avoids the frequent estimations of the large covariance matrix and the diverging condition number related to the covariance matrix, and the advantage of the t-statistic is that all significant features can be found under certain conditions. FAIR shows an oracle property with the classification error rate and Fan and Fan (2008) give the upper bound of the classification error of FAIR.

It is also worth mentioning that regularisation methods, as discussed in Section 2.1 and Section 2.2, are another promising approach for handling high-dimensional data in classification problems. Specifically, under the assumption of sparsity, penalised likelihood methods such as LR with regularisation can effectively perform feature selection and estimation simultaneously. The SIS and ISIS methods, introduced by Fan and Lv (2008) and further discussed and extended by Fan et al. (2009), Fan and Song (2010), are insightful and natural extensions of the two-sample t-statistics. In collaboration with SIS and ISIS, the performance of penalised likelihood methods in the analysis of high-dimensional data has been remarkably improved, resulting in efficient classification results. More details about SIS and ISIS are provided in Section 2.1 and Section 2.2.

2.5 Classification with imperfect labels

Another challenge of the classification problem, especially for large-scale data, is that it is difficult to obtain a dataset with all perfect labels, and noise in the labels is usually unavoidable. Many factors can lead to imperfect labels, such as inadequate information, misjudgments by researchers, and encoding mistakes to name but three (Cannings et al., 2020).

The random noise model is first proposed by Angluin and Laird (1988). They have proposed the random noise model and inspired a series of subsequent studies. There are several types of label noise that have been studied the most. The simplest setting is that each label is mislabelled independently with a fixed probability and that the noise is homogeneous. Class-dependent label noise, also known as label-dependent noise, refers to the fact that the probability of mislabelling is the same for labels from the same class. The label noise associated with features is feature-dependent, which

is heterogeneous. It is relatively common for label noise to depend on both class and features (Frénay et al., 2014, Cheng et al., 2020). In our study, we focus on the case where the noise is both class- and feature-dependent.

Cannings et al. (2020) investigate the classification problem with imperfect labels, considering noise in the data that is dependent on both the features and the class. They introduce a novel approach to evaluate the performance of any classifier by introducing a metric called "excess risk". This metric allows them to bound the excess risk of an arbitrary classifier trained with imperfect labels, based on its excess risk for predicting a noisy label. In detail, if a classifier $(\mathcal{C}_n)$ is consistent, it shows $R(\mathcal{C}_n) - R(\mathcal{C}^{\text{Bayes}}) \to 0$ as $n \to \infty$, where n is the number of the test data and $R(\mathcal{C})$ is the risk of the classifier.

$$R(\mathcal{C}) = P\{\mathcal{C}(X) \neq Y\}, \tag{2.14}$$

and Y is the true label of the test data.

Cannings et al. (2020) first identify a condition whereby classifiers are trained by imperfect labelled datasets that show consistency when classifying fully perfect labelled test data. Cannings et al. (2020) investigate the performance of widely-used classifiers—KNN, SVM, and LDA, as discussed in the preceding section—when applied to mislabelled datasets. They conclude that the convergence rates of excess risk of KNN and SVM are not affected by label noise, whereas LDA is not consistent in dealing with imperfect labels unless both classes had the same prior probability. Contrary to previous studies, Krause et al. (2016) found that label noise does not lead to adverse effects in the case of fine-grained classification, and Cannings et al. (2020) claim that they are the first to suggest that imperfect labels can help improve the performance of KNNs and SVMs under certain conditions. However, in Cannings et al. (2020)'s work, they do not consider any modifications to the corrupted data set, but simply ignore the imperfect labels and used the raw data directly as training data, though this is not appropriate for some classifiers, such as LR, as we discuss in our study. Furthermore, their work has not included high-dimensional data sets.

Sukhbaatar et al. (2014) and Goldberger and Ben-Reuven (2016) train the deep neural networks with the data sets having the label noise. They consider an additional noise channel, also known as a Noisy Layer, and estimate a noise parameter from the unknown noise distribution.

Bootkrajang and Kabán (2012) study the robust Logistic Regression (rLR), which takes into account estimates of the noise distribution from the raw data collected. After the introduction of rLR, more models have been proposed. Robust sparse logistic regression consisting of LR and L_1 penalty function is introduced by Bootkrajang and Kabán (2013) and it can be seen as a label-noise robust extension of the Bayesian logistic regression classifier. Robust sparse logistic regression can be used in the more general case where the dataset can be high-dimensional with labelled noise. The kernel rLR is investigated by Bootkrajang and Kabán (2014) without trusted verification sets and with noisy labels. Bootkrajang and Chaijaruwanich (2020) use Gaussian Mixture Model (GMM) to estimate the mislabelling probabilities and proposed a model named Gaussian Mixture Model-based Robust Logistic Regression (GMMLR). GMMLR performs better than rLR when the noise distribution is not uniform and GMMLR is more flexible for the mixture noise of Gaussian and gamma noise than rLR.

Although Bootkrajang and Kabán (2013) demonstrate the good performance of rLR and its extensions to study noisy labels, and mention that robust sparse logistic regression has an attractive side feature for detecting imperfect labels, there is little discussion of the statistical properties and gaps in the theoretical contributions. Frénay et al. (2014) point out that in theory, many existing learning algorithms are rarely fully robust to noise labels, except for a few simple cases. Also, all of the work mentioned above uses raw, noisy data to train the model without relabelling any imperfect labels. We are concerned about how imperfect labels can provide reasonable information or accuracy of posterior probabilities when training models with very noisy data sets, both in practice and in theory. An iterative algorithm to correct the imperfect labels is introduced with the feature-dependent label noise (Zhang et al., 2021). Nevertheless, when considering the computational cost, it becomes evident that the process of checking and correcting each label in this iterative algorithm is both expensive and impractical, especially when working with large datasets or studying high-dimensional data.

Obtaining a high-quality dataset through the conventional data cleansing process involves the identification of incorrect labels, the detection of errors, and their subsequent correction. Ridzuan and Zainon (2019) demonstrate this process in five phases, including data analysis, the definition of a transformation workflow and mapping

rules, verification, transformation, and the backflow of cleaned data. However, they also mention that these traditional data cleansing processes can be time-consuming and are not suitable for situations that require quick analysis results. In their review, Ridzuan and Zainon (2019) critically evaluate the traditional data cleansing approaches and provide detailed reasons why they are not well-suited for big data. Additionally, they discuss existing data cleansing methods specifically designed for big data, namely SCalable Automatic REpairing (SCARE), KATARA Cleanix, and BigDansing. They compare the features, execution methods, and mechanisms of each method, outlining their respective advantages and shortcomings. Importantly, Ridzuan and Zainon (2019) conclude that there are still gaps in current research on data cleansing techniques, particularly when it comes to addressing the challenges posed by big data.

Besides this, one must consider the traditional ways to view noisy labels as statistical outliers. However, for most practical problems, noisy labels rarely behave like independent random outliers, but rather exhibit the same multi-modal properties as perfect labels. Detecting imperfect labels is therefore a challenge. More importantly, if they are removed, some useful information is lost (Li et al., 2017).

Most of the existing work discusses data cleaning, data correction of entire datasets, and training models from raw data with unknown imperfect labels. As we have noted, the methods can be computationally expensive, less practical to apply, or produce less credible results due to unknown information from noisy data. The concept of "distillation" is later introduced by Hinton et al. (2015) without the need to detect all the data or delete the noisy labels. The distillation method first studies a clean data set and uses the predictions to train a corrupted data set. Inspired by Hinton et al. (2015)'s work, Li et al. (2017) propose a unified distillation framework that has a different way to leverage clean and noisy labels from Hinton et al. (2015)'s. The model is trained from a small subset of clean data and then follows a large noisy data set, suggesting that even with imperfect labels, the performance of the classifier is improved with the second training step.

Li et al. (2017) compare the distillation method with the previous works such as the label smoothing and bootstrapping method to reduce the impact of the effects of the noisy labels. They also compare the distillation method with the Finetuning, Importance Reweighting, and Noisy Layer which is often used in deep learning liter-

ature. All the results show the distillation method outperforms other methods. As Li et al. (2017)'s work only discuss the general case where there are no specific assumptions about the noise distribution, there are still gaps in the theoretical contribution but the concept of distillation is very illuminating.

3 Classification with mislabelled data

This chapter focuses on the classification problem in datasets that are corrupted by class- and feature-dependent noise. We use logistic regression (LR) for our study on classification. The arrangement of this chapter is delineated as follows.

Assuming data sparsity and initially focusing on low-dimensional datasets, Section 3.1 offers a detailed exposition of our proposed two-step estimation method, which capitalises on resampling. In Section 3.2, we delineate the metrics used to assess the performance of the methodologies explored in the simulation study of this chapter. Section 3.3 provides an in-depth account of how mislabelled datasets are generated for the simulation study undertaken in this chapter. Through simulated examples, we scrutinise the efficacy of our approach across a spectrum of mislabelled datasets.

The cross-validation process for determining a tuning parameter of the penalty function is expounded in Section 3.4. This section focuses on the examination of validation data, which consists of diverse types of observations. Section 3.5 explores different estimation orders on the three unknown parameters of our method. The simulation study conducted in both Section 3.4 and Section 3.5 demonstrates the consistent performance of our method across different scenarios. Through a simulation study, we compare the effectiveness of our proposed method with the methods discussed in Section 3.6 assuming "oracle information", where prior knowledge of the parameters associated with flipping probabilities is known.

In Section 3.7, we further explain alternative methods that utilise perfectly labelled datasets, ignore incorrect labels, or selectively correct specific labels while disregarding noise during the estimation process. We conduct various simulated examples to compare the performance of these methods with our proposed approach. Importantly, our proposed method exhibits competitive performance compared to the method that analyses all corrected data, and it noticeably outperforms the method that neglects imperfect labels and uses corrupted data directly. Furthermore, the results show that the LR classifier performs no better than random guessing when trained on raw data directly.

Section 3.8 outlines the strategy utilised to mitigate class- and feature-dependent label distortion in our model to being solely class-dependent, by approximating

the flipping probabilities based on the ratios of mislabels in resampled datasets. A simulation study is implemented to contrast the effectiveness of the stated method against our proposed approach, and our method demonstrates competitive outcomes.

Following the discussion on low-dimensional sparse data with mislabelling, we present the two-step estimation methods designed for high-dimensional mislabelled data. In Sections 3.9 and 3.10, we elucidate our integration of the Sure Independence Screening (SIS) method, as proposed by Fan and Lv (2008), Fan and Song (2010). We term this the Independence Screening (IS) method. Additionally, we incorporate the Iterative Sure Independence Screening (ISIS) method, put forth by Fan et al. (2009), which we refer to as the Iterative Independence Screening (IIS) approach. Additionally, we conduct a simulation study to demonstrate the performance of our proposed methods and compare them to the estimation process that uses all corrected data. Our method, which incorporates the IIS method, exhibits superior performance compared to the method that uses the IS approach. Furthermore, our method demonstrates more consistent performance across various scenarios.

Finally, in Section 3.11, we present a study on a real dataset concerning coronary heart disease prediction. The original dataset consisted of all perfect labels, but we introduce class-dependent noise by manually mislabelling it. We assess the performance of our method in comparison to alternative approaches, and our method consistently demonstrates competitive and robust performance.

3.1 Methodology

In this section, we provide a detailed explanation of the algorithm for our proposed two-step estimation method for low-dimensional sparse mislabelled data.

Suppose we have a sample consisting of i.i.d. observations as (X_i, y_i), $i = 1, \cdots, n$, where $X_i = (x_{1i}, \cdots, x_{pi})^T$ is a p-vector representing the predictor variables, with $x_{1i} \equiv 1$, and $y_i = \{0, 1\}$ is a binary response variable for the two-class classification. In the dataset, some y_is are incorrectly labelled and the exact information about the incorrect labels of the dataset is unknown at the time of data collection. We denote the unknown correct label of y_i as z_i, where $i = 1, \cdots, n$.

3.1.1 The proposed classifier

After collecting the corrupted dataset, we randomly resampled a subset of size m from $(X_i,\ y_i)$, $i = 1,\ \cdots,\ n$, and denote the subscripts of the subsample by $\mathcal{D} = \{i_1,\ \cdots,\ i_m\}$. $\mathcal{D}$ is a random set. The size of a subsample is smaller than the size of the dataset and even $m \ll n$. We have the assumption that the correct label of the original data can be known through further study of the subsample. In the following, the arguments are conditional on that $\mathcal{D}$ is given and to avoid replication, we state the arguments without saying "conditional on that $\mathcal{D}$ is given".

After the resampling, we combine the subsample with the original sample to obtain a new sample $(X_i,\ z_i,\ y_i)$, $i = 1,\ \cdots,\ n$, $z_i \in \{0,\ 1\}$, $y_i\ \in\ \{0,\ 1\}$. X_i and y_i are the original observations, z_i is the perfect label of y_i and can only be observed when $i \in \mathcal{D}$. Let

$$\mathcal{D}^c = \{1,\ \cdots,\ n\} - \mathcal{D}, \quad \pi_z(X_i) = P(z_i = 1|X_i), \quad \pi_y(X_i) = P(y_i = 1|X_i).$$

We define the probabilities of a label being mislabelled, also referred to as the "flipping probabilities", which are denoted as $\pi_{y|0}(X_i)$ and $1 - \pi_{y|1}(X_i)$, where

$$\pi_{y|0}(X_i) = P(y_i = 1|X_i, z_i = 0), \quad \pi_{y|1}(X_i) = P(y_i = 1|X_i, z_i = 1).$$

We assume

$$\text{logit}\{\pi_z(X_i)\} = X_i^T\boldsymbol{\beta}, \quad \text{logit}\{\pi_{y|0}(X_i)\} = X_i^T\boldsymbol{\eta_0}, \quad \text{logit}\{\pi_{y|1}(X_i)\} = X_i^T\boldsymbol{\eta_1},$$

here $\boldsymbol{\beta}$, $\boldsymbol{\eta_0}$, and $\boldsymbol{\eta_1}$ are all p-dimensional unknown parameters that need to be estimated. In particular, $\boldsymbol{\beta}$ denotes the regression coefficients that are of primary interest, while $\boldsymbol{\eta_0}$ and $\boldsymbol{\eta_1}$ are related to the flipping probabilities for a label.

It is easy to see

$$\pi_y(X_i) = E\{P(y_i = 1|X_i, z_i)|X_i\} = \pi_{y|1}(X_i)\pi_z(X_i) + \pi_{y|0}(X_i)(1 - \pi_z(X_i))$$

$$= \pi_{y|0}(X_i) + \left\{\pi_{y|1}(X_i) - \pi_{y|0}(X_i)\right\}\pi_z(X_i).$$

Therefore, we have the following likelihood function of $\boldsymbol{\beta}$, $\boldsymbol{\eta_0}$ and $\boldsymbol{\eta_1}$:

$$\begin{aligned}
&\left\{\prod_{i\in\mathcal{D}^c}\pi_y(X_i)^{y_i}(1-\pi_y(X_i))^{1-y_i}\right\}\times\\
&\prod_{i\in\mathcal{D}}\Big[\left\{\pi_{y|0}(X_i)^{y_i}(1-\pi_{y|0}(X_i))^{1-y_i}(1-\pi_z(X_i))^{1-z_i}\right\}^{1-z_i}\\
&\qquad\times\left\{\pi_{y|1}(X_i)^{y_i}(1-\pi_{y|1}(X_i))^{1-y_i}\pi_z(X_i)^{z_i}\right\}^{z_i}\Big],
\end{aligned}$$

and the penalised log-likelihood function is

$$\begin{aligned}
&L(\boldsymbol{\beta},\ \boldsymbol{\eta_0},\ \boldsymbol{\eta_1})\\
=\ &\sum_{i\in\mathcal{D}^c}\Big(y_i\text{logit}\left[\pi_{y|0}(X_i)+\left\{\pi_{y|1}(X_i)-\pi_{y|0}(X_i)\right\}\pi_z(X_i)\right]+\\
&\qquad\log\left[1-\pi_{y|0}(X_i)-\left\{\pi_{y|1}(X_i)-\pi_{y|0}(X_i)\right\}\pi_z(X_i)\right]\Big)\\
&+\sum_{i\in\mathcal{D}}(1-z_i)\left\{y_i(X_i^T\boldsymbol{\eta_0})+\log(1-\pi_{y|0}(X_i))+(1-z_i)\log(1-\pi_z(X_i))\right\}\\
&+\sum_{i\in\mathcal{D}}z_i\left\{y_i(X_i^T\boldsymbol{\eta_1})+\log(1-\pi_{y|1}(X_i))+z_i\log(\pi_z(X_i))\right\}\\
&-\mathbf{P}_{\lambda_\beta}(\|\boldsymbol{\beta}\|_1)-\mathbf{P}_{\lambda_{\eta_0}}(\|\boldsymbol{\eta_0}\|_1)-\mathbf{P}_{\lambda_{\eta_1}}(\|\boldsymbol{\eta_1}\|_1),
\end{aligned}\tag{3.1}$$

where $\mathbf{P}_\lambda(\cdot)$ is a penalty function with tuning parameter λ, $\|\boldsymbol{a}\|_1=\sum_{j=1}^{p}|a_j|$ for any p dimensional vector $\boldsymbol{a}$. Our goal is the find the MLEs of $\boldsymbol{\beta}$, $\boldsymbol{\eta_0}$ and $\boldsymbol{\eta_1}$ respectively. The MLEs are maximisers of the log-likelihood function (3.1), which we denote as $(\hat{\boldsymbol{\beta}},\ \hat{\boldsymbol{\eta}}_0,\ \hat{\boldsymbol{\eta}}_1)$.

For a new observation X, if

$$\frac{\exp(X^T\hat{\boldsymbol{\beta}})}{1+\exp(X^T\hat{\boldsymbol{\beta}})}\geqslant\frac{1}{2},\tag{3.2}$$

and we classify it into the group with $z=1$. Namely, the proposed classifier is denoted as $C_{\hat{\beta}}$ and we have

$$C_{\hat{\beta}}(X)=\begin{cases}1 & \text{if (3.2) holds}\\ 0 & \text{otherwise.}\end{cases}\tag{3.3}$$

3.1.2 Computational algorithm

In this section, we illustrate the algorithm of our method, which takes into account resampling and the estimation of parameters related to the flipping probabilities. The estimate is denoted as $\hat{\boldsymbol{\beta}}$ and the classifier denoted as $C_{\hat{\boldsymbol{\beta}}}$.

In general, we utilise the Newton-Raphson method to ascertain the MLEs of the unknown parameters presented in equation (3.1). Given the inherent sensitivity of the Newton-Raphson method to initial guesses—a factor that can affect both convergence and estimation accuracy (Casella and Bachmann, 2021)—we first analyse the resampled data. This preliminary analysis aims to yield more precise estimates, serving as robust initiators for subsequent estimations, where both resampled and unresampled data are collectively analysed.

The biggest obstacle to the implementation of the proposed method is the maximisation of $L(\boldsymbol{\beta},\ \boldsymbol{\eta_0},\ \boldsymbol{\eta_1})$, namely (3.1), because there is no closed form for the maximiser of (3.1). In detail, given the complexity of (3.1), it would be very difficult to apply the Newton-Raphson method to maximise the penalised log-likelihood function directly. However, we introduce an iterative algorithm that overcomes this difficulty and can be easily conducted. The intricacies of the two-step estimation computation are elaborated upon as follows:

Step 1: Estimation using the resampled data

In the first step, we maximise each of the following log-likelihood functions in (3.4) with the data from the subsample. Let $\mathcal{D}_{z_0}$ be the set of i such that $i \in \mathcal{D}$ and $z_i = 0$, and $\mathcal{D}_{z_1} = \mathcal{D} - \mathcal{D}_{z_0}$. We have $|\mathcal{D}_{z_0}| = m_0$ and $|\mathcal{D}_{z_1}| = m_1$. We can easily obtain the estimates of the parameters by applying penalised Iteratively Re-weighted Least Squares estimates to the subsamples $\{(X_i,\ z_i) :\ i \in \mathcal{D}\}$, $\{(X_i,\ y_i) :\ i \in \mathcal{D}_{z_0}\}$, and $\{(X_i,\ y_i) :\ i \in \mathcal{D}_{z_1}\}$ respectively. We denote the maximisers of $L(\boldsymbol{\beta})$, $L(\boldsymbol{\eta_0})$ and $L(\boldsymbol{\eta_1})$ shown in (3.4) as $\boldsymbol{\beta}^{(0)}$, $\boldsymbol{\eta}_{\mathbf{0}}^{(0)}$ and $\boldsymbol{\eta}_{\mathbf{1}}^{(0)}$ and they are taken as initial estimates for the second-step estimation. It is noteworthy that the LAQ method (Fan and Li, 2001) is utilised in the estimation process.

$$L(\boldsymbol{\beta}) = \sum_{i \in \mathcal{D}} \Big[z_i \text{logit}(\pi_z(X_i)) + \log(1 - \pi_z(X_i)) \Big] - \mathbf{P}_{\lambda_{\boldsymbol{\beta}}}(\|\boldsymbol{\beta}\|_1),$$

$$L(\boldsymbol{\eta_0}) = \sum_{i \in D_{z_0}} \Big[y_i \text{logit}(\pi_{y|0}(X_i)) + \log(1 - \pi_{y|0}(X_i)) \Big] - \mathbf{P}_{\lambda_{\boldsymbol{\eta_0}}}(\|\boldsymbol{\eta_0}\|_1),$$

$$L(\boldsymbol{\eta_1}) = \sum_{i \in D_{z_1}} \left[y_i \text{logit}(\pi_{y|1}(X_i)) + \log(1 - \pi_{y|1}(X_i)) \right] - \mathbf{P}_{\lambda_{\eta_1}}(\|\boldsymbol{\eta_1}\|_1). \quad (3.4)$$

Step 2: Estimations using the combined data

After we obtain the initial estimates from the first-step estimation, we estimate the model using the combined dataset, which consists of the resampled data with subscripts in $\mathcal{D}$ and the data not being scrutinised with subscripts in $\mathcal{D}^c$.

The proposed iterative algorithm starts with estimators of $\boldsymbol{\beta}^{(0)}$, $\boldsymbol{\eta}_0^{(0)}$ and $\boldsymbol{\eta}_1^{(0)}$. In the $k-$th, for $k = 1, \cdots$, iteration, we employ the LQA method to the penalty functions present in $L(\boldsymbol{\beta}, \boldsymbol{\eta_0}, \boldsymbol{\eta_1})$. The specifics of the iterative estimation are detailed below:

(1) we apply the Newton-Raphson method to maximise $L(\boldsymbol{\beta}^{(k-1)}, \boldsymbol{\eta_0}, \boldsymbol{\eta}_1^{(k-1)})$ with respect to $\boldsymbol{\eta_0}$, and denote the maximiser by $\boldsymbol{\eta}_0^{(k)}$. The tuning parameter $\lambda_{\boldsymbol{\eta}_0^{(k)}}$ is updated and retained for the next estimation step.

(2) we apply the Newton-Raphson method to maximise $L(\boldsymbol{\beta}^{(k-1)}, \boldsymbol{\eta}_0^{(k)}, \boldsymbol{\eta_1})$ with respect to $\boldsymbol{\eta_1}$, and denote the maximiser by $\boldsymbol{\eta}_1^{(k)}$. The tuning parameter $\lambda_{\boldsymbol{\eta}_1^{(k)}}$ is updated and retained for the next estimation step.

(3) we apply the Newton-Raphson method to maximise $L(\boldsymbol{\beta}, \boldsymbol{\eta}_0^{(k)}, \boldsymbol{\eta}_1^{(k)})$ with respect to $\boldsymbol{\beta}$, and denote the maximiser by $\boldsymbol{\beta}^{(k)}$. The tuning parameter $\lambda_{\boldsymbol{\beta}^{(k)}}$ is updated and retained for the next estimation step.

We continue the above iteration until convergence and take the converged $(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\eta}}_0, \hat{\boldsymbol{\eta}}_1)$ as the maximisers of $L(\boldsymbol{\beta}, \boldsymbol{\eta_0}, \boldsymbol{\eta_1})$. We provide a detailed derivation for obtaining the MLEs of the unknown parameters working with LQA for the two-step estimation. Further specifics can be found in Appendix A.

In the context of *Classification with mislabelled data*, we utilise the sandwich formula proposed by Fan and Li (2001) to obtain the estimated standard error (ESE) of the non-zero coefficients of the estimates $\hat{\boldsymbol{\beta}}$, $\hat{\boldsymbol{\eta}}_0$ and $\hat{\boldsymbol{\eta}}_1$. The sandwich formula, introduced by Fan and Li (2001), serves as an efficient method for accurately estimating the covariance of an estimate and is crucial for evaluating the significance of estimated coefficients and constructing confidence ellipsoids.

In the following contexts, our main objective is to estimate the standard error of $\hat{\boldsymbol{\beta}}$. Since the ESEs for $\hat{\boldsymbol{\eta}}_0$ and $\hat{\boldsymbol{\eta}}_1$ are the same as that for $\hat{\boldsymbol{\beta}}$, we do not provide additional

details in this regard. More specifically, equation (3.5) illustrates the estimation of the covariance matrix for d non-zero coefficients in $\boldsymbol{\beta}$, where $1 \leqslant d < p$.

$$\widehat{\text{cov}}(\hat{\boldsymbol{\beta}}_1) = \left\{\nabla^2 \ell(\hat{\boldsymbol{\beta}}_1) - n\Sigma_{\lambda_\beta}(\hat{\boldsymbol{\beta}}_1)\right\}^{-1} \widehat{\text{cov}}(\ell(\hat{\boldsymbol{\beta}}_1)) \left\{\nabla^2 \ell(\hat{\boldsymbol{\beta}}_1) - n\Sigma_{\lambda_\beta}(\hat{\boldsymbol{\beta}}_1)\right\}^{-1}, \tag{3.5}$$

where $\hat{\boldsymbol{\beta}}_1$ is a d-dimensional vector that consists of non-zero coefficients and represents the estimate for non-zero coefficients of $\boldsymbol{\beta}$. We have

$$\Sigma_{\lambda_\beta}(\hat{\boldsymbol{\beta}}_1) = \text{diag}\{p'_{\lambda_\beta}(|(\hat{\beta}_1)_1|)/|(\hat{\beta}_1)_1|, \cdots, p'_{\lambda_\beta}(|(\hat{\beta}_1)_d|)/|(\hat{\beta}_1)_d|\},$$

where $p_{\lambda_\beta}(\cdot)$ is the penalty function, λ_β is a tuning parameter, and $\ell(\cdot)$ denotes an unpenalised log-likelihood function. ESE of $\hat{\beta}_j$, $j = 1, \cdots, d$, can be obtained from the diagonal elements of $\widehat{\text{cov}}(\hat{\boldsymbol{\beta}}_1)$.

3.2 Evaluation metrics for simulation study

In this chapter's simulation study, we evaluate the effectiveness of the proposed methodologies through numerous simulated experiments presented in the subsequent sections. In this section, we outline multiple metrics and criteria used to assess the accuracy of the estimation, variable selection, hypothesis testing capabilities, classification accuracy, and computational efficiency.

In our simulation study, we conduct 100 repetitions for each experiment. We implement each experiment using C++ and utilise the computing platform provided by the Viking computing cluster. Further details about the computing platform can be found on the website: Viking computing cluster.

We evaluate the estimates derived from different methods based on the criteria outlined below. In the subsequent formulas, we use $\boldsymbol{\beta}$, a p-dimensional unknown parameter, as a representative example and its estimate $\hat{\boldsymbol{\beta}}$ for illustration. It should be noted that any estimate of the unknown parameter can be substituted into these formulas to analyse the performance.

(1) For a single trial, we calculate the Mean Relative Squared Error (MRSE) of $\hat{\boldsymbol{\beta}}$ as follows:

$$\text{MRSE}(\hat{\boldsymbol{\beta}}) = \frac{1}{p} \times \frac{\|\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}\|_2^2}{\|\boldsymbol{\beta}\|_2^2},$$

where $\|\|_2$ represents the L_2 norm. The Averaged Mean Relative Squared Errors (AMRSE) and Median Mean Relative Squared Errors (MMRSE) are the average and median values of the MRSEs, respectively, computed over the 100 repetitions of a single experiment. AMRSE can measure the overall performance of the 100 estimates, while MMRSE, being less sensitive to outliers, provides a more stable and robust measure of performance. According to the definition of MRSE, the smaller the values of MRSE, AMRSE, and MMRSE, the closer the estimate is to $\boldsymbol{\beta}$.

(2) For the research topic of *Classification with mislabelled data*, we employ the sandwich formula (3.5) as detailed in the subsequent sections to obtain the Estimated Standard Errors (ESEs) of $\hat{\boldsymbol{\beta}}$. Additionally, we adhere to the strategy outlined in Fan and Li (2001) to assess the performance of the standard error formulas (3.5). Specifically, we compute the median of ESEs from 100 estimates and denote it as $\mathrm{ESE}_{\mathrm{m}}$. To measure the overall performance of the standard error formula, we divide $\mathrm{ESE}_{\mathrm{m}}$ by 0.6745 to obtain $\mathrm{ESE}_{\mathrm{o}}$. We also compute the median absolute deviation divided by 0.6745, denoted as SD, to compare with ESEs.

(3) For a given significance level of $\alpha = 0.05$, 95% confidence ellipsoids are constructed separately for the non-zero coefficients and the zero coefficients of $\boldsymbol{\beta}$. The coverage probabilities (CPs) are then computed and recorded for the non-zero and zero coefficients respectively.

(4) For the 100-repetition experiment, the averaged number of the incorrect zero estimates for non-zero coefficients of $\boldsymbol{\beta}$ and incorrect non-zero estimates for zero coefficients of $\boldsymbol{\beta}$ are calculated.

(5) As the binary classification problem is studied, the efficiency of the classifier is considered to be evaluated with the Bayes classifier. The risk of any classifier $C(X)$ is defined as

$$R(C) = P(C(X) \neq y),$$

where $y = \{0, 1\}$ is the correct label of the new observation X. For any classifier,

we have

$$C(X) = \begin{cases} 1 & \text{if} \quad P(y = 1|X) \geqslant 0.5, \\ 0 & \text{otherwise.} \end{cases} \tag{3.6}$$

It is widely acknowledged that the Bayes classifier, represented in our study as $C_B(X)$, possesses the minimum risk. We introduce a specific metric, termed "Excess Risk", to evaluate the performance of a classifier. It is defined as:

$$ER(C) = R(C_B)/R(C), \tag{3.7}$$

where $R(C_B)$ denotes the risk associated with the Bayes classifier, and $R(C)$ signifies the risk of any given classifier C. By this definition, a higher excess risk value, nearing 1, suggests that the classifier performs comparably to the Bayes classifier in terms of misclassification rate.

In our simulation study, by utilising the pre-established values of the model's unknown parameters, we formulate a classifier. This classifier, informed by the accurate parameter values, functions in a manner akin to a Bayes classifier, which is intrinsically designed to reduce classification risk. Specifically, with reference to equation (3.7), we generate 500 test data points from out-of-sample data. The misclassification rate of the classifiers is then benchmarked against the Bayes classifier constructed using the known parameter values.

(6) The computation time for each of the 100 replications are recorded for some simulation study discussed below and in the format of *hours:minutes:seconds*.

For regularisation techniques, we incorporate the SCAD penalty function for simulation studies, with the penalty term $\mathbf{P}_\lambda(\cdot)$ given by equation (2.4). In this equation, both a and λ serve as tuning parameters acting as thresholds that require determination. As recommended in Fan and Li (2001), we set $a = 3.7$. The process of determining a suitable value for λ is discussed in Section 3.4 for the topic of *Classification with mislabelled data*. Across an experiment with 100 independent repetitions, the settings for selecting tuning parameters are consistent. This encompasses the same predetermined search intervals, search step sizes, and sizes of validation data.

3.3 Mislabelled dataset generation and simulation study set-up

In this section, we explain how we generate the mislabelled dataset for the simulation study. For all the simulation examples in *Classification with mislabelled data*, we first generate a dataset with all perfect labels and then corrupt the dataset using the flipping probabilities.

To elaborate, a dataset with a sufficiently large sample size $(X_i,\ z_i)$, $i = 1,\ \cdots,\ N$, is first generated. $x_{1i} \equiv 1$, and the remaining covariates have a correlation structure as $x_{[2:p]i} \sim \mathcal{N}(0, \boldsymbol{V})$, where $\boldsymbol{V}$ is a $(p-1) \times (p-1)$ compound symmetry covariance matrix. $z_i = \{0, 1\}$ is considered a perfect label. We have the index set $\mathcal{S}_0$ for the data labelled as 0 and $\mathcal{S}_1$ for label 1, with $|\mathcal{S}_0| = N_0$, $|\mathcal{S}_1| = N_1$, and $N_0 + N_1 = N$.

Given the parameters $\boldsymbol{\eta_0}$ and $\boldsymbol{\eta_1}$ associated with the flipping probabilities, we corrupt the dataset and obtain the observed labels denoted as $y_i,\ i = 1,\ \cdots,\ N$, according to (3.8) and (3.9),

$$y_i = \begin{cases} 1 & \pi_{y|0}(X_i) > \epsilon_{0i} \\ 0 & \text{otherwise,} \end{cases} \qquad \text{for } i \in \mathcal{S}_0, \tag{3.8}$$

$$y_i = \begin{cases} 1 & \pi_{y|1}(X_i) > \epsilon_{1i} \\ 0 & \text{otherwise,} \end{cases} \qquad \text{for } i \in \mathcal{S}_1, \tag{3.9}$$

where ϵ_{0i} and ϵ_{1i} are independent varaibles, and $\epsilon_{0i} \overset{\text{i.i.d.}}{\sim} U(0,1)$, and $\epsilon_{1i} \overset{\text{i.i.d.}}{\sim} U(0,1)$, $i = 1, \cdots, N$.

We now obtain a dataset $\{(X_1, z_1, y_1), (X_2, z_2, y_2), \cdots, (X_N, z_N, y_N)\}$, where y_is can be imperfect, meaning $y_i \neq z_i$. From the generated dataset, we record the ratios of the number of imperfect labels in the dataset to the class size, which are

$$M_{y|0} = \sum_{i \in \mathcal{S}_0} \frac{\mathbf{1}_{y_i|0}}{N_0}, \tag{3.10}$$

$$M_{y|1} = \sum_{i \in \mathcal{S}_1} \frac{\mathbf{1}_{y_i|1}}{N_1}, \tag{3.11}$$

and $\mathbf{1}_{y_i|0}$ and $\mathbf{1}_{y_i|1}$ are given by

$$\mathbf{1}_{y_i|0} = \begin{cases} 1 & y_i = 1 \\ 0 & \text{otherwise} \end{cases} \quad \text{for } i \in \mathcal{S}_0, \quad \mathbf{1}_{y_i|1} = \begin{cases} 1 & y_i = 0 \\ 0 & \text{otherwise} \end{cases} \quad \text{for } i \in \mathcal{S}_1.$$

We also record the total mislabelling rate of the dataset, denoted as M and defined as

$$M = \frac{\sum_{i \in \mathcal{S}_0} \mathbf{1}_{y_{i|0}} + \sum_{i \in \mathcal{S}_1} \mathbf{1}_{y_{i|1}}}{N}. \tag{3.12}$$

For the analysis, z_i is unknown, and X_i and $y_i = \{0, 1\}$ are both observed.

For all simulation studies discussed in the sections on *Classification with mislabelled data*, we conduct 100 independent trails for each experiment, with $N = 15000$ data points generated for each run. We randomly select n samples (X_i, y_i), $i = 1, \cdots, n$, from each dataset, where $0 < n < N$, to form the full-size training data. For each trial, we also select 500 data points as the testing data, and their labels are correct. The testing dataset is independent of the training dataset. Referring to (3.10)-(3.12), the mean mislabelling rates of the 100 trials for each class and the total datasets with a size of $N = 15000$ are presented in some examples of the simulation study below and denoted as $\bar{M}_{y|0}$, $\bar{M}_{y|1}$, and $\bar{M}$, respectively.

In the following simulation study, we introduce SCAD into the model as the penalty term. As suggested by Fan and Li (2001), we fix the tuning parameter $a = 3.7$ in Equation (2.4). For another tuning parameter λ, we modify the classical Leave-P-Out Cross Validation method to find a desirable value for the tuning parameter λ. This is explained in detail in Section 3.4, and the validation dataset is set as $n_{cv} = 20$ unless otherwise specified.

3.3.1 Resampling for the mislabelled dataset

When we collect the dataset with mislabelling $(X_i,\ y_i)$, $i = 1,\ \cdots,\ n$, we resample them to correct the incorrect labels. In detail, we randomly select m, $1 < m < n$, data points and obtain a subsample $(X_i,\ z_i,\ y_i)$, $i \in \mathcal{D}$, where z_i is the correct label only known in this subset. We then have two index sets, $\mathcal{D}_{z_0}$ and $\mathcal{D}_{z_1}$, consisting of elements which are indices of correct labels $z_i = 0$ and $z_i = 1$, respectively, where

$i \in \mathcal{D}$, and $|\mathcal{D}_{z_0}| = m_0$ and $|\mathcal{D}_{z_1}| = m_1$.

In the resampled dataset, the ratios of incorrect labels to class size are

$$\hat{M}_{y|0} = \sum_{i \in \mathcal{D}_{z_0}} \frac{I_{y_i|0}}{m_0}, \tag{3.13}$$

$$\hat{M}_{y|1} = \sum_{i \in \mathcal{D}_{z_1}} \frac{I_{y_i|1}}{m_1}, \tag{3.14}$$

where

$$I_{y_i|0} = \begin{cases} 1 & y_i = 1 \\ 0 & \text{otherwise} \end{cases} \quad \text{for } i \in \mathcal{D}_{z_0}, \quad I_{y_i|1} = \begin{cases} 1 & y_i = 0 \\ 0 & \text{otherwise} \end{cases} \quad \text{for } i \in \mathcal{D}_{z_1}.$$

The mislabelling rate of the resampled dataset is

$$\hat{M} = \frac{\sum_{i \in \mathcal{D}_{z_0}} \mathbf{1}_{y_{i|0}} + \sum_{i \in \mathcal{D}_{z_1}} \mathbf{1}_{y_{i|1}}}{m}. \tag{3.15}$$

In our study, as explained, we conduct 100 independent trials for each experiment. To calculate the average mislabelling rates of the 100 resampled datasets where the true labels are only known, we use equations (3.13)-(3.15), and denote the resulting rates of the datasets as $\bar{\hat{M}}_{y|0}$, $\bar{\hat{M}}_{y|1}$, and $\bar{\hat{M}}$ respectively.

3.3.2 Simulation study

In this section, we design two simulation examples to demonstrate the performance of our proposed method. The low-dimensional mislabelled dataset is discussed with varying sample and subsample sizes and varying correlated covariates. The parameter settings for model (3.1) are the same for both examples in this section and we set $p = 15$. The details are as follows

$$\boldsymbol{\beta} = (0,\ \beta_2,\ \beta_3,\ 0,\ \cdots,\ 0)^T_{p\times 1}, \qquad \beta_2 = 2,\ \beta_3 = 1.5;$$

$$\boldsymbol{\eta_0} = (0,\ \eta_{0,2},\ \eta_{0,3},\ 0,\ \cdots,\ 0)^T_{p\times 1}, \qquad \eta_{0,2} = -0.65,\ \eta_{0,3} = 1.2;$$

$$\boldsymbol{\eta_1} = (0,\ \eta_{1,2},\ \eta_{1,3},\ 0,\ \cdots,\ 0)^T_{p\times 1}, \qquad \eta_{1,2} = 1.5,\ \eta_{1,3} = -1.$$

Example 1: mislabelled datasets with different sizes. In the first example, we investigate the impact of mislabels in datasets with varying proportions relative to the total training sample size, denoted as n, as well as the resample size, denoted as m. We have the covariance matrix of covariates $x_{[2:p]i}$, $i = 1, \cdots, N$, as $\mathbf{V}_{14\times14}$, which is a compound symmetry covariance matrix with the correlation parameter is $\rho = 0.2$.

Referring to Section 3.3, after generating $N = 15000$ i.i.d. data points, we select the training dataset with mislabelling and each of size n. We assess the performance of our proposed method by resampling each dataset using a subsample size of m, and the full-size of the training dataset n has a relationship to the resampled size as $\frac{n}{m}$ to $2, 3, 4, 5,$ and 6, respectively. Specifically, we set m to $300,\ 400,\ 500$. We estimate the parameters $\boldsymbol{\beta}$, $\boldsymbol{\eta}_0$, and $\boldsymbol{\eta}_1$ using our proposed method and record the resulting numerical performance in Tables 3.1-Table 3.8, which summarise the overall performance across the 100 repeated trials.

The AMRSEs and the MMRSEs of $\hat{\boldsymbol{\beta}}$s, $\hat{\boldsymbol{\eta}}_0$s, and $\hat{\boldsymbol{\eta}}_1$s are recorded in Table 3.1-3.3 respectively. We can observe consistent trends among the three estimates from the results presented in the tables. First, as expected, the performance of all estimates improves when the corrupted dataset maintains the same size while more data is resampled. Second, it is noticeable that when the resampled data has a fairly large size and remains fixed at $m = 500$, the performance of the estimates of $\boldsymbol{\beta}$, $\boldsymbol{\eta}_0$, and $\boldsymbol{\eta}_1$ improves with increases in the full training size, indicating a growing size of corrupted data. However, when the resampled size is $m = 300$ or 400, all estimates exhibit enhanced performance when the ratio of the size of the two datasets changes from 2 to 3, with a more significant improvement in the performance of the estimates, particularly for the estimates of parameters related to the flipping probabilities, $\hat{\boldsymbol{\eta}}_0$s and $\hat{\boldsymbol{\eta}}_1$s. Nonetheless, the accuracy of estimates increases within a certain range of the ratio of the sizes of the two datasets and remains at a certain level or decreases slightly when the ratio exceeds a certain value. This suggests that our method cannot be indefinitely improved with an increase in the size of the unresampled data, especially when the size of the resampled dataset is limited. This can be attributed to the lower accuracy of the first-step estimates when the resampled dataset is small, while the addition of the unresampled data introduces more mislabels that interfere with the estimates.

The comparison of the ESEs with respect to $\hat{\boldsymbol{\beta}}$s obtained by Equation (3.5) and

SDs as the true values are recorded in Table 3.4. As can be seen from the table, in all cases, the sandwich formula works well even with the increasing number of incorrect labels in the dataset. We omit the details of ESEs and SDs of $\hat{\boldsymbol{\eta}}_0$s and $\hat{\boldsymbol{\eta}}_1$s because of the similar results which validate that the sandwich formula performs well.

We also present the variable selection performance of our method for the three parameters in Table 3.5-3.7. The tables present the averaged number of incorrect zero estimates and non-zero estimates, and the CPs of the non-zero coefficients and zero coefficients for $\boldsymbol{\beta}$, $\boldsymbol{\eta}_0$, and $\boldsymbol{\eta}_1$, respectively. In detail, we can see from Table 3.5 that SCAD works very well, as there are no erroneous zero estimates of the non-zero coefficients of $\boldsymbol{\beta}$ in all cases. Not surprisingly, when studying the same corrupted dataset, there are fewer incorrect non-zero estimates of the zero coefficients of $\boldsymbol{\beta}$ when the size of the resampled dataset is increased. Meanwhile, the incorrect non-zero estimates for zero coefficients of $\boldsymbol{\beta}$ also decrease. It is also noticeable that, with the resampled dataset fixed, as n increases and more mislabels are involved, there are fewer incorrect non-zero estimates for zero coefficients of $\boldsymbol{\beta}$, and CPs for zero coefficients increase as well. While when the fixed resampled datasets are fairly large ($m = 500$), the CPs for non-zero coefficients of $\boldsymbol{\beta}$ show a trend of increasing, while for a smaller size ($m = 300$ or 400) of resampled datasets, CPs for non-zero coefficients increase first and then retain at certain levels.

The same trends as in Table 3.5 are also shown in Table 3.6 and Table 3.7 for the other two estimates. A noticeable phenomenon, slightly different from Table 3.5 in this example, is that with the fixed small resampled dataset ($m = 300$) and as the training datasets increase, there are more significant features of $\boldsymbol{\eta}_0$ missed, and the decrease of CPs during the 100 repetitions is more evident. This can be related to the small size of the data used for the first step estimation, whose estimates join in the second step estimation as initiators, affecting the final results.

As can be seen from Table 3.8, our classifier exhibits almost the same performance as the Bayes classifier, shown in the table with all excess risks being close to 1. Specifically, consistent with the findings shown in the tables above, when more resampled data are available in the same corrupted dataset or when more unresampled data join the training dataset with the resampled datasets remaining the same, the efficiency of our classifiers improves, with excess risks increasing and approaching 1.

Table 3.1: AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\beta}}$s

$\mathfrak{m}$ \ $\frac{n}{m}$	2	3	4	5	6
300	2.7410(1.8500)	2.4840(1.3787)	2.3640(1.4867)	2.3280(1.3894)	2.3670(1.4937)
400	1.6150(1.0561)	1.5070(1.0088)	1.4370(0.9797)	1.4880(0.9353)	1.4410(0.8139)
500	1.0690(0.8739)	0.9850(0.6711)	0.9300(0.4653)	0.8900(0.4473)	0.8480(0.4418)

* All values recorded in this table multiplied by 10^{-3} are true values.

Table 3.2: AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\eta}}_{\mathbf{0}}$s

$\mathfrak{m}$ \ $\frac{n}{m}$	2	3	4	5	6
300	10.449(8.8148)	8.466(5.1157)	8.645(3.9888)	8.721(2.4163)	9.464(2.2435)
400	6.975(4.9439)	4.551(2.3797)	3.656(1.4081)	3.829(0.8762)	3.618(0.7508)
500	5.635(3.4819)	3.930(1.7182)	3.049(1.0307)	2.960(0.7218)	2.837(0.4554)

* All values recorded in this table multiplied by 10^{-3} are true values.

Table 3.3: AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\eta}}_{\mathbf{1}}$s

$\mathfrak{m}$ \ $\frac{n}{m}$	2	3	4	5	6
300	7.2850(3.4344)	4.4220(2.2357)	3.5290(2.0771)	4.4560(1.5639)	4.781(1.0401)
400	3.9370(2.2296)	2.4990(1.7209)	1.941(1.2711)	2.468(0.7316)	2.507(0.7993)
500	3.7320(2.0085)	2.353(1.00280)	1.670(0.6747)	1.823(0.6461)	1.286(0.5258)

* All values recorded in this table multiplied by 10^{-3} are true values.

Table 3.4: SDs and ESEs for non-zero $\boldsymbol{\beta}_{15\times1}$ coefficient estimates

	$\mathfrak{m}$ \ $\frac{n}{m}$	2	3	4	5	6
	300	0.241(0.235,0.348)	0.224(0.225,0.303)	0.251(0.220,0.326)	0.285(0.214,0.318)	0.292(0.208,0.308)
β_2	400	0.212(0.204,0.303)	0.204(0.195,0.288)	0.218(0.191,0.283)	0.225(0.182,0.270)	0.221(0.176,0.261)
	500	0.171(0.180,0.267)	0.169(0.171,0.254)	0.189(0.165,0.245)	0.177(0.159,0.236)	0.193(0.153,0.227)
	300	0.153(0.197,0.292)	0.192(0.184,0.273)	0.166(0.178,0.264)	0.176(0.172,0.255)	0.211(0.165,0.245)
β_3	400	0.136(0.172,0.254)	0.146(0.161,0.238)	0.140(0.150,0.223)	0.159(0.144,0.214)	0.173(0.139,0.206)
	500	0.144(0.151,0.223)	0.133(0.142,0.211)	0.122(0.135,0.200)	0.120(0.128,0.190)	0.113(0.121,0.180)

* For each cell, the value outside the brackets represents Standard Deviation (SD), while the first and second numbers inside the brackets represent $\mathrm{ESE_m}$ and $\mathrm{ESE_o}$, respectively.

Table 3.5: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$

	$\frac{n}{m}$ / m	2	3	4	5	6
non0 coeff.	300	0(0.9300)	0(0.9350)	0(0.9300)	0(0.9050)	0(0.8950)
	400	0(0.9100)	0(0.9050)	0(0.9200)	0(0.8700)	0(0.9100)
	500	0(0.9300)	0(0.9300)	0(0.9300)	0(0.9250)	0(0.9200)
0 coeff.	300	2.13(0.8762)	1.38(0.9285)	1.12(0.9485)	0.94(0.9615)	0.87(0.9662)
	400	1.6(0.8908)	1.11(0.9331)	0.93(0.9469)	0.84(0.9585)	0.74(0.9700)
	500	1.5(0.8877)	0.88(0.9369)	0.55(0.9646)	0.44(0.9746)	0.39(0.9785)

* $\boldsymbol{\beta}_{15\times 1}$ cosists of 2 non-zero coefficients.

Table 3.6: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\eta_0}$

	$\frac{n}{m}$ / m	2	3	4	5	6
non0 coeff.	300	0.05(0.7950)	0.07(0.7750)	0.10(0.7250)	0.12(0.6800)	0.11(0.6650)
	400	0.03(0.8450)	0.03(0.8500)	0.03(0.8350)	0.05(0.8100)	0.04(0.8050)
	500	0.01(0.8750)	0.01(0.8350)	0.01(0.8450)	0.02(0.8250)	0.01(0.8400)
0 coeff.	300	2.49(0.8592)	1.46(0.9138)	1.07(0.9269)	0.84(0.9469)	0.63(0.9538)
	400	2.48(0.8631)	1.37(0.9231)	0.95(0.9385)	0.66(0.9569)	0.49(0.9662)
	500	2.22(0.8746)	1.15(0.9331)	0.82(0.9462)	0.58(0.9585)	0.49(0.9631)

* $(\boldsymbol{\eta_0})_{15\times 1}$ cosists of 2 non-zero coefficients.

Table 3.7: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\eta_1}$

	$\frac{n}{m}$ / m	2	3	4	5	6
non0 coeff.	300	0.01(0.9150)	0.01(0.8800)	0.01(0.8650)	0.04(0.8300)	0.03(0.8000)
	400	0(0.9000)	0(0.9050)	0(0.8950)	0(0.8550)	0(0.8650)
	500	0(0.8800)	0(0.8950)	0(0.8600)	0(0.8650)	0(0.8600)
0 coeff.	300	2.05(0.8792)	1.29(0.9292)	0.77(0.9608)	0.62(0.9662)	0.51(0.9685)
	400	1.82(0.9015)	1.00(0.9469)	0.73(0.9654)	0.51(0.9669)	0.38(0.9754)
	500	1.96(0.8992)	1.09(0.9415)	0.63(0.9677)	0.51(0.9677)	0.38(0.9777)

* $(\boldsymbol{\eta_1})_{15\times 1}$ cosists of 2 non-zero coefficients.

Table 3.8: Excess risks of $C_{\hat{\beta}}$s

m \ $\frac{n}{m}$	2	3	4	5	6
300	0.971382	0.974752	0.977862	0.982230	0.982993
400	0.985675	0.987407	0.988179	0.991764	0.991958
500	0.988759	0.992250	0.994299	0.995179	0.994592

* The misclassification rate of the Bayes Classifier is 20.23% across different cases.

Example 2: mislabelled datasets with different correlated covariates. The second example in this section discusses the performance of our proposed approach when dealing with different corrupted datasets that consist of various correlated covariates. Specifically, we have multiple correlation parameters $\rho = 0, 0.2, 0.5$ for the covariance matrix $\boldsymbol{V}_{14\times14}$, a compound symmetry covariance matrix. The covariates $x_{[2:p]i} \overset{\text{i.i.d.}}{\sim} N(0, \boldsymbol{V})$, $i = 1, \cdots, 15000$. We select $n = 1000$ data points to build the training dataset and test on different sizes of resampled datasets, which have $m = 300, 500, 900$.

The comparison of AMRSEs and MMRSEs of $\hat{\boldsymbol{\beta}}$s, $\hat{\boldsymbol{\eta}}_0$s, and $\hat{\boldsymbol{\eta}}_1$s are presented respectively in Table 3.9-3.11. Under the settings with the same size of resampled datasets, it can be observed that when the covariates are more correlated, the accuracy of all estimates decreases. When $\rho = 0.5$, the AMRSE, and MMRSE are the largest among all cases. When m increases, the gaps become smaller among cases with varying correlated covariates. For the case of $\rho = 0.5$, having a larger resampled dataset helps enhance the performance of the estimates, which is consistent with **Example 1**.

Table 3.12 records the ESEs from the sandwich formula (3.5) and SDs of $\hat{\boldsymbol{\beta}}$s as true values. Consistent with **Example 1** above, the results reveal once again that the sandwich formula (3.5) performs well in the analysis of mislabelled data. This can be seen by the fact that the ESEs do not deviate significantly from the true values across different cases with varying correlated covariates.

The performance of variable selection for the three parameters is recorded in Table 3.13-3.15. Variable selection performance can be affected by settings such as search intervals, search sizes, and validation datasets, among others. These details are discussed in the following section, Section 3.4. Under fair settings for finding the tuning parameter λs of the penalty function, these numerical results indicate that

SCAD performs well in estimation on mislabelled datasets, regardless of whether the correlations of the observed covariates are low or high. As the results show, there are no incorrect zero estimates for the non-zero coefficients of $\boldsymbol{\beta}$ in all cases and only a few incorrect zero estimates for the non-zero coefficients of $\boldsymbol{\eta_0}$ and $\boldsymbol{\eta_1}$ during the 100 repetitions when the resampled data is small, with $m = 300$. As the covariates become more correlated, CPs for the non-zero coefficients exhibit a decreasing trend, but not drastically. With the settings of dataset sizes, the incorrect number of zero coefficients and CPs for the zero coefficients of the three unknown parameters are close across different cases. In this case, as the covariates become more correlated, there are more erroneous non-zero estimates and CPs decrease.

As shown in Table 3.16, our proposed classifier exhibits performance close to that of the Bayes classifier and also maintains similar efficiency when trained with different correlated covariates. In this example, when analysing the same size resampled datasets and as the correlation parameter becomes larger, the excess risks slightly decrease, but not substantially.

Table 3.9: AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\beta}}$s

ρ \ m	300	500	900
0.00	2.3990(1.3730)	1.2890(1.0517)	0.6570(0.5243)
0.20	2.5600(1.6467)	1.3310(0.9801)	0.8550(0.7461)
0.50	3.3030(2.0069)	1.8310(1.2229)	0.8890(0.5558)

* All values recorded in this table multiplied by 10^{-3} are true values.

Table 3.10: AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\eta}}_{\mathbf{0}}$s

ρ \ m	300	500	900
0.00	7.682(3.9475)	4.856(2.9494)	3.927(2.2850)
0.20	9.700(5.4233)	5.637(3.5818)	5.678(5.0459)
0.50	13.058(10.0345)	11.137(7.2601)	7.042(5.1153)

* All values recorded in this table multiplied by 10^{-3} are true values.

Table 3.11: AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\eta}}_1$s

ρ \ m	300	500	900
0.00	5.3190(1.3847)	3.1550(1.8225)	2.553(1.7561)
0.20	5.1260(1.8916)	3.7530(1.8802)	3.4700(2.5394)
0.50	6.2660(2.2727)	4.5260(2.8331)	3.5280(1.9398)

* All values recorded in this table multiplied by 10^{-3} are true values.

Table 3.12: SDs and ESEs for non-zero $\boldsymbol{\beta}$ coefficient estimates

	ρ \ m	300	500	900
β_2	0	0.206(0.229,0.340)	0.241(0.184,0.273)	0.158(0.144,0.214)
	0.2	0.264(0.225,0.334)	0.162(0.181,0.268)	0.146(0.141,0.210)
	0.5	0.301(0.234,0.346)	0.195(0.186,0.275)	0.126(0.143,0.212)
β_3	0	0.203(0.188,0.279)	0.141(0.154,0.228)	0.113(0.123,0.182)
	0.2	0.200(0.185,0.274)	0.142(0.153,0.227)	0.123(0.122,0.180)
	0.5	0.213(0.195,0.289)	0.134(0.158,0.235)	0.167(0.125,0.185)

* For each cell, the value outside the brackets represents SD, while the first and second numbers inside the brackets represent $\mathrm{ESE_m}$ and $\mathrm{ESE_o}$, respectively.

Table 3.13: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$

	ρ \ m	300	500	900
non0 coeff.	0.00	0(0.9250)	0(0.9300)	0(0.9250)
	0.20	0(0.9150)	0(0.9350)	0(0.9450)
	0.50	0(0.8850)	0(0.9250)	0(0.9350)
0 coeff.	0.00	1.24(0.9331)	1.87(0.8754)	2.78(0.7915)
	0.20	1.22(0.9385)	1.96(0.8731)	3.37(0.9138)
	0.50	1.54(0.9315)	1.85(0.8962)	2.67(0.8162)

* $\boldsymbol{\beta}_{15\times1}$ cosists of 2 non-zero coefficients.

Table 3.14: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\eta_0}$

	ρ \ m	300	500	900
non0 coeff.	0.00	0.05(0.7500)	0(0.8850)	0(0.9050)
	0.20	0.08(0.7000)	0(0.8700)	0(0.9000)
	0.50	0.14(0.6300)	0.08(0.6950)	0(0.9000)
0 coeff.	0.00	0.83(0.9423)	2.17(0.8715)	4.93(0.9546)
	0.20	0.93(0.9331)	2.16(0.8777)	5.65(0.9469)
	0.50	1.63(0.9138)	1.89(0.9008)	2.67(0.9346)

* $(\boldsymbol{\eta_0})_{15\times 1}$ cosists of 2 non-zero coefficients.

Table 3.15: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\eta_1}$

	ρ \ m	300	500	900
non0 coeff.	0.00	0.01(0.8550)	0(0.8850)	0(0.9500)
	0.20	0.02(0.8200)	0(0.8850)	0(0.8850)
	0.50	0.03(0.8600)	0(0.9100)	0(0.9150)
0 coeff.	0.00	0.60(0.9623)	1.66(0.9038)	4.13(0.9569)
	0.20	0.54(0.9646)	1.97(0.9000)	4.74(0.9562)
	0.50	1.42(0.9331)	1.65(0.9223)	3.48(0.9246)

* $(\boldsymbol{\eta_1})_{15\times 1}$ cosists of 2 non-zero coefficients.

Table 3.16: Excess risks of $C_{\hat{\boldsymbol{\beta}}}$s

ρ \ m	300	500	900
0.00	0.977204	0.984790	0.991655
0.20	0.977011	0.982707	0.995179
0.50	0.973591	0.981800	0.988152

* The misclassification rates of the Bayes classifiers are 18.776% for the case of $\rho = 0$, 20.23% for the case of $\rho = 0.2$, and 23.52% for the case of $\rho = 0.5$.

In summary, from the two simulated examples in this section, we can conclude

that our proposed method demonstrates good performance for mislabelled low-dimensional datasets, and our classifiers are very efficient, exhibiting almost the same efficiency as the Bayes classifiers in all cases. In addition, we find that increasing the sizes of training datasets within a certain range, even if they contain mislabelled data, can improve the performance of the estimates and classifiers using our proposed two-step estimation method. Additionally, our method displays stability when analysing different sizes of mislabelled datasets with varying sizes of resampled datasets or varying correlations among covariates in the datasets. Lastly, we find that the performance of the estimates of the parameters associated with the flipping probabilities follows the same trend as that of $\boldsymbol{\beta}$ in the model of most interest across different cases. Therefore, for simplicity, we omit the detailed results for $\hat{\boldsymbol{\eta}}_0$ and $\hat{\boldsymbol{\eta}}_1$ from here on, and focus on the estimation of $\boldsymbol{\beta}$, as its accuracy has a more direct impact on the classifier.

3.4 Determining the tuning parameter for the penalty function in the presence of mislabelled data

It is undeniable that the performance of variable selection is influenced by the settings used to determine the values for the tuning parameter of the penalty function. Consequently, in this section, we delve into the settings for tuning parameter selection in the context of mislabelled data. Specifically, we investigate how our proposed method is impacted by various sizes and types of validation datasets. In this section, we initially provide a concise description of the process of selecting the tuning parameters for the penalty function. Subsequently, we discuss in depth the selection of different types and sizes of validation data to identify the tuning parameters in our second-step estimation. More specifically, for our proposed two-step estimation method, the first step estimation only scrutinises data in the subsample, which contains perfectly labelled data. However, the second estimation step examines a composite dataset composed of the corrected labels and other potentially imperfect labels. We are hence intrigued by the prospect of identifying any significant discrepancies when we employ different types of data for validation and the stability of our approach for various types of validation datasets. We also conduct tests on validation datasets of different sizes to ascertain whether the size of the validation dataset exerts a significant influence on

the results.

The simulation study conducted in the preceding sections demonstrates that our proposed method integrates with SCAD. The results obtained from our method for mislabelling data demonstrate its ability to effectively capture significant features while successfully filtering out insignificant features. This capability is essential for constructing accurate and interpretable models.

3.4.1 Modified Leave-P-Out Cross-Validation (mLPOCV)

In this section, we describe the Modified Leave-P-Out Cross-Validation (mLPOCV) process applied to find a tuning parameter λ of the SCAD penalty function.

The same as the traditional Leave-P-Out Cross-Validation (LPOCV), in mLPOCV, the dataset with a size of n is divided into two sets, one for training and another for validation, with the validation dataset of size n_{cv}, $1 \leqslant n_{cv} < n$. While unlike LPOCV testing all validation data at one time, only one data is tested. In detail, at the $k-$th time validation, where $k = 1, \cdots, n_{cv}$, (X_k, y_k) is used for validation. After testing, (X_k, y_k) joins in the training dataset for $(k+1)$-th validation, except for the last time when $k = n_{cv}$.

3.4.2 Selection of validation data from the unresampled dataset for second-step estimation

In this section, we describe mLPOCV$_{\mathrm{UR}}$, a method for tuning parameter selection during the second step of our proposed two-step estimation approach. To construct the validation dataset, we randomly select n_{cv}, $1 \leqslant n_{cv} \leqslant n - m$, observations from the dataset $\{(X_i, y_i)\}$, where $i \in \mathcal{D}^c$ and $|\mathcal{D}^c| = n - m$. Note that in this validation dataset, some labels are incorrect, and the information about the correct labels is unknown. Following mLPOCV with a given search interval and search step, we find a value for λ that maximises the sum of the penalised log-likelihood functions with a form of

$$\sum_{k=1}^{n_{cv}} \left\{ \left[y_k (X_k^T \hat{\boldsymbol{\beta}}_k) + \log\{1 - \pi(X_k)\} \right] - \mathbf{P}_{\lambda_{\boldsymbol{\beta}}}(\|\hat{\boldsymbol{\beta}}_k\|_1) \right\},$$

where $\hat{\boldsymbol{\beta}}_k$ is the estimate of $\boldsymbol{\beta}$ obtained at the k-iteration of mLPOCV, and $\mathrm{logit}(\pi(X_k)) = X_k^T \hat{\boldsymbol{\beta}}_k$.

3.4.3 Selection of validation data from the resampled dataset for second-step estimation

In this section, we present mLPOCV$_{\mathrm{S}}$, a method for selecting tuning parameters during the second step estimation of our proposed two-step estimation method, which uses only the resampled data from the subsample for validation. We randomly selected n_{cv}, where $1 \leqslant n_{cv} \leqslant m$, observations for validation from the subsample $\{(X_i, z_i)\}$, $i \in \mathcal{D}$ and $|\mathcal{D}| = m$. It is worth noting that all labels in the validation dataset are correct. Following mLPOCV with a given search interval and search step, we find a value for λ that maximises the sum of the penalised log-likelihood functions with a form of

$$\sum_{k=1}^{n_{cv}} \left\{ \left[z_k (X_k^T \hat{\beta}_k) + \log\{1 - \pi(X_k)\} \right] - \mathbf{P}_{\lambda_\beta}(\|\hat{\beta}_k\|_1) \right\},$$

where $\hat{\beta}_k$ is the estimate of β obtained at the k-iteration of mLPOCV, and $\mathrm{logit}(\pi(X_k)) = X_k^T \hat{\beta}_k$.

3.4.4 Selection of validation data from resampled and unresampled datasets for second-step estimation

In this section, we introduce a method called mLPOCV$_{\mathrm{CD}}$ for tuning parameter selection at the second step estimation of our method. In mLPOCV$_{\mathrm{CD}}$, the validation data are selected from both the resampled dataset, $\{(X_i, z_i)\}$, $i \in \mathcal{D}$ and $|\mathcal{D}| = m$, and unresampled datasets $\{(X_i, y_i)\}$, where $i \in \mathcal{D}^c$ and $|\mathcal{D}^c| = n - m$.

In detail, we randomly select n_{cv} observations from the resampled dataset $\{(X_i, z_i)\}$, where $i \in \mathcal{D}$ and $|\mathcal{D}| = m$, and n_{cv} observations from the unresampled dataset $\{(X_i, y_i)\}$, where $i \in \mathcal{D}^c$ and $|\mathcal{D}^c| = n - m$. We have $1 \leqslant n_{cv} < \min(m, n - m)$. These observations are then paired to form validation sets denoted by $\{ ((X_{\mathcal{D}})_k,\, z_k), ((X_{\mathcal{D}^c})_k,\, y_k) \}$, where $k = 1, \cdots, n_{cv}$. For a given search interval and step and following mLPOCV, the tuning parameter λ is the one to maximise the sum of the following log-likelihood functions

$$\sum_{k=1}^{n_{cv}} \Big\{ \Big[z_k ((X_{\mathcal{D}})_k^T \hat{\beta}_k) + \log(1 - \pi_z((X_{\mathcal{D}})_k)) \Big] + \Big[y_k ((X_{\mathcal{D}^c})_k^T \hat{\beta}_k) + \log(1 - \pi_y((X_{\mathcal{D}^c})_k)) \Big] - \mathbf{P}_{\lambda_\beta}(\|\hat{\beta}_k\|_1) \Big\},$$

where $\hat{\boldsymbol{\beta}}_k$ is the estimate of $\boldsymbol{\beta}$ obtained at the k-iteration of mLPOCV, and $\text{logit}(\pi_z(X_k)) = (X_{\mathcal{D}})_k^T \hat{\boldsymbol{\beta}}_k$ and $\text{logit}(\pi_y(X_k)) = (X_{\mathcal{D}^c})_k^T \hat{\boldsymbol{\beta}}_k$.

3.4.5 Simulation study

In this section, we have a simulation example to compare the performance of the above introduced methods from Section 3.4.2-3.4.4 for the tuning parameter selection at the second step estimation.

We set the parameters of the model (3.1) in this section as

$$\begin{aligned}
&\boldsymbol{\beta} = (0, \beta_2,\ 0,\ \beta_4,\ \beta_5,\ 0, \beta_7 \cdots, 0)_{p\times 1}^T, && \beta_2 = 1, \beta_4 = -0.85, \beta_5 = 1, \beta_7 = -0.85;\\
&\boldsymbol{\eta_0} = (0, \eta_{0,2}, \eta_{0,3}, 0, \cdots, 0)_{p\times 1}^T, && \eta_{0,2} = -0.65, \eta_{0,3} = 1.2;\\
&\boldsymbol{\eta_1} = (0, \eta_{1,2}, \eta_{1,3}, 0, \cdots, 0)_{p\times 1}^T, && \eta_{1,2} = 1.5, \eta_{1,3} = -1,
\end{aligned}$$

where $p = 120$. As described in Section 3.3, we initially generate a dataset of $N = 15000$ data points, where the correlation between $x_{j_1 i}$ and $x_{j_2 i}$ is determined by $\rho^{|j_1 - j_2|}$, with $\rho = 0.2$ and $j_{1,2} = 2, \cdots, 120$.

In this section, we use a fixed training dataset size of $n = 1000$ and a resampled dataset size of $m = 300$. For the validation dataset, we consider different sizes of n_{cv}, specifically $n_{cv} = 10, 20, 30, 40$, in this example. We set the step size as 0.15 for each tuning parameter search process, and the search interval for λ related to $\boldsymbol{\beta}$ as $[0.05, 0.5]$, and for both $\boldsymbol{\eta_0}$ and $\boldsymbol{\eta_1}$ as $[0.065, 0.65]$. We denote the results obtained from the method described in Section 3.4.2 as $\text{mLPOCV}_{\text{UR}}$, the results from Section 3.4.3 as mLPOCV_{S}, and the results from Section 3.4.4 as $\text{mLPOCV}_{\text{CD}}$. Table 3.17 to Table 3.19 present the recorded results of the simulation study conducted in this section.

Table 3.17 records the AMRSEs and MMRSEs of the estimates for $\boldsymbol{\beta}$. It demonstrates that, when using the same size of validation dataset, the performance of $\hat{\boldsymbol{\beta}}$s obtained from the three methods is remarkably similar. The $\hat{\boldsymbol{\beta}}$s from $\text{mLPOCV}_{\text{UR}}$ perform marginally better than the other two in some cases, but the difference is not substantial. As the size of the validation dataset n_{cv} increases, we notice a trend of decreasing AMRSEs and MMRSEs for all estimates. When $n_{cv} = 40$, the estimates in this example are closest to the true values of $\boldsymbol{\beta}$.

Table B.1 lists the ESEs of $\hat{\boldsymbol{\beta}}$s obtained from the three methodologies, as well as the

true values denoted as SDs. The results once more validate the effective performance of the sandwich formula (3.5), and all estimates from different methods are closely aligned. For further details, see Appendix B.

Table 3.18 displays the performance of variable selection by the three methods, including the numbers of incorrect non-zero and zero estimators and CPs for non-zero and zero coefficients of β respectively. It can be observed that there are few or no incorrect zero estimators for non-zero coefficients and small sizes of non-zero estimators for zero coefficients. The results reaffirm that our proposed method, which employs SCAD, can identify almost all significant features and filter out the most insignificant features, thereby contributing to accurate estimation and generating interpretable models. It is also worth noting that for all tested methods, CPs for non-zero coefficients tend to increase as the size of the validation dataset grows, with a notable rise when n_{cv} increases from 10 to 20.

Table 3.19 showcases the excess risks of the classifiers from the three methods and the computing time under different settings. Firstly, we observe that under identical settings (with the same n_{cv}), the efficiency of all classifiers and computation time are very similar across diverse cases. As anticipated, computation time escalates with an increase in data selected for validation. Moreover, as the size of the validation dataset expands, fewer labels are misclassified and the classifiers perform more closely to the Bayes classifier, as evidenced by the excess risks approaching 1. However, the improvement does not consistently increase with the amount of validation data, as observed from the numerical results showing a relatively small increase in excess risks when n_{cv} is increased from 20 to 40.

Based on the simulated example in this section, we can confidently conclude that our proposed method can maintain consistent performance, regardless of different types and sizes of data being filtered out for validation. This is evident from the minimal difference observed when using various types and numbers of data from the validation set in the second-step estimation to determine the tuning parameters λ. Nevertheless, to circumvent the problem of overfitting, given that the resampled data has already been used for validation to find λ in the first step estimation, we choose to utilise the validation data from the unresampled datasets for the second-step estimation, as described in Section 3.4.2. Finally, taking into account both the accuracy and efficiency of the classifier, we have set the validation dataset size to

$n_{cv} = 20$ for the simulation study in other sections' tuning parameter selection, unless otherwise specified.

Table 3.17: AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\beta}}$s from methods using different validation datasets

n_{cv}	10	20	30	40
$\text{mLPOCV}_{\text{UR}}$	2.840(1.4940)	2.392(1.1165)	2.410(0.9912)	1.949(0.8526)
mLPOCV_{S}	2.863(1.5500)	2.403(1.1165)	2.422(0.9912)	1.972(0.8702)
$\text{mLPOCV}_{\text{CD}}$	2.868(1.5324)	2.395(1.1165)	2.421(0.9912)	1.967(0.8579)

* All values recorded in this table multiplied by 10^{-3} are true values.

Table 3.18: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$

	n_{cv}	10	20	30	40
non0 coeff.	$\text{mLPOCV}_{\text{UR}}$	0.01(0.642500)	0.01(0.722500)	0(0.705000)	0(0.760000)
	mLPOCV_{S}	0.02(0.647500)	0.01(0.722500)	0(0.700000)	0(0.752500)
	$\text{mLPOCV}_{\text{CD}}$	0.02(0.645000)	0.01(0.722500)	0(0.700000)	0(0.752500)
0 coeff.	$\text{mLPOCV}_{\text{UR}}$	10.02(0.915172)	12.20(0.896897)	13.07(0.889052)	13.98(0.880603)
	mLPOCV_{S}	9.98(0.917069)	12.18(0.897069)	13.03(0.889397)	14.01(0.880345)
	$\text{mLPOCV}_{\text{CD}}$	10.07(0.916207)	12.13(0.897414)	13.05(0.889224)	14.04(0.880086)

* $\boldsymbol{\beta}_{120\times 1}$ cosists of 4 non-zero coefficients.

Table 3.19: Excess risks of various classifiers and computing time (in brackets)

n_{cv}	10	20	30	40
$\text{mLPOCV}_{\text{UR}}$	0.894454(17:14:14)	0.915602(22:54:21)	0.915182(26:15:46)	0.921667(34:11:54)
mLPOCV_{S}	0.892386(16:24:30)	0.915812(23:54:58)	0.915392(24:35:58)	0.920462(30:59:16)
$\text{mLPOCV}_{\text{CD}}$	0.892319(16:40:35)	0.915882(18:52:02)	0.915462(26:00:27)	0.920603(29:55:04)

* The misclassification rates of the Bayes classifiers are 23.932%.

* We implemented each experiment with 100 independent trials using C++ and ran them on 1 *Intel Xeon 6138 CPUs*. For each experiment, we requested 15 cores.

3.5 The order of estimation in the second-step estimation

As described in Section 3.1, the second-step estimation of our method has an iterative process performed sequentially for estimating $\boldsymbol{\eta_0}$, $\boldsymbol{\eta_1}$ and $\boldsymbol{\beta}$. Therefore, we would like

to explore how stable our method is by testing different estimation sequences for the three unknown parameters.

In detail, there are six different estimation sequences in total. However, we only discuss four sequences listed below, as we reckon that the performance of the other two sequences unlisted to be similar to the sequence **Order3** and **Order4**, which swaps the order of estimation of $\boldsymbol{\eta_0}$ and $\boldsymbol{\eta_1}$.

$$\textbf{Order1: } \hat{\boldsymbol{\eta}}_{\mathbf{0}} \rightarrow \hat{\boldsymbol{\eta}}_{\mathbf{1}} \rightarrow \hat{\boldsymbol{\beta}}.$$

$$\textbf{Order2: } \hat{\boldsymbol{\eta}}_{\mathbf{1}} \rightarrow \hat{\boldsymbol{\eta}}_{\mathbf{0}} \rightarrow \hat{\boldsymbol{\beta}}.$$

$$\textbf{Order3: } \hat{\boldsymbol{\eta}}_{\mathbf{1}} \rightarrow \hat{\boldsymbol{\beta}} \rightarrow \hat{\boldsymbol{\eta}}_{\mathbf{0}}.$$

$$\textbf{Order4: } \hat{\boldsymbol{\beta}} \rightarrow \hat{\boldsymbol{\eta}}_{\mathbf{0}} \rightarrow \hat{\boldsymbol{\eta}}_{\mathbf{1}}.$$

A simulation study is carried out in the following section with the different corrupted datasets so we can find out using different estimation sequences in the second step estimation, how our method performs with corrupted datasets under different settings.

3.5.1 Simulation study

In this section, we compare the performance of estimates obtained by iterating through different estimation orders as **Order1**-**Order4**, introduced above. It is important to note that convergence failures in regression are more likely to occur, especially when the sample size is small (Allison, 2008). In case of such a failure, at the k-th, $k = 1, \cdots$, iteration, we set the estimates of $\boldsymbol{\eta_0}$ or $\boldsymbol{\eta_1}$ to $\mathbf{0}$ and proceed to the next parameter estimation.

Specifically, we test our proposed method, which carries out different sequences of estimations, on two corrupted datasets. We have the settings of the three parameters in two cases as follows:

Case 1:

$$\begin{aligned}
&\boldsymbol{\beta} = (0, \beta_2,\ 0,\ \beta_4, \cdots, 0)^{\mathsf{T}}_{\mathrm{p}\times 1}, && \beta_2 = 2, \beta_4 = -1.5;\\
&\boldsymbol{\eta_0} = (0, \eta_{0,2}, \eta_{0,3}, \eta_{0,4}, \eta_{0,5}, \cdots, 0)^{\mathsf{T}}_{\mathrm{p}\times 1}, && \eta_{0,2} = \eta_{0,4} = -0.65,\ \eta_{0,3} = \eta_{0,5} = 1.2;\\
&\boldsymbol{\eta_1} = (0, \eta_{1,2}, \eta_{1,3}, \eta_{1,4}, \eta_{1,5}, \cdots, 0)^{\mathsf{T}}_{\mathrm{p}\times 1}, && \eta_{1,2} = \eta_{1,4} = 1.5,\ \eta_{1,3} = \eta_{1,5} = -1,
\end{aligned}$$

where $p = 15$.

Case 2:

$$\begin{aligned}
&\boldsymbol{\beta} = (0, \beta_2,\ 0,\ \beta_4, \cdots, 0)^T_{p\times 1}, && \beta_2 = 2, \beta_4 = -1.5;\\
&\boldsymbol{\eta_0} = (0, \eta_{0,2}, \eta_{0,3}, \cdots, 0)^T_{p\times 1}, && \eta_{0,2} = -0.65, \eta_{0,3} = 1.2;\\
&\boldsymbol{\eta_1} = (0, \eta_{1,2}, \eta_{1,3}, \cdots, 0)^T_{p\times 1}, && \eta_{1,2} = 1.5, \eta_{1,3} = -1,
\end{aligned}$$

where $p = 150$.

The correlation between the generated covariates, $x_{j_1 i}$ and $x_{j_2 i}$, $i = 1, \cdots, 15000$, is determined by $0.5^{|j_1 - j_2|}$, for $j_{1,2} = 2, \cdots, p$. We randomly select $n = 1000$ data to construct the training dataset and discuss the resampled dataset with sizes of $m = 200, 400$. Tables 3.20-3.22 record the performance of the method in this section.

Table 3.20 records AMRSEs and MMRSEs of $\hat{\boldsymbol{\beta}}$s from the two-step estimation method carrying out different estimation sequences of the unknown parameters in the second step estimation. It shows that studying the same corrupted dataset with the same resampled data, AMRSEs, and MMRSEs of $\hat{\boldsymbol{\beta}}$s are very similar across the methods testing on the four different sequence estimations. In detail, when $p = 15$, a negligible difference among the estimates is observed when $m = 200$, while for other cases, the numerical results are strikingly close or identical.

The ESEs of estimates for non-zero coefficients of $\boldsymbol{\beta}$ from different experiments and the true values SDs are recorded in Table B.2, which shows the same findings as the simulation study in the previous sections, and we include it in the Appendix B.

Table 3.21 showcases the performance of variable selection for our proposed method, carrying out different estimation sequences at the second step estimation. It is obvious that methods having different iterating orders at the second step estimation have close performance across various cases, which is evidenced by none of incorrect zero estimators in most scenarios (except when $p = 15$ and $n = 200$, there are negligible numbers for **Order 3** and **Order4**), and the similar counts of incorrect non-zero estimators of zero coefficients of $\boldsymbol{\beta}$. Additionally, the CPs for non-zero and zero coefficients are consistent across different cases.

Table 3.22 demonstrates that when studying the same datasets, the classifiers from our proposed method conducting different sequence estimations at the second step estimation have almost the same efficiency. Meanwhile, all the classifiers exhibit

comparable performance to the Bayes classifiers in all scenarios.

In conclusion, from the simulation example in this section, we observe that our two-step estimation method, carrying out different orders of estimation in the second step, has a negligible impact on the simulation results. In this way, the consistency of our proposed method is demonstrated. In our study, unless otherwise stated, we chose **Order1** as the order of estimation for all simulation studies in other sections. As explained at the beginning of the section, we would assign $\mathbf{0}$ to the estimates of $\boldsymbol{\eta_0}$ or $\boldsymbol{\eta_1}$ if the estimation cannot converge during the iterations. In doing so, we aim to avoid the inaccurate estimations of $\boldsymbol{\eta_0}$ or $\boldsymbol{\eta_1}$ affecting the subsequent estimation process.

Table 3.20: AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\beta}}$ from the methods employing different estimation orders for the three unknown parameters

Case 1: $\mathrm{p} = 15$	**Order1**	**Order2**	**Order3**	**Order4**
$\mathrm{m} = 200$	7.561(3.222)	7.201(3.188)	7.186(2.791)	7.450(2.2867)
$\mathrm{m} = 400$	1.369(1.017)	1.363(1.016)	1.341(0.992)	1.340(0.995)
Case 2: $\mathrm{p} = 150$	**Order1**	**Order2**	**Order3**	**Order4**
$\mathrm{m} = 200$	1.150(0.210)	1.150(0.210)	1.150(0.210)	1.150(0.210)
$\mathrm{m} = 400$	0.422(0.136)	0.422(0.136)	0.416(0.139)	0.405(0.125)

* All values recorded in this table multiplied by 10^{-3} are true values.

Table 3.21: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$

Case 1: $\mathrm{p} = 15$		**Order1**	**Order2**	**Order3**	**Order4**
non0 coeff.	$\mathrm{m} = 200$	0(0.920000)	0(0.925000)	0.01(0.915000)	0.02(0.915000)
	$\mathrm{m} = 400$	0(0.995000)	0(0.995000)	0(0.995000)	0(0.995000)
0 coeff.	$\mathrm{m} = 200$	1.54(0.943846)	1.55(0.945385)	1.02(0.967692)	0.74(0.976923)
	$\mathrm{m} = 400$	1.25(0.936923)	1.23(0.939231)	1.15(0.942308)	1.01(0.953846)
Case 2: $\mathrm{p} = 150$		**Order1**	**Order2**	**Order3**	**Order4**
non0 coeff.	$\mathrm{m} = 200$	0(0.820000)	0(0.820000)	0(0.820000)	0(0.820000)
	$\mathrm{m} = 400$	0(0.880000)	0(0.880000)	0(0.880000)	0(0.890000)
0 coeff.	$\mathrm{m} = 200$	9.76(0.934459)	9.76(0.934459)	9.76(0.934459)	9.76(0.934459)
	$\mathrm{m} = 400$	16.22(0.891622)	16.22(0.891622)	16.22(0.891824)	16.23(0.891824)

* Both $\boldsymbol{\beta}_{15\times 1}$ and $\boldsymbol{\beta}_{150\times 1}$ have 2 non-zero coefficients.

Table 3.22: Excess risks of $C_{\hat{\beta}}$s

Case 1: $p = 15$	**Order1**	**Order2**	**Order3**	**Order4**
$m = 200$	0.952970	0.951931	0.955316	0.955404
$m = 400$	0.992798	0.992327	0.992609	0.992798
Case 2: $p = 150$	**Order1**	**Order2**	**Order3**	**Order4**
$m = 200$	0.948014	0.947026	0.947111	0.947111
$m = 400$	0.969124	0.974240	0.975328	0.974693

* The misclassification rates of the Bayes classifiers are 20.716% for both cases where $p = 15$ and $p = 150$.

3.6 Estimation using oracle information of flipping probabilities

In this section, we describe the two-step methods under the assumption that the parameters related to flipping probabilities are known, which we refer to as "oracle information" in our study. We explain three different cases where the parameters $\boldsymbol{\eta_0}$ and $\boldsymbol{\eta_1}$ are known. The resampling is still considered in all the approaches in this section, as illustrated in Section 3.3.1.

While the assumption that the parameters associated with the flipping probabilities is known is too idealistic to be true, comparing these methods with our proposed method through the simulated examples that follow provides support for the validity of our proposed method.

3.6.1 $\boldsymbol{\eta_0}$ is known

When only $\boldsymbol{\eta_0}$ is known, the flipping probability associated with $\boldsymbol{\eta_0}$ is also known. Referring to our approach presented in Section 3.1, in the first step, we obtain the MLEs denoted as $\boldsymbol{\beta}^{(0)}$ and $\boldsymbol{\eta}_1^{(0)}$ which are maximisers of $L(\boldsymbol{\beta})$ and $L(\boldsymbol{\eta_1})$ in (3.4) respectively.

When $\boldsymbol{\eta_0}$ is known, the penalised log-likelihood function for the second step estimation is

$$
\begin{aligned}
& L(\boldsymbol{\beta},\ \boldsymbol{\eta_1}) \\
&= \sum_{i \in \mathcal{D}^c} \Big\{ y_i \text{logit}\left[\pi_{y|0}(X_i) + \left\{\pi_{y|1}(X_i) - \pi_{y|0}(X_i)\right\} \pi_z(X_i)\right] + \\
&\qquad \log\left[1 - \pi_{y|0}(X_i) - \left\{\pi_{y|1}(X_i) - \pi_{y|0}(X_i)\right\} \pi_z(X_i)\right] \Big\}
\end{aligned}
$$

$$
\begin{aligned}
&+\sum_{i\in D}(1-z_i)\left\{y_i\log(\pi_{y|0}(X_i))+(1-y_i)\log(1-\pi_{y|0}(X_i))+\right.\\
&\qquad\left.(1-z_i)\log(1-\pi_z(X_i))\right\}\\
&+\sum_{i\in\mathcal{D}} z_i\left\{y_i(X_i^T\boldsymbol{\eta_1})+\log(1-\pi_{y|1}(X_i))+z_i\log(\pi_z(X_i))\right\}\\
&-\mathbf{P}_{\lambda_\beta}(\|\boldsymbol{\beta}\|_1)-\mathbf{P}_{\lambda_{\eta_1}}(\|\boldsymbol{\eta_1}\|_1),
\end{aligned}
\tag{3.16}
$$

where $\pi_{y|0}(X_i)$ for $i=1,\cdots,n$, are constants.

To begin the iteration of the second step estimation, we use $\boldsymbol{\beta}^{(0)}$ and $\boldsymbol{\eta}_1^{(0)}$ as initial values, and then apply the following iterative algorithm: in the k-th, where $k=1,\cdots$, iteration, we use LQA to approximate the log-likelihood penalty function (3.16) as follows:

(1) we apply the Newton-Raphson method to maximise $L(\boldsymbol{\beta}^{(k-1)},\ \boldsymbol{\eta_1})$ with respect to $\boldsymbol{\eta_1}$, and denote the maximiser by $\boldsymbol{\eta}_1^{(k)}$. The tuning parameter $\lambda_{\boldsymbol{\eta}_1^{(k)}}$ is updated and retained for the next estimation step.

(2) we apply the Newton-Raphson method to maximise $L(\boldsymbol{\beta},\ \boldsymbol{\eta}_1^{(k)})$ with respect to $\boldsymbol{\beta}$, and denote the maximiser by $\boldsymbol{\beta}^{(k)}$. The tuning parameter $\lambda_{\boldsymbol{\beta}^{(k)}}$ is updated and retained for the next estimation step.

The above iteration stops upon convergence and we then obtain the maximisers of (3.16) denoted as $(\hat{\boldsymbol{\beta}}_{|\boldsymbol{\eta_0}},\ \hat{\boldsymbol{\eta}}_{1|\boldsymbol{\eta_0}})$.

3.6.2 $\boldsymbol{\eta}_1$ is known

When only $\boldsymbol{\eta}_1$ is known, the estimation process is very similar to the method in Section 3.6.1 introduced above. For the first step estimation, we obtain maximisers of $L(\boldsymbol{\beta})$ and $L(\boldsymbol{\eta_0})$ in (3.4), which are denoted as $\boldsymbol{\beta}^{(0)}$ and $\boldsymbol{\eta}_0^{(0)}$, respectively.

We have the penalised log-likelihood function for the second step as

$$
\begin{aligned}
&L(\boldsymbol{\beta},\ \boldsymbol{\eta_0})\\
&=\sum_{i\in\mathcal{D}^c}\left\{y_i\text{logit}\left[\pi_{y|0}(X_i)+\left\{\pi_{y|1}(X_i)-\pi_{y|0}(X_i)\right\}\pi_z(X_i)\right]+\right.\\
&\qquad\left.\log\left[1-\pi_{y|0}(X_i)-\left\{\pi_{y|1}(X_i)-\pi_{y|0}(X_i)\right\}\pi_z(X_i)\right]\right\}\\
&+\sum_{i\in\mathcal{D}}(1-z_i)\left\{y_i(X_i^T\boldsymbol{\eta_0})+\log(1-\pi_{y|0}(X_i))+(1-z_i)\log(\pi_z(X_i))\right\}
\end{aligned}
$$

$$+\sum_{i\in D} z_i\left\{y_i\log(\pi_{y|1}(X_i))+(1-y_i)\log(1-\pi_{y|1}(X_i))+z_i\log(\pi_z(X_i))\right\}$$
$$-\mathbf{P}_{\lambda_\beta}(\|\boldsymbol{\beta}\|_1)-\mathbf{P}_{\lambda_{\eta_0}}(\|\boldsymbol{\eta_0}\|_1), \tag{3.17}$$

where $\pi_{y|1}(X_i)=P(y_i=1|X_i,z_i=1)$, $i=1,\cdots,n$, are constants.

To find the maximisation of the penalty function (3.17), we use the following iterative estimation process. In each of the k iterations, where $k=1,\cdots,$

(1) we apply the Newton-Raphson method to maximise $L(\boldsymbol{\beta}^{(k-1)},\ \boldsymbol{\eta_0})$ with respect to $\boldsymbol{\eta_0}$, and denote the maximiser by $\boldsymbol{\eta}_0^{(k)}$. The tuning parameter $\lambda_{\boldsymbol{\eta}_0^{(k)}}$ is updated and retained for the next estimation step.

(2) we apply the Newton-Raphson method to maximise $L(\boldsymbol{\beta},\ \boldsymbol{\eta}_0^{(k)})$ with respect to $\boldsymbol{\beta}$, and denote the maximiser by $\boldsymbol{\beta}^{(k)}$. The tuning parameter $\lambda_{\boldsymbol{\beta}^{(k)}}$ is updated and retained for the next estimation step.

The above iteration stops until convergence. The MLEs of the unknown parameters, denoted as $(\hat{\boldsymbol{\beta}}_{|\eta_1},\ \hat{\boldsymbol{\eta}}_{\mathbf{0}|\boldsymbol{\eta}_1})$, are then obtained.

3.6.3 Both η_0 and η_1 are known

When both parameters $\boldsymbol{\eta_0}$ and $\boldsymbol{\eta_1}$ are known, only $\boldsymbol{\beta}$ in (3.1) is needed to be estimated. After resampling the dataset, as introduced in Section 3.1, we maximise $L(\boldsymbol{\beta})$ in (3.4) and achieve the maximiser denoted as $\boldsymbol{\beta}^{(0)}$. $\boldsymbol{\beta}^{(0)}$ as the initiator joins in the next step estimation.

In the second step estimation, we aim to find the maximiser of the penalised log-likelihood function (3.18), and the MLE of the unknown parameter $\boldsymbol{\beta}$ is denoted as $\hat{\boldsymbol{\beta}}_{|\eta_0,\eta_1}$. The penalised log-likelihood function has the form as follows

$$\begin{aligned}
&L(\boldsymbol{\beta})\\
&=\sum_{i\in\mathcal{D}^c}\left\{y_i\text{logit}\left[\pi_{y|0}(X_i)+\left\{\pi_{y|1}(X_i)-\pi_{y|0}(X_i)\right\}\pi_z(X_i)\right]+\right.\\
&\qquad\left.\log\left[1-\pi_{y|0}(X_i)-\left\{\pi_{y|1}(X_i)-\pi_{y|0}(X_i)\right\}\pi_z(X_i)\right]\right\}\\
&+\sum_{i\in D}\left[z_i(X_i^T\boldsymbol{\beta})+\log(1-\pi_z(X_i))\right]-\mathbf{P}_{\lambda_\beta}(\|\boldsymbol{\beta}\|_1),
\end{aligned} \tag{3.18}$$

where both $\pi_{y|0}(X_i)$ and $\pi_{y|1}(X_i)$ for $i=1,\cdots,n$, are constants.

3.6.4 Simulation study

In this section, we design a simulated example having different corrupted datasets to compare the performance of our proposed method with the methods having oracle information described in Section 3.6.1-3.6.3.

The parameters in model (3.1) are set as follows:

$$\begin{aligned}
&\boldsymbol{\beta} = (0, \beta_2, \beta_3,\ 0, \cdots, 0)^{\mathsf{T}}_{p\times 1}, && \beta_2 = 2, \beta_3 = -1.5;\\
&\boldsymbol{\eta_0} = (0, \eta_{0,2}, \eta_{0,3}, 0, \cdots, 0)^{\mathsf{T}}_{p\times 1}, && \eta_{0,2} = -0.65, \eta_{0,3} = 1.2;\\
&\boldsymbol{\eta_1} = (0, \eta_{1,2}, \eta_{1,3}, 0, \cdots, 0)^{\mathsf{T}}_{p\times 1}, && \eta_{1,2} = 1.5, \eta_{1,3} = -1,
\end{aligned}$$

where $p = 15$. The correlation between $x_{j_1 i}$ and $x_{j_2 i}$, where $i = 1, \cdots, N$, is defined as $\rho^{|j_1 - j_2|}$, with $j_{1,2} = 2, \cdots, 15$, and $N = 15000$ as introduced in Section 3.3. We test the four methods under three different values of ρ, which are 0, 0.2, and 0.5. The size of each training dataset is fixed as $n = 1000$, and the resampled dataset is fixed as $m = 300$. Tables 3.23-3.25 present the numerical results for this simulation study.

Table 3.23 reveals that the estimates of the four methods exhibit highly similar performance when examining the same dataset. This is evidenced by the values of AMRSEs and MMRSEs of $\hat{\boldsymbol{\beta}}$s, $\hat{\boldsymbol{\beta}}_{|\boldsymbol{\eta}_0}$s, $\hat{\boldsymbol{\beta}}_{|\boldsymbol{\eta}_1}$s and $\hat{\boldsymbol{\beta}}_{|\boldsymbol{\eta}_0,\boldsymbol{\eta}_1}$s. This substantiates that our proposed method exhibits exceptional performance, as there is minimal difference in estimation accuracy among all the methods. When the covariates are highly correlated ($\rho = 0.5$), the accuracy of our proposed method diminishes slightly, yet the variance in estimation accuracy with the other three methods remains relatively insignificant.

Table B.3 documents ESEs of these estimates alongside the true values SDs, demonstrating the robust performance of the sandwich formula (3.5) and the comparable performance across these four estimates. Detailed results can be found in Appendix B.

The variable selection performance of the methods is illustrated in Table 3.24. Our methods perform almost identically to the other methods incorporating oracle information, as indicated by the near-equal values of CPs for non-zero and zero coefficients. Once more, the simulation study confirms that the methods integrating with the SCAD penalty function can identify all significant features. Additionally, in

this example, $\hat{\boldsymbol{\beta}}$s have fewer incorrect non-zero estimators and higher CPs for the zero coefficients, albeit the difference is not substantial, compared to other methods with oracle information. This can be attributed to our method estimating more unknown parameters than the other three, necessitating more iterations before convergence, and thus providing more opportunities for variable selection processes.

The excess risk of the classifiers presented in Table 3.25 demonstrates that in all cases, all classifiers perform remarkably close to the Bayes classifier. This is evidenced by all numerical results approaching 1 and the similar efficiency across different scenarios. It is worth noting that our classifier $C_{\hat{\boldsymbol{\beta}}}$s exhibits nearly identical efficiency to $C_{\hat{\boldsymbol{\beta}}_{|\eta_0,\eta_1}}$s, which are derived from the method possessing precise information concerning both unknown parameters related to flipping probabilities.

In conclusion, the simulation study in this section allows us to compare our proposed method with other methods that have knowledge of the parameters associated with the flipping probabilities of labels. The almost identical numerical results provide compelling evidence supporting the effectiveness of our proposed estimation method. This method, which considers resampling, performs remarkably well when dealing with mislabelled datasets, even in the absence of exact information about mislabelling in the dataset.

Table 3.23: AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\beta}}$s, $\hat{\boldsymbol{\beta}}_{|\eta_0}$s, $\hat{\boldsymbol{\beta}}_{|\eta_1}$s and $\hat{\boldsymbol{\beta}}_{|\eta_0,\eta_1}$s.

ρ	0	0.2	0.5
$\hat{\boldsymbol{\beta}}$	2.399(1.3730)	2.411(1.7196)	3.737(2.0319)
$\hat{\boldsymbol{\beta}}_{\|\eta_0}$	2.452(1.4024)	2.387(1.5122)	2.502(1.7185)
$\hat{\boldsymbol{\beta}}_{\|\eta_1}$	2.318(1.4081)	2.254(1.4478)	2.652(1.6394)
$\hat{\boldsymbol{\beta}}_{\|\eta_0,\eta_1}$	2.344(1.3851)	2.204(1.4481)	2.390(1.4814)

* All values recorded in this table multiplied by 10^{-3} are true values.

Table 3.24: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for β

	ρ	0	0.2	0.5
non0 coeff.	$\hat{\beta}$	0(0.925000)	0(0.915000)	0(0.890000)
	$\hat{\beta}_{\|\eta_0}$	0(0.905000)	0(0.915000)	0(0.895000)
	$\hat{\beta}_{\|\eta_1}$	0(0.940000)	0(0.935000)	0(0.915000)
	$\hat{\beta}_{\|\eta_0,\eta_1}$	0(0.930000)	0(0.930000)	0(0.910000)
0 coeff.	$\hat{\beta}$	1.24(0.933077)	1.28(0.925385)	1.46(0.920769)
	$\hat{\beta}_{\|\eta_0}$	1.38(0.923077)	1.38(0.920000)	1.56(0.914615)
	$\hat{\beta}_{\|\eta_1}$	1.28(0.927692)	1.39(0.922308)	1.46(0.923077)
	$\hat{\beta}_{\|\eta_0,\eta_1}$	1.63(0.901538)	1.75(0.898462)	1.83(0.897692)

* $\beta_{15\times1}$ cosists of 2 non-zero coefficients.

Table 3.25: Excess risks of various classifiers.

ρ	0	0.2	0.5
$ER(C_{\hat{\beta}})$	0.977204	0.976257	0.970137
$ER(C_{\hat{\beta}_{\|\eta_0}})$	0.979141	0.978145	0.978044
$ER(C_{\hat{\beta}_{\|\eta_1}})$	0.979038	0.977673	0.977374
$ER(C_{\hat{\beta}_{\|\eta_0,\eta_1}})$	0.979958	0.977389	0.975772

* The misclassification rates of the Bayes classifiers are 18.776% for the case of $\rho = 0$, 20.23% for the case of $\rho = 0.2$, and 23.52% for the case of $\rho = 0.5$.

3.7 Classifiers from different ways to cope with mislabelling

In the following sections, we describe three alternative approaches that handle mislabelling differently from our method. The same corrupted dataset (X_i, y_i), $i = 1, \cdots, n$, as described in Section 3.1, is discussed.The simulation study has been designed to compare the performance of these methods with our proposed approach.

3.7.1 Estimation on datasets with all labels corrected

This section describes the estimation method for studying a dataset where all labels have been corrected. Although it is difficult to correct each label in a real-world problem, we make an ideal assumption to obtain a fully perfectly labelled dataset, meaning $z_i, i = 1, \cdots, n$, are observable.

We then have the penalised log-likelihood function of $\boldsymbol{\beta}$ as

$$L(\boldsymbol{\beta}) = \sum_{i}^{n} \left[z_i (X_i^T \boldsymbol{\beta}) + \log\{1 - \pi_z(X_i)\} \right] - \mathbf{P}_{\lambda_\beta}(\|\boldsymbol{\beta}\|_1). \tag{3.19}$$

The maximiser of (3.19) can be achieved by penalised Iteratively Re-weighted Least Squares method. The estimate denoted as $\hat{\boldsymbol{\beta}}^*$ is the MLE of $\boldsymbol{\beta}$.

For a new observation X, if

$$\frac{\exp(X^T \hat{\boldsymbol{\beta}}^*)}{1 + \exp(X^T \hat{\boldsymbol{\beta}}^*)} > \frac{1}{2}, \tag{3.20}$$

we classify it in the group labelled as 1. The classifier trained by all perfect labelled datasets is as follows

$$C_{\hat{\boldsymbol{\beta}}^*}(X) = \begin{cases} 1 & \text{if (3.20) holds} \\ 0 & \text{otherwise.} \end{cases}$$

3.7.2 Estimation on raw datasets

In this section, we outline the method that studies raw observations with mislabelling $(X_i,\ y_i)$, $i = 1, \cdots, n$, and no perfect label is known. The penalised log-likelihood function with respect to $\boldsymbol{\beta}$ is

$$L(\boldsymbol{\beta}) = \sum_{i}^{n} \left[y_i (X_i^T \boldsymbol{\beta}) + \log\{1 - \pi_y(X_i)\} \right] - \mathbf{P}_{\lambda_\beta}(\|\boldsymbol{\beta}\|_1). \tag{3.21}$$

We denote the maximiser of (3.21) as $\hat{\boldsymbol{\beta}}_R$. For a new observation X, if

$$\frac{\exp(X^T \hat{\boldsymbol{\beta}}_R)}{1 + \exp(X^T \hat{\boldsymbol{\beta}}_R)} > \frac{1}{2}, \tag{3.22}$$

we classify it into the group labelled as 1. Specifically, we have a classifier trained from the original mislabelled data, as

$$C_{\hat{\boldsymbol{\beta}}_R}(X) = \begin{cases} 1 & \text{if (3.22) holds} \\ 0 & \text{otherwise.} \end{cases}$$

3.7.3 Estimation on combined datasets having resampled and unresampled data without considering flipping probabilities

In this section, the method studies datasets consisting of resampled and unresampled data. As described in Section 3.3.1, m observations from $(X_i,\ y_i)$, $i = 1, \cdots, n$, are randomly selected and resampled. The subsample, $(X_i, z_i, y_i), i \in \mathcal{D}$, has perfect label z_i which can only be observed in this resampled dataset. The method in this section does not take into account the flipping probabilities and the penalised log-likelihood function of $\boldsymbol{\beta}$ is

$$\begin{aligned} L(\boldsymbol{\beta}) = &\sum_{i\in\mathcal{D}} \Big[z_i \text{logit}(\pi_z(X_i)) + \log(1-\pi_z(X_i)) \Big] \\ &+ \sum_{i\in\mathcal{D}^c} \Big[y_i \text{logit}(\pi_y(X_i)) + \log(1-\pi_y(X_i)) \Big] - \mathbf{P}_{\lambda_{\boldsymbol{\beta}}}(\|\boldsymbol{\beta}\|_1). \end{aligned} \tag{3.23}$$

We denote the maximiser of (3.23) as $\hat{\boldsymbol{\beta}}_{\text{CD}}$. For a new observation X, if

$$\frac{\exp(X^T\hat{\boldsymbol{\beta}}_{\text{CD}})}{1+\exp(X^T\hat{\boldsymbol{\beta}}_{\text{CD}})} > \frac{1}{2}, \tag{3.24}$$

we classify it into the group labelled as 1. We denote the classifier that is trained on a dataset that has been partially corrected, without taking into account the estimation of flipping probabilities as

$$C_{\hat{\boldsymbol{\beta}}_{\text{CD}}}(X) = \begin{cases} 1 & \text{if (3.24) holds} \\ 0 & \text{otherwise.} \end{cases}$$

3.7.4 Simulation study

In this section, two simulation examples are presented to compare the performance of our proposed method with three methods that employ different strategies to handle the mislabelling in the dataset. These methods are discussed in Section 3.7-Section 3.7.3, and also with the method that exclusively studies data from the resampled dataset of size m.

In this section, we denote the estimate of our proposed method as $\hat{\boldsymbol{\beta}}$ and the estimate trained only by the resampled dataset as $\boldsymbol{\beta}^{(0)}$. Other estimates are denoted as

$\hat{\boldsymbol{\beta}}^*$, $\hat{\boldsymbol{\beta}}_R$, and $\hat{\boldsymbol{\beta}}_{CD}$ for the methods introduced in Section 3.7-Section 3.7.3, respectively.

In the first example, the training datasets have different sizes while sharing the same resampled dataset, and in the second example, a fixed-size dataset is paired with resampled datasets of different sizes. For the two examples in this section, we maintain the same parameter settings in (3.1) as follows

$$\begin{aligned}
&\boldsymbol{\beta} = (0, \beta_2, \beta_3, \beta_4, \beta_5, 0, \cdots, 0)^T_{p\times 1}, && \beta_2 = 1, \beta_3 = -0.55, \beta_4 = 0.55, \beta_5 = -1;\\
&\boldsymbol{\eta_0} = (0, \eta_{0,2}, \eta_{0,3}, 0, \cdots, 0)^T_{p\times 1}, && \eta_{0,2} = -0.55, \eta_{0,3} = 0.55;\\
&\boldsymbol{\eta_1} = (0, \eta_{1,2}, \eta_{1,3}, 0, \cdots, 0)^T_{p\times 1}, && \eta_{1,2} = 0.65, \eta_{1,3} = -0.5,
\end{aligned}$$

where $p = 15$. We have the covariance matrix of covariates $x_{[2:p]i}$, $i = 1, \cdots, N$, as $\Sigma_{14\times 14}$, a compound symmetry covariance matrix, with the correlation parameter $\rho = 0.2$.

Example 1: Datasets of different sizes have the same resampled data. In the first example, we investigate training datasets of different sizes with the same resampled dataset of size $m = 400$, and the training datasets with varying sizes as $n = 1000, 1200, 1500$ are tested. Table 3.26-Table 3.29 display the performance of the methods in **Example 1**.

Table 3.26 records the AMRSEs and MMRSEs of four estimates. It is unsurprising that $\hat{\boldsymbol{\beta}}^*$, trained from all perfectly labelled datasets, is the most accurate. Meanwhile, considering resampling and utilising the two-step estimation method, $\hat{\boldsymbol{\beta}}$s trained by the datasets with mislabelling outperform the other three estimates. The two methods that do not account for resampling of mislabelled data perform the worst, with $\hat{\boldsymbol{\beta}}_{CD}$s slightly more accurate than $\hat{\boldsymbol{\beta}}R$ due to partially corrected information. Moreover, it is noteworthy that our proposed method achieves the superior performance of $\hat{\boldsymbol{\beta}}$s compared to $\boldsymbol{\beta}^{(0)}$s. This improvement in performance underscores our method's capacity to enhance the accuracy of estimates obtained from a small clean sample by utilising supplementary information, even in the presence of an unknown amount of mislabelled data. In this example, both the performance of $\hat{\boldsymbol{\beta}}^*$s and $\hat{\boldsymbol{\beta}}$s improves as the dataset size increases. Conversely, $\hat{\boldsymbol{\beta}}_{CD}$s deteriorate as more mislabelled data is added to the training datasets while the resampled datasets remain constant, and $\hat{\boldsymbol{\beta}}_R$s have similar performance for varying sizes of corrupted datasets.

From Table 3.27, we can observe that the sandwich formula (3.5) performs reason-

ably well for $\hat{\boldsymbol{\beta}}^*$s, $\hat{\boldsymbol{\beta}}$s, and $\boldsymbol{\beta}^{(0)}$s, with ESEs of the estimates from Equation (3.5) do not deviate significantly from SDs in all cases. ESEs of $\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$s are far from the true values, SDs, as the estimates are considerably less accurate. We do not provide $\hat{\boldsymbol{\beta}}_{\mathrm{R}}$s' ESE records because this method fails to identify almost all of the non-zero coefficients and cannot provide any ESE information on the non-zero coefficients of $\boldsymbol{\beta}$.

The performance of variable selection is presented in Table 3.28, including the averaged incorrect numbers and CPs for the non-zero and zero coefficients, respectively. Almost all significant features can be identified by our proposed two-step method using SCAD and the methods studying perfectly labelled datasets, which have estimates $\hat{\boldsymbol{\beta}}^*$s and $\boldsymbol{\beta}^{(0)}$s. In comparison, the method studying raw data with mislabelling fails to find the most important features, and the method that corrects part of the mislabels while ignoring label noise during the estimation process performs better than it. Nonetheless, our proposed method still outperforms the two methods that disregard label noise to a notable degree. Our method yields higher CPs for non-zero coefficients compared to the method using only resampled data, and the method using the full corrected datasets in all cases. In this example, we observe that compared to $\boldsymbol{\beta}^{(0)}$s and $\hat{\boldsymbol{\beta}}^*$s, $\hat{\boldsymbol{\beta}}$s have fewer incorrect non-zero estimates for zero coefficients of $\boldsymbol{\beta}$ and yield higher CPs. Additionally, as the size of the dataset increases, these differences among the three methods become more pronounced.

The excess risk of the classifiers in Table 3.29 demonstrates that our proposed classifier is highly efficient, exhibiting performance close to that of the Bayes classifier and similar efficiency to $\mathrm{C}_{\hat{\boldsymbol{\beta}}^*}$ across different cases. It can also be seen that our classifier $\mathrm{C}_{\hat{\boldsymbol{\beta}}}$ is superior to the classifier trained only using the resampled data, and as the size of the corrupted dataset increases with a fixed resampled dataset, the gaps become larger. This indicates that by employing our proposed method, the additional mislabelled data can enhance the performance of the classifier. Classifier $\mathrm{C}_{\hat{\boldsymbol{\beta}}_{\mathrm{R}}}$, obtained by the method using raw data, incorrectly classifies more than half of the labels. Classifier $\mathrm{C}_{\hat{\boldsymbol{\beta}}_{\mathrm{CD}}}$ performs better than classifier $\mathrm{C}_{\hat{\boldsymbol{\beta}}_{\mathrm{R}}}$, but is noticeably inferior to ours and worsens as the dataset size increases. It is safe to conclude that the label noise cannot be ignored to train the LR classifier.

Table 3.26: AMRSEs and MMRSEs (in brackets) of estimates of $\boldsymbol{\beta}$ from different methods handling mislabelled data using various approaches

n	1000	1200	1500
$\hat{\boldsymbol{\beta}}$	5.518(4.4779)	5.441(4.2847)	5.382(3.9351)
$\boldsymbol{\beta}^{(0)}$	7.345(5.407)	7.345(5.407)	7.345(5.407)
$\hat{\boldsymbol{\beta}}^{*}$	2.156(1.5767)	1.616(1.0568)	1.554(0.8334)
$\hat{\boldsymbol{\beta}}_{\mathrm{R}}$	67.527(66.6715)	67.028(66.6694)	67.682(67.7314)
$\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$	53.414(63.3681)	53.143(62.3511)	56.090(63.4979)

* All values recorded in this table multiplied by 10^{-3} are true values.

Table 3.27: SDs and ESEs for non-zero $\boldsymbol{\beta}_{15\times 1}$ coefficient estimates

	n	1000	1200	1500
	$\hat{\boldsymbol{\beta}}$	0.163(0.144,0.214)	0.162(0.141,0.209)	0.165(0.139,0.206)
	$\boldsymbol{\beta}^{(0)}$	0.160(0.148,0.219)	0.160(0.148,0.219)	0.160(0.148,0.219)
β_2	$\hat{\boldsymbol{\beta}}^{*}$	0.089(0.093,0.138)	0.084(0.084,0.125)	0.072(0.075,0.111)
	$\hat{\boldsymbol{\beta}}_{\mathrm{R}}$	-	-	-
	$\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$	1.431(0.004,0.005)	1.402(0.006,0.009)	1.437(0.004,0.005)
	$\hat{\boldsymbol{\beta}}$	0.138(0.131,0.193)	0.147(0.129,0.191)	0.153(0.128,0.189)
	$\boldsymbol{\beta}^{(0)}$	0.127(0.134,0.199)	0.127(0.134,0.199)	0.127(0.134,0.199)
β_3	$\hat{\boldsymbol{\beta}}^{*}$	0.080(0.084,0.124)	0.076(0.076,0.113)	0.065(0.068,0.101)
	$\hat{\boldsymbol{\beta}}_{\mathrm{R}}$	-	-	-
	$\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$	0.811(0,0)	0.811(0,0)	0.813(0,0)
	$\hat{\boldsymbol{\beta}}$	0.150(0.134,0.199)	0.136(0.133,0.196)	0.114(0.133,0.196)
	$\boldsymbol{\beta}^{(0)}$	0.141(0.135,0.200)	0.141(0.135,0.200)	0.141(0.135,0.200)
β_4	$\hat{\boldsymbol{\beta}}^{*}$	0.076(0.084,0.125)	0.062(0.077,0.114)	0.070(0.068,0.101)
	$\hat{\boldsymbol{\beta}}_{\mathrm{R}}$	-	-	-
	$\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$	0.804(0.008,0.011)	0.803(0.001,0.002)	0.807(0,0)
	$\hat{\boldsymbol{\beta}}$	0.158(0.147,0.219)	0.162(0.148,0.219)	0.170(0.146,0.217)
	$\boldsymbol{\beta}^{(0)}$	0.158(0.148,0.219)	0.158(0.148,0.219)	0.158(0.148,0.219)
β_5	$\hat{\boldsymbol{\beta}}^{*}$	0.104(0.093,0.138)	0.099(0.086,0.127)	0.093(0.075,0.112)
	$\hat{\boldsymbol{\beta}}_{\mathrm{R}}$	-	-	-
	$\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$	1.448(0.014,0.021)	1.436(0.005,0.007)	1.449(0.017,0.026)

* For each cell, the value outside the brackets represents SD, while the first and second numbers inside the brackets represent $\mathrm{ESE_m}$ and $\mathrm{ESE_o}$, respectively.

* In some of the trials in a 100-repetition experiment, the method that directly uses the corrupted datasets fails to converge, leading to the inability to generate the SDs, and $\mathrm{ESE_m}$s and $\mathrm{ESE_o}$s of $\hat{\boldsymbol{\beta}}_{\mathrm{R}}$s.

Table 3.28: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$

	n	1000	1200	1500
non0 coeff.	$\hat{\boldsymbol{\beta}}$	0(0.917500)	0.01(0.920000)	0.01(0.920000)
	$\boldsymbol{\beta}^{(0)}$	0.01(0.877500)	0.01(0.877500)	0.01(0.877500)
	$\hat{\boldsymbol{\beta}}^{*}$	0(0.950000)	0(0.94750)	0(0.937500)
	$\hat{\boldsymbol{\beta}}_{\mathrm{R}}$	2.69(-)	2.45(-)	2.09(-)
	$\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$	0.52(0)	0.48(0)	0.68(0)
0 coeff.	$\hat{\boldsymbol{\beta}}$	4.36(0.954545)	3.65(0.882727)	3.41(0.907273)
	$\boldsymbol{\beta}^{(0)}$	6.13(0.875455)	6.13(0.875455)	6.13(0.875455)
	$\hat{\boldsymbol{\beta}}^{*}$	5.83(0.842727)	5.76(0.839091)	5.26(0.836364)
	$\hat{\boldsymbol{\beta}}_{\mathrm{R}}$	3.85(-)	4.47(-)	5.31(-)
	$\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$	3.57(0.837273)	4.28(0.805455)	3.99(0.782727)

* $\boldsymbol{\beta}_{15\times 1}$ cosists of 4 non-zero coefficients.

* In some of the trials in a 100-repetition experiment, the method that directly utilises the corrupted datasets fails to attain convergence, leading to an inability to generate the ESEs of $\hat{\boldsymbol{\beta}}_{\mathrm{R}}$s and the associated coverage probabilities.

Table 3.29: Excess risks of various classifiers

n	1000	1200	1500
$\mathrm{ER}(\mathrm{C}_{\hat{\boldsymbol{\beta}}})$	0.965114	0.965596	0.967876
$\mathrm{ER}(\mathrm{C}_{\boldsymbol{\beta}^{(0)}})$	0.957802	0.957802	0.957802
$\mathrm{ER}(\mathrm{C}_{\hat{\boldsymbol{\beta}}^{*}})$	0.983068	0.990192	0.992735
$\mathrm{ER}(\mathrm{C}_{\hat{\boldsymbol{\beta}}_{\mathrm{R}}})$	0.547227	0.548403	0.545726
$\mathrm{ER}(\mathrm{C}_{\hat{\boldsymbol{\beta}}_{\mathrm{CD}}})$	0.854796	0.827552	0.763302

* The misclassification rates of the Bayes classifiers is 27.056%.

Example 2: Datasets of the same size with different sizes of resampled datasets In the second example, the training dataset has a fixed size of $n = 1000$, and resampled datasets with different sizes ranging from $m = 300, 400, 500$ are discussed. Table 3.30-Table 3.32 present the results of **Example 2** in this section.

In Table 3.30, the numerical results of AMRSEs and MMRSEs of the estimates show that the performance of $\hat{\boldsymbol{\beta}}$s, $\boldsymbol{\beta}^{(0)}$s, and $\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$s improves when more labels are checked. However, $\hat{\boldsymbol{\beta}}$s have performance closer to $\hat{\boldsymbol{\beta}}^{*}$s and are much more accurate than the other two estimates, especially when there is less resampled data, $m = 300$. It can also be seen that as more data are resampled, with m increasing from 300 to 500, the estimation accuracy of our method has manifestly improved and is closer to

that of the method having all corrected labels in the dataset. $\hat{\boldsymbol{\beta}}_{\mathrm{R}}$s perform the worst, which is consistent with the finding in **Example 1** in this section.

The results in Table B.4 record ESEs of $\hat{\boldsymbol{\beta}}$s from Equation (3.5) and their corresponding true values SDs. We can draw similar conclusions regarding the performance of the estimator as those in Table 3.27 of Example 1 above. These details are provided in Appendix B.

Table 3.31 shows the averaged incorrect numbers and CPs for the non-zero and zero coefficients of $\boldsymbol{\beta}$. Our proposed method, similar to the two methods using perfect labels, is able to identify all important features in almost all cases. The only exception is when $m = 300$, where a small number of significant features were missed in 100 repetitions. However, both methods that ignore label noise in the estimation process miss most of the important features. As more data in the dataset is resampled, CPs of non-zero coefficients obtained by our method become closer to those of the method correcting all training data, while CPs of zero coefficients by our method are higher than those related to $\hat{\boldsymbol{\beta}}^*$. Additionally, CPs obtained by our method are always higher than the method only using a subsample. The method using all raw data has missed almost all important features and performs the worst. Furthermore, compared to our method, which does not account for label noise during estimation, the method that partially corrects mislabelled data misses more important features. The results again demonstrate that our method has good performance in variable selection for mislabelled datasets and can find the important features to generate interpretable models.

As shown in Table 3.32, the performance of our classifier $C_{\hat{\boldsymbol{\beta}}}$s exhibits similar efficiency to the Bayes classifier, and as the resampled data size increases in this example, the performance of $C_{\hat{\boldsymbol{\beta}}}$s improves, as indicated by the increasing excess risks approaching 1. Additionally, $C_{\hat{\boldsymbol{\beta}}}$s also demonstrate performance close to $C^*_{\hat{\boldsymbol{\beta}}}$s, which have the best performance among the tested classifiers. It is worth mentioning that our classifiers outperform $C_{\boldsymbol{\beta}^{(0)}}$s trained by the subsample in all cases. The two classifiers of methods ignoring label noise during the estimation perform the worst, and specifically, as more data are resampled, the efficiency of classifiers $C_{\hat{\boldsymbol{\beta}}_{\mathrm{CD}}}$s improves but still remains evidently inferior to our classifiers.

Table 3.30: AMRSEs and MMRSEs (in brackets) of estimates of $\boldsymbol{\beta}$ from different methods handling mislabelled data using various approaches

m	300	400	500
$\hat{\boldsymbol{\beta}}$	9.1570(6.4227)	5.5180(4.4779)	4.2740(3.5790)
$\boldsymbol{\beta}^{(0)}$	14.6730(7.0017)	7.3450(5.4070)	7.1490(4.0363)
$\hat{\boldsymbol{\beta}}^{*}$	2.1560(1.5767)	2.1560(1.5767)	2.1560(1.5767)
$\hat{\boldsymbol{\beta}}_{\mathrm{R}}$	67.5270(66.6715)	67.5270(66.6715)	67.5270(66.6715)
$\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$	56.0620(64.1352)	53.4140(63.3681)	53.6050(63.7611)

* All values recorded in this table multiplied by 10^{-3} are true values.

Table 3.31: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$

	m	300	400	500
non0 coeff.	$\hat{\boldsymbol{\beta}}$	0.08(0.892500)	0(0.917500)	0(0.945000)
	$\boldsymbol{\beta}^{(0)}$	0.05(0.797500)	0.01(0.877500)	0(0.872500)
	$\hat{\boldsymbol{\beta}}^{*}$	0(0.950000)	0(0.950000)	0(0.950000)
	$\hat{\boldsymbol{\beta}}_{\mathrm{R}}$	2.69(0)	2.69(0)	2.69(0)
	$\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$	0.76(0)	0.52(0)	0.47(0)
0 coeff.	$\hat{\boldsymbol{\beta}}$	2.19(0.889091)	4.36(0.954545)	4.68(0.951818)
	$\boldsymbol{\beta}^{(0)}$	5.84(0.877273)	6.13(0.875455)	6.05(0.868182)
	$\hat{\boldsymbol{\beta}}^{*}$	5.83(0.842727)	5.83(0.842727)	5.83(0.842727)
	$\hat{\boldsymbol{\beta}}_{\mathrm{R}}$	3.85(0.6500)	3.85(0.6500)	3.85(0.6500)
	$\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$	4.12(0.778182)	3.57(0.837273)	2.15(0.901818)

* $\boldsymbol{\beta}_{15\times 1}$ cosists of 4 non-zero coefficients.

Table 3.32: Excess risks of various classifiers

m	300	400	500
$\mathrm{ER}(\mathrm{C}_{\hat{\boldsymbol{\beta}}})$	0.953616	0.965114	0.974780
$\mathrm{ER}(\mathrm{C}_{\boldsymbol{\beta}^{(0)}})$	0.938207	0.957802	0.965803
$\mathrm{ER}(\mathrm{C}_{\hat{\boldsymbol{\beta}}^{*}})$	0.983068	0.983068	0.983068
$\mathrm{ER}(\mathrm{C}_{\hat{\boldsymbol{\beta}}_{\mathrm{R}}})$	0.547227	0.547227	0.547227
$\mathrm{ER}(\mathrm{C}_{\hat{\boldsymbol{\beta}}_{\mathrm{CD}}})$	0.776133	0.854796	0.874128

* The misclassification rate of the Bayes classifiers is 27.056%.

In summary, the simulated examples in this section highlight an important finding:

LR classifiers trained on corrupted datasets without accounting for label noise perform no better than random guessing. In contrast, our proposed two-step estimation method, which uses resampling and estimates the unknown flipping probabilities, greatly improves the performance of LR classifiers in the presence of mislabelling. The approach that corrects only some of the mislabels without considering label noise provides limited improvement to the LR classifier, and the performance can worsen as the mislabels accumulate, making it less practical for real problems. In contrast, as more data is collected, even if it is incorrectly labelled, our proposed method can improve the performance of the method that uses small-scale, perfectly labelled data by leveraging the additional information from the imperfect labels. The results demonstrate that our method can achieve comparable efficiency to the method that corrects all training data.

3.8 The mislabelling probabilities are estimated from the mislabelling ratios obtained from the resampled dataset

From the preceding sections, it has been illustrated through various simulated examples that mislabels have a profound impact on the results of logistic regression. Trained using the raw collected dataset, the LR classifier performs no better than a random guess. Likewise, rectifying part of the mislabels without considering the label noise during estimation results in limited improvements on LR classifiers. All the results have consistently demonstrated that our proposed method, which involves resampling the corrupted dataset and estimating the parameters associated with label flipping probabilities, can yield significant enhancements for the classifiers. Our method is designed to handle a general scenario where the label noise depends on both the class and the features of the data.

In this section, we introduce another alternative estimation method. As described in Section 3.3.1, the process of resampling the dataset allows us to obtain the number of mislabelled instances and calculate the ratios of incorrect labels to the size of the resampled dataset. In the method introduced in this section, the flipping probabilities are estimated using these ratios and then incorporated into the estimation of β. It is worth mentioning that in this approach, the estimation of flipping probabilities is simplified to be class-dependent. This means that for any observation X, the

flipping probabilities are treated as constants denoted as $\hat{M}_{y|0}$ and $\hat{M}_{y|1}$, as described in equations (3.13) and (3.14). The ensuing simulation study is conducted to compare the method with our proposed method estimating three unknown parameters in Section 3.1.2.

3.8.1 Methodology

In this section, we describe the approach that estimates the flipping probabilities using the mislabelled information from the resampled dataset. The estimate of this method is denoted as $\hat{\boldsymbol{\beta}}_{|\hat{M}_{y|0},\hat{M}_{y|1}}$.

After resampling the dataset and calculating the mislabelling ratios $\hat{M}_{y|0}$ and $\hat{M}_{y|1}$, we first follow the first-step estimation process described in Section 3.1 and obtain an estimate $\boldsymbol{\beta}^{(0)}$ from the resampled dataset. Next, we utilise LQA to find the maximiser denoted as $\hat{\boldsymbol{\beta}}_{|\hat{M}_{y|0},\hat{M}_{y|1}}$ of the penalised log-likelihood function (3.25).

$$\begin{aligned} L(\boldsymbol{\beta}) \;=& \sum_{i\in D^c} \Big\{ y_i \operatorname{logit}(\hat{M}_{y|0} + \{\hat{M}_{y|1} - \hat{M}_{y|0}\}\pi_z(X_i)) + \\ & \qquad \log\big(1 - (\hat{M}_{y|0} + \{\hat{M}_{y|1} - \hat{M}_{y|0}\}\pi_z(X_i))\big) \Big\} \\ & + \sum_{i\in D} \big\{ z_i(X_i\boldsymbol{\beta}) + \log(1-\pi_z(X_i)) \big\} - \mathbf{P}_{\boldsymbol{\lambda}_\beta}(\|\boldsymbol{\beta}\|_1). \end{aligned} \tag{3.25}$$

3.8.2 Simulation study

Two simulation examples are conducted in this section to compare the effectiveness of our method, as presented in Section 3.1, with the method of estimating flipping probabilities using the mislabelling ratios in the resampled dataset. We express the results of our method as $\hat{\boldsymbol{\beta}}$, and those from the latter method, as introduced in Section 3.8.1, as $\hat{\boldsymbol{\beta}}_{|\hat{M}_{y|0},\hat{M}_{y|1}}$.

More specifically, the first example scrutinises the mislabelling dataset, comprising the i.i.d. data with covariates from the same distribution. The second example investigates the performance of the two methods in dealing with different correlated covariates within the same training dataset. For both examples, as explained in Section 3.3, we initially generate 100 datasets, each consisting of $N = 15000$ samples. Subsequently, we perform resampling on each training dataset, with a sample size of m. Referring to equations (3.10)-(3.12) and equations (3.13)-(3.15), we calculate the mislabelling ratios for each full-size generated dataset and resampled dataset. In

the simulation, we compute and present the average mislabelling ratios over the 100 experiments, denoted as $\bar{M}$, $\bar{M}_{y|0}$, and $\bar{M}_{y|1}$. Similarly, we calculate and present the averaged mislabelling ratios for the resampled datasets and denote them as $\bar{\hat{M}}$, $\bar{\hat{M}}_{y|0}$, and $\bar{\hat{M}}_{y|1}$.

Example 1: In this example, we establish the parameter settings for model 3.1 as follows

$$
\begin{aligned}
&\boldsymbol{\beta} = (0, \beta_2, \beta_3, 0, \beta_5, \cdots, 0)^T_{p\times 1}, && \beta_2 = 1.8, \beta_3 = -1.15, \beta_5 = 1.8;\\
&\boldsymbol{\eta_0} = (0, \eta_{0,[2:8]}, 0)^T_{p\times 1}, && \eta_{0,j_0'} = 0.55, j_0' = 2,4,6,8, \eta_{0,j_0''} = -0.95, j_0'' = 3,5,7;\\
&\boldsymbol{\eta_1} = (0, \eta_{1,[2:6]}, 0)^T_{p\times 1}, && \eta_{1,j_1'} = -0.85, j_1' = 2,4,6, \eta_{1,j_1''} = 0.6, j_1'' = 3,5;
\end{aligned}
$$

where $p = 10$. We generate the covariates $x_{j_1 i}$ and $x_{j_2 i}$, $i = 1, \cdots, 15000$, with the correlation determined by $0.5^{|j_1 - j_2|}$, for $j_{1,2} = 2, \cdots, p$.

In this example, we resample the same data with a size of $m = 300$ in each trail and consider training data with sizes of $n = 1000, 2000, 3000, 5000,$ respectively. Tables 3.33-3.36 present the outcomes of **Example 1**.

Table 3.33 presents the average mislabelling ratios obtained from 100 full-size generated datasets, each consisting of $N = 15000$ samples, as well as the average mislabelling ratios from resampled datasets with a sample size of $m = 300$. These results provide an overall perspective on the mislabelling data in this example. It is noteworthy that the average mislabelling ratios of the resampled datasets closely match those of the generated datasets.

Table 3.34 displays the performance of estimates from two methods with values for AMRSEs and MMRSEs. The results obtained from our proposed method align with the findings presented in Section 3.3.2. These results indicate that when the resampled data is kept constant and additional unresampled data with incorrect labels is introduced, the effectiveness of our proposed method decreases. Similarly, the method described in Section 3.1 also exhibits poorer performance in this scenario. However, it is evident that our proposed method, which considers the estimation of the parameters $\boldsymbol{\eta_0}$ and $\boldsymbol{\eta_1}$ associated with flipping probabilities, yields more precise estimates than the method utilising the mislabelling ratios from the resampled datasets for estimating flipping probabilities. Particularly when the training dataset increases, and more unresampled data is involved, $\hat{\boldsymbol{\beta}}_{|\hat{M}_{y|0}, \hat{M}_{y|1}}$s exhibit a notable

decline in accuracy compared to $\hat{\boldsymbol{\beta}}$s.

The ESEs of estimates for non-zero coefficients of $\boldsymbol{\beta}$ from both methods and true values (SDs), are presented in Table B.5 and are available for viewing in Appendix B. Table 3.35 demonstrates the performance of variable selections for both methods under different settings for varying training datasets. It includes the incorrect numbers of zero and non-zero estimates for $\boldsymbol{\beta}$, CPs for non-zero and zero coefficients of $\boldsymbol{\beta}$ respectively. From the table, it can be discerned that as n increases, there are more incorrect zero estimates of $\hat{\boldsymbol{\beta}}_{|\hat{M}_{y|0},\hat{M}_{y|1}}$s. In contrast, $\hat{\boldsymbol{\beta}}$s have fewer incorrect zero estimators, and the CPs for non-zero coefficients are higher than those from the method generating $\hat{\boldsymbol{\beta}}_{|\hat{M}_{y|0},\hat{M}_{y|1}}$s. Both methods using SCAD can filter out most of the insignificant predictors in the example, which displays a similar number of incorrect non-zero estimators and close values of CPs for zero coefficients. Our proposed method, considering the estimations for $\boldsymbol{\eta_0}$ and $\boldsymbol{\eta_1}$, is less likely to overlook important features and also produce more parsimonious models than the method employing the mislabelling ratios of resampled datasets as the flipping probabilities during estimations.

Table 3.36 illustrates the performance of the classifiers from the two testing methods compared with the Bayes classifier for 500 testing data and the computation time for each experiment with 100 independent repetitions. As anticipated, the method that does not consider estimating the unknown parameters associated with flipping probabilities requires less computation time than our proposed method when studying the same datasets. However, the decrease in excess risks of the classifiers $C_{\hat{\boldsymbol{\beta}}_{|\hat{M}_{y|0},\hat{M}_{y|1}}}$s indicates the limited consistency of the method with increasing unresampled datasets. Conversely, our classifiers perform closely with the Bayes classifier, having excess risks near 1 and maintaining consistency across different cases.

Table 3.33: Averaged mislabelling ratios of full-size generated and resampled datasets.

$\bar{M}_{y\|0}$	0.521920	$\bar{\hat{M}}_{y\|0}$	0.521184
$\bar{M}_{y\|1}$	0.577837	$\bar{\hat{M}}_{y\|1}$	0.574297
$\bar{M}$	0.549872	$\bar{\hat{M}}$	0.547933

* This table displays the averaged mislabelling ratios for the generated datasets with a size of $N = 15000$ and the resampled datasets with a fixed size of $m = 300$. The ratios remain consistent across the training datasets, which vary in size, with $n = 1000, 2000, 3000, 5000$, as they have the same resampled datasets.

Table 3.34: AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\beta}}$s and $\hat{\boldsymbol{\beta}}_{|\hat{M}_{y|0},\hat{M}_{y|1}}$s

n	1000	2000	3000	5000
$\hat{\boldsymbol{\beta}}$	5.716(2.333)	6.858(2.699)	6.219(3.967)	7.409(4.494)
$\hat{\boldsymbol{\beta}}_{\|\hat{M}_{y\|0},\hat{M}_{y\|1}}$	6.762(2.471)	7.213(2.648)	11.171(4.882)	22.483(7.891)

* All values recorded in this table multiplied by 10^{-3} are true values.

Table 3.35: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$

	n	1000	2000	3000	5000
non0 coeff.	$\hat{\boldsymbol{\beta}}$	0.03(0.900000)	0.03(0.883333)	0.01(0.873333)	0.01(0.823333)
	$\hat{\boldsymbol{\beta}}_{\|\hat{M}_{y\|0},\hat{M}_{y\|1}}$	0.03(0.896667)	0.03(0.863333)	0.13(0.810000)	0.45(0.646667)
0 coeff.	$\hat{\boldsymbol{\beta}}$	0.43(0.948571)	0.23(0.971429)	1.37(0.928571)	1.10(0.947143)
	$\hat{\boldsymbol{\beta}}_{\|\hat{M}_{y\|0},\hat{M}_{y\|1}}$	0.54(0.924286)	0.28(0.964286)	1.14(0.932857)	0.80(0.950000)

* $\boldsymbol{\beta}_{10\times 1}$ has 3 non-zero coefficients.

Table 3.36: Excess risks of classifiers $C_{\hat{\boldsymbol{\beta}}}$s and $C_{\hat{\boldsymbol{\beta}}_{|\hat{M}_{y|0},\hat{M}_{y|1}}}$s, and computing time (in brackets)

n	1000	2000	3000	5000
$ER(C_{\hat{\boldsymbol{\beta}}})$	0.965396(00:28:48)	0.969185 (00:38:02)	0.965396(01:16:38)	0.961917(02:08:50)
$ER(C_{\hat{\boldsymbol{\beta}}_{\|\hat{M}_{y\|0},\hat{M}_{y\|1}}})$	0.964077(00:06:28)	0.959953(00:10:50)	0.899290(00:16:36)	0.782878(00:25:56)

* The misclassification rate of the Bayes classifiers is 19.752% in all cases.

Example 2: In this example, we evaluate the efficiency of two methods using the same training dataset which has different correlated covariates. The parameters of

the log-likelihood function (3.1) are set as

$$
\begin{aligned}
&\boldsymbol{\beta} = (0, \beta_2, \beta_3, 0, \beta_5, \cdots, 0)^{\mathsf{T}}_{p\times 1},\\
&\beta_2 = 1.8, \beta_3 = -1.15, \beta_5 = 1.8;\\
&\boldsymbol{\eta_0} = (0, \eta_{0,2}, \eta_{0,3}, 0, \eta_{0,5}, \cdots, \eta_{0,8}, \eta_{0,9}, \cdots, 0)^{\mathsf{T}}_{p\times 1},\\
&\eta_{0,j_0'} = 0.45, j_0' = 2, 5, 9, \eta_{0,j_0''} = -1, j_0'' = 3, 8;\\
&\boldsymbol{\eta_1} = (0, 0, \eta_{1,3}, \eta_{1,4}, 0, \eta_6, \eta_7, \cdots, 0)^{\mathsf{T}}_{p\times 1},\\
&\eta_{1,j_1'} = -0.5, j_1' = 3, 6, \eta_{1,j_1''} = 0.8, j_1'' = 4, 7.
\end{aligned}
$$

where $p = 10$.

To begin, we elucidate the data generation process for **Example 2**. We generate the data in $b = 30$ sequential batches, each containing 500 i.i.d. data with an identical correlation parameter denoted as ρ_l, where $l = 1, \cdots, b$. Specifically, for the data in batch B_l, $l = 1, \cdots, b$, the correlation between $x_{j_1 i}$ and $x_{j_2 i}$, for $i = 1, \cdots, N$, is $\rho_l^{|j_1 - j_2|}$, with $j_{1,2} = 2, \cdots, p$, and we set

$$
\rho_l = \begin{cases} 0 & \text{if} \quad l+3 \pmod 3 \equiv 0, \\ 0.15 & \text{if} \quad l+3 \pmod 3 \equiv 1, \\ 0.5 & \text{otherwise.} \end{cases}
$$

From these $N = 15000$ generated data, $n = 3000$ samples are chosen as training datasets. In this illustration, we consider different sizes of resampled data, where $m = 300, 600, 900$. To ensure the training datasets contain diverse correlated covariates, we select the first m data for resampling and the subsequent $n - m$ data from the generated datasets. We designate the data from the final batch B_{30} as the out-of-sample testing data. The numerical results for **Example 2** are presented in Table 3.37-Table 3.40.

Table 3.37 records the averaged mislabelling ratios from 100 generated datasets of size $N = 15000$ and those from the resampled datasets of different sizes. Similar to Example 1, the overall ratios obtained from the resampled datasets have close values to those obtained from the generated datasets.

Table 3.38 reports the AMRSEs and MMRSEs of $\hat{\boldsymbol{\beta}}$s and $\hat{\boldsymbol{\beta}}_{|\hat{M}_{y|0},\hat{M}_{y|1}}$s. As evidenced in the table, $\hat{\boldsymbol{\beta}}$s are more precise, particularly when the resampled data size is small ($m = 300$), as indicated by the smaller AMRSE and MMRSE. Even though the performance discrepancy reduces when more data are resampled, our proposed method still outperforms the method that estimates the flipping probabilities of labels by the mislabelling ratios of the resampled dataset.

Table B.6 showcases the robust performance of the sandwich formula (3.5), with the details provided in Appendix B. The variable selection performance in Table 3.39 reveals the number of incorrect non-zero and zero estimates and CPs for non-zero and zero coefficients of $\boldsymbol{\beta}$. This indicates that both methods utilising SCAD can identify nearly all significant features in the simulation, and they can filter out most non-essential features, thus aiding in generating interpretable models. Specifically, we observe that the CPs for the coefficients derived from our method exceed those from the method using mislabelling ratios to estimate flipping probabilities during estimation across various cases.

Table 3.40 highlights the excess risks of both classifiers. As the amount of resampled data grows, both classifiers draw closer to the performance of the Bayes classifier, as the excess risks of each converge to 1. When examining the same datasets, our method proves to be more efficient than the classifier attained by the method detailed in Section 3.8.1 in all instances.

Table 3.37: Average mislabelling ratios of full-size generated and resampled datasets

$\bar{\mathrm{M}}_{\mathrm{y}\|0}$	$\mathfrak{m}$	$\bar{\hat{\mathrm{M}}}_{\mathrm{y}\|0}$
	300	0.477984
0.405305	600	0.398556
	900	0.411664
$\bar{\mathrm{M}}_{\mathrm{y}\|1}$	$\mathfrak{m}$	$\bar{\hat{\mathrm{M}}}_{\mathrm{y}\|1}$
	300	0.482311
0.477984	600	0.481634
	900	0.477210
$\bar{\mathrm{M}}$	$\mathfrak{m}$	$\bar{\hat{\mathrm{M}}}$
	300	0.438000
0.441488	600	0.440167
	900	0.444544

* This table presents the averaged mislabelling ratios for the generated datasets with a size of $\mathsf{N} = 15000$ and the resampled datasets with varying sizes of $\mathfrak{m} = 300, 600, 900$. The training datasets have a fixed size of $\mathfrak{n} = 3000$.

Table 3.38: AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\beta}}$s and $\hat{\boldsymbol{\beta}}_{|\hat{\mathrm{M}}_{\mathrm{y}|0},\hat{\mathrm{M}}_{\mathrm{y}|1}}$s

$\mathfrak{m}$	300	600	900
$\hat{\boldsymbol{\beta}}$	8.021(1.960)	5.179(0.846)	2.108(0.524)
$\hat{\boldsymbol{\beta}}_{\|\hat{\mathrm{M}}_{\mathrm{y}\|0},\hat{\mathrm{M}}_{\mathrm{y}\|1}}$	9.914(4.134)	5.597(1.181)	3.019(0.719)

* All values recorded in this table multiplied by 10^{-3} are true values.

Table 3.39: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$

	$\mathfrak{m}$	300	600	900
non0 coeff.	$\hat{\boldsymbol{\beta}}$	0.05(0.856667)	0.04(0.913333)	0.03(0.923333)
	$\hat{\boldsymbol{\beta}}_{\|\hat{\mathrm{M}}_{\mathrm{y}\|0},\hat{\mathrm{M}}_{\mathrm{y}\|1}}$	0.01(0.726667)	0.01(0.863333)	0.03(0.890000)
0 coeff.	$\hat{\boldsymbol{\beta}}$	0.11(0.987143)	0.08(0.988571)	0.18(0.974286)
	$\hat{\boldsymbol{\beta}}_{\|\hat{\mathrm{M}}_{\mathrm{y}\|0},\hat{\mathrm{M}}_{\mathrm{y}\|1}}$	0.13(0.984286)	0.13(0.981429)	0.32(0.954286)

* $\boldsymbol{\beta}_{10\times 1}$ has 3 non-zero coefficeints.

Table 3.40: Excess risks of $C_{\hat{\beta}}$s and $C_{\hat{\beta}_{|\hat{M}_{y|0},\hat{M}_{y|1}}}$s

m	300	600	900
$ER(C_{\hat{\beta}})$	0.963642	0.976435	0.983179
$ER(C_{\hat{\beta}_{|\hat{M}_{y|0},\hat{M}_{y|1}}})$	0.943856	0.968294	0.977084

* The misclassification rate of the Bayes classifiers is 17.652% across different cases.

In summary, the two simulation examples make it evident that the method, which estimates probabilities based on observed data grouped into $y = 1$ using mislabelling ratios from the resampled datasets, is not as effective as our proposed method. This observation holds even though there's a noticeable similarity between the mislabelling ratios of the resampled datasets and the complete generated datasets. The former cannot maintain consistent performance, particularly when the corrupted datasets grow while more incorrect labels involved. In contrast, our proposed two-step estimation method performs well and remains stable under various circumstances. Although the method detailed in Section 3.8.1 requires less computational time, the precision of the estimations and the efficiency of the classifiers are not as high as ours. Furthermore, the credibility of the method is compromised as it assumes that flipping probabilities only depend on the class. However, in many real-world scenarios, label noise can be related to both the features and the class, which is not accounted for in this method.

3.9 Estimation with the Independence Screening (IS) method

In the preceding sections, we have discussed methods applied to low-dimensional mislabelled data and compared the performance of our proposed method with other alternative methods. Through various examples, the numerical results demonstrate that our method, when combined with SCAD, consistently performs well. Our classifier also shows comparable performance with that of the method using all perfectly labelled training data and has close performance with the Bayes classifier.

Building upon the previous discussions on low-dimensional corrupted data, this section delves deeper into sparser models, specifically high-dimensional datasets affected by mislabelling. We denote the index set of non-zero coefficients of an unknown p-dimensional parameter vector $\mathbf{a}$ as $\mathcal{M}^*$. Here, $\mathcal{M}^*$ represents the index

set of non-zero coefficients, defined as $\mathcal{M}^* = \{1 \leqslant j \leqslant p : a_j \neq 0\}$. The size of this set is denoted as $s = |\mathcal{M}^*|$. The vector $\boldsymbol{a}$ can be any $p \times 1$ vector and is given by $\boldsymbol{a} = (a_1, \cdots, a_p)^T$. The remaining $p - s$ variables are considered less important and are set to 0. However, these insignificant features can exhibit correlations with the response variable through their association with the s significant predictors, making it challenging to effectively screen them out.

Prior studies conducted by Fan and Lv (2008) and Fan et al. (2011) have provided evidence that traditional variable selection methods, such as LASSO, SCAD, and DS, are less effective when applied to high-dimensional data for both variable selection and parameter estimation. Furthermore, in subsequent simulation studies utilising high-dimensional mislabelled data ($p > n$), we observe that the SCAD-only method fails to converge. Therefore, we incorporate Vanilla Sure Independence Screening, also known as Sure Independence Screening (SIS) into our two-step estimation method. SIS has demonstrated its efficacy in effectively reducing dimensions while simultaneously achieving variable selection and estimation (Fan and Lv, 2008, Fan and Song, 2010). Our primary focus is to evaluate the performance of our two-step estimation method when combined with SIS and SCAD on the corrupted dataset. We refer to the variable selection process as 'Independence Screening' (IS).

In the following sections, we begin by explaining how the screening method is incorporated into our two-step estimation approach. We then proceed to present two simulation examples. The first example compares the performance of the SCAD-only method with the method utilising both the Independence Screening (IS) method and regularisation method for low-dimensional mislabelled data. The second example applies the IS-SCAD method to study corrupted datasets that contain a larger number of covariates than the training sample size.

3.9.1 Methodology

We present the two-step estimation method incorporating IS for mislabelled data in this section, focusing on the same model (3.1).

Similar to the method described in Section 3.1, we continue to employ resampling on the corrupted datasets for the approach presented in this section. All the notations and annotations for the samples remain the same as described previously. This

section also follows a two-step estimation method. In this section, we maintain the assumption of an intercept term in the model. The details are as follows:

Step 1: Estimation with IS using resampled data only. In the first step of estimation, the analysis is performed exclusively on the data within the subsample, denoted as $\{(X_i, z_i, y_i)\}$, where $i \in \mathcal{D}$. As explained in Section 3.1.2, the estimation of the three parameters is independent during the first step estimation.

The goal is to find the maximisers of the componentwise regressions for the penalised log-likelihood functions described in (3.4), which are known as Maximum Marginal Likelihood Estimators (MMLEs), for $\boldsymbol{\beta}$, $\boldsymbol{\eta_0}$, and $\boldsymbol{\eta_1}$, respectively. The MMLEs can be expressed as follows:

$$\begin{aligned}
\hat{\boldsymbol{\beta}}_j^M &= (\hat{\beta}_{j,1}^M, \hat{\beta}_j^M) = \underset{\beta_1,\beta_j}{\arg\max} \sum_{i\in\mathcal{D}} \ell(\beta_1 + \beta_j x_{ij}, z_i), \\
(\hat{\boldsymbol{\eta}}_{\mathbf{0}})_j^M &= ((\hat{\eta}_0)_{j,1}^M, (\hat{\eta}_0)_j^M) = \underset{(\eta_0)_{j,1},(\eta_0)_j}{\arg\max} \sum_{i\in\mathcal{D}_0} \ell((\eta_0)_{j,1} + (\eta_0)_j x_{ij}, y_{i|0}), \\
(\hat{\boldsymbol{\eta}}_{\mathbf{1}})_j^M &= ((\hat{\eta}_1)_{j,1}^M, (\hat{\eta}_1)_j^M) = \underset{(\eta_1)_{j,1},(\eta_1)_j}{\arg\max} \sum_{i\in\mathcal{D}_1} \ell((\eta_1)_{j,1} + (\eta_1)_j x_{ij}, y_{i|1}),
\end{aligned} \tag{3.26}$$

for $j = 2, \cdots, p$, and the log-likelihood function with respect to $\boldsymbol{\beta}$ has following form of

$$\ell(\beta_1 + \beta_j x_{ij}, z_i) = \sum_{i\in D} \left\{ z_i \text{logit}(\pi_z(\mathbf{X}_{i,j}^M)) + \log(1 - \pi_z(\mathbf{X}_{i,j}^M)) \right\}, \tag{3.27}$$

where $\mathbf{X}_{i,j}^M = (1, x_{ij})$ and $\text{logit}(\pi(\mathbf{X}_{i,j}^M)) = \beta_1 + \beta_j x_{ij}$. The log-likelihood functions for the other two parameters follow similar forms, and specific details are omitted here.

Now, we explain the process of selecting significant features and estimating $\boldsymbol{\beta}$. After ranking the absolute values of the $p-1$ MMLEs, denoted as $|\hat{\boldsymbol{\beta}}_j^M|_1$ for $j = 2, \cdots, p$, we obtain the index set of selected variables in $\hat{\mathcal{M}}_m$ and have

$$\hat{\mathcal{M}}_m = \{2 \leqslant j \leqslant p : |\hat{\beta}_j^M| > \sigma_m\},$$

where σ_m is a given positive constant threshold. We then select d_m variables, where $1 \leqslant d_m < p$, based on the highest ranked absolute-value MMLEs. Subsequently, we apply penalised likelihood estimation to the updated dataset, which only includes the selected d_m variables. The estimation process is described in **Step 1** of Section 3.1.2. The resulting MLE of $L(\boldsymbol{\beta})$ in (3.4) is achieved and denoted as $\boldsymbol{\beta}^{(0)}$.

A similar process is followed for the other two parameters, resulting in the MLEs of $\boldsymbol{\eta_0}$ and $\boldsymbol{\eta_1}$, which are denoted as $\hat{\boldsymbol{\eta}}_{\mathbf{0}}^{(0)}$ and $\hat{\boldsymbol{\eta}}_{\mathbf{1}}^{(0)}$, respectively. These estimators are obtained by maximising the log-likelihood functions $L(\boldsymbol{\eta_0})$ and $L(\boldsymbol{\eta_1})$ in (3.4).

Step 2: Estimation with IS using both the resampled and unresampled data. In the second-step estimation, we describe the method that analyses both the resampled and unresampled data together. Similar to **Step 2** in Section 3.1.2, an iterative algorithm is applied. In each iteration, denoted by $k = 1, \cdots$, the following details are carried out.

(1) We apply IS to find d_0, where $1 \leqslant d_0 < p$, top absolute-value MMLEs of log-likelihood function (3.1) with respect to $\boldsymbol{\eta_0}$. The index set of important variables is $\hat{\mathcal{M}}_0 = \{2 \leqslant j \leqslant p : |(\hat{\eta}_0)_j^M| > \sigma_0\}$, σ_0 is a threshold. The penalised Newton-Raphson method is applied to maximise $L(\boldsymbol{\beta}_{d_0}^{(k-1)}, (\boldsymbol{\eta_0})_{d_0}, (\boldsymbol{\eta_1})_{d_0}^{(k-1)})$ with respect to $(\boldsymbol{\eta_0})_{d_0}$, the maximiser is denoted as $(\boldsymbol{\eta_0})_{d_0}^{(k)}$. Then the MLE of $\boldsymbol{\eta_0}$ is then obtained and denoted as $\boldsymbol{\eta}_{\mathbf{0}}^{(k)}$.

(2) We apply IS to find d_1, where $1 \leqslant d_1 < p$, top absolute-value MMLEs of log-likelihood function (3.1) with respect to $\boldsymbol{\eta_1}$. The index set of variables is $\hat{\mathcal{M}}_1 = \{2 \leqslant j \leqslant p : |(\hat{\eta}_1)_j^M| \geqslant \sigma_1\}$, σ_1 is a threshold. The penalised Newton-Raphson method is applied to maximise $L(\boldsymbol{\beta}_{d_1}^{(k-1)}, (\boldsymbol{\eta_0})_{d_1}^{(k)}, (\boldsymbol{\eta_1})_{d_1})$ with respect to $(\boldsymbol{\eta_1})_{d_1}$, the maximiser is denoted as $(\boldsymbol{\eta_1})_{d_1}^{(k)}$. Then the MLE of $\boldsymbol{\eta_1}$ is then obtained and denoted as $\boldsymbol{\eta}_{\mathbf{1}}^{(k)}$.

(3) We apply IS to find d, where $1 \leqslant d < p$, top absolute-value MMLEs of log-likelihood function (3.1) with respect to $\boldsymbol{\beta}$, and the index set of variables is obtained, which is denoted as $\hat{\mathcal{M}} = \{2 \leqslant j \leqslant p : |\hat{\beta}_j^M| \geqslant \sigma\}$, where σ is a threshold. The penalised Newton-Raphson method is used to maximise $L(\boldsymbol{\beta}_d, (\boldsymbol{\eta_0})_d^{(k)}, (\boldsymbol{\eta_1})_d^{(k)})$ with respect to $\boldsymbol{\beta}_d$, the maximiser is denoted as $\boldsymbol{\beta}_d^{(k)}$. Then the MLE of $\boldsymbol{\beta}$ is obtained and denoted as $\boldsymbol{\beta}^{(k)}$.

We continue the above iteration until convergence and take the converged $(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\eta}}_{\mathbf{0}}, \hat{\boldsymbol{\eta}}_{\mathbf{1}})$ as the maximiser of (3.1).

We provide a more detailed explanation of step (1) in the iterations above. As for the estimation processes (2) and (3), they follow similar procedures to (1), and therefore, we do not expand on the details here. To be more specific, in the k-th,

where $k = 1, \cdots$, iteration of estimation (1), we obtain the MMLEs of $\boldsymbol{\eta_0}$, denoted as follows:

$$\begin{aligned}(\hat{\boldsymbol{\eta}}_{\mathbf{0}})_j^M &= ((\hat{\eta}_0)_{j,1}^M, (\hat{\eta}_0)_j^M) \\ &= \underset{(\eta_0)_1,(\eta_0)_j}{\arg\max} \left\{ \sum_{i\in\mathcal{D}_0} l((\eta_0)_1 + (\eta_0)_j x_{ij}, y_{i|0}) + \sum_{i\in\mathcal{D}^c} l((\eta_0)_1 + (\eta_0)_j x_{ij}, y_i) \right\},\end{aligned}$$

for $j = 2, \cdots, p$, and $\ell(\cdot)$ has the same form as (3.26) for $i \in \mathcal{D}_0$, and for $i \in \mathcal{D}^c$ we have the following form of the log-likelihood function,

$$\ell((\eta_0)_1 + (\eta_0)_j x_{ij}, y_i) = \sum_{i\in\mathcal{D}^c} \{y_i \text{logit}(\pi((\mathbf{X}_i)_j^M)) + \log(1 - \pi((\mathbf{X}_i)_j^M))\}. \tag{3.28}$$

Here, $\pi((\mathbf{X}_i)_j^M) = \pi_{y|0}((\mathbf{X}_i)_j^M) + \left[\pi_{y|1}((\mathbf{X}_i)_j^M) - \pi_{y|0}((\mathbf{X}_i)_j^M)\right]\pi_z((\mathbf{X}_i)_j^M)$, where $(\mathbf{X}_i)_j^M = (1, x_{ij})$. Specifically, we have

$$\begin{aligned}&\text{logit}(\pi_z((\mathbf{X}_i)_j^M)) = \beta_1^{(k-1)} + \beta_j^{(k-1)} x_{ij}, \\ &\text{logit}(\pi_{y|0}((\mathbf{X}_i)_j^M)) = (\eta_0)_1 + (\eta_0)_j x_{ij}, \\ &\text{logit}(\pi_{y|1}((\mathbf{X}_i)_j^M)) = (\eta_1)_1^{(k-1)} + (\eta_1)_j^{(k-1)} x_{ij}.\end{aligned}$$

After obtaining the top d_0 absolute-value MMLEs of $\boldsymbol{\eta_0}$ with indices kept in $\hat{\mathcal{M}}_0$, we use the updated dataset to compute the MLE denoted as $\hat{\boldsymbol{\eta}}_{\mathbf{0}}^{(k)}$, for the penalised log-likelihood function (3.1) with respect to $\boldsymbol{\eta_0}$.

In this study, we do not delve deeply into determining the thresholds σ, σ_0, and σ_1 used for variable selection in the IS process. Instead, we predefine certain values for these thresholds in our simulation studies. This leaves room for further exploration in future research. We make a conjecture that our two-step estimation method has sure screening property shown in (2.5). The subsequent simulation study has empirical evidence while the proof will be left as future work.

3.9.2 Simulation study

In this section, we present two simulation examples. The first experiment aims to compare the performance of two-step methods that incorporate SCAD and IS-SCAD, respectively, for low-dimensional sparse data with mislabelling. In the second example, we do not present the performance of the SCAD-only method because it

fails to converge during the first step of estimation for the high-dimensional dataset. In the second example, we shift our focus to evaluating the performance of the two-step method using IS-SCAD. We compare its effectiveness with another method that analyses the same datasets but with all labels corrected.

For both examples, we fix training datasets with a size of $n = 1000$ and resampled datasets with a size of $m = 300$. The shared settings for the parameters to be estimated are as follows:

$$\begin{aligned}
&\boldsymbol{\beta} = (0, \beta_2, \beta_3, \beta_4, \cdots, 0)^T_{p\times 1}, && \beta_2 = -1.3, \beta_3 = 1.8, \beta_5 = -1.3;\\
&\boldsymbol{\eta_0} = (0, \eta_{0,2}, \eta_{0,3}, \cdots, 0)^T_{p\times 1}, && \eta_{0,2} = -1.5, \eta_{0,3} = 1.2;\\
&\boldsymbol{\eta_1} = (0, \eta_{1,2}, \eta_{1,3}, \cdots, 0)^T_{p\times 1}, && \eta_{1,2} = 1.15, \eta_{1,3} = -0.45.
\end{aligned}$$

We generate data having the covariates $x_{j_1 i}$ and $x_{j_2 i}$, $i = 1, \cdots, N$, with correlation defined by $\rho^{|j_1 - j_2|}$, for $j_{1,2} = 2, \cdots, p$, and we consider $\rho = 0, 0.2, 0.5$ for separate experiments in this section.

Example 1: In this example, we compare estimates obtained from the two-step estimation method by applying SCAD alone with those obtained from the method by applying IS-SCAD. As SCAD is employed for both methods, we omit the term "SCAD" when denoting estimates for simplicity. To elaborate, we denote $\hat{\boldsymbol{\beta}}$s from the method using SCAD for variable selection and $\hat{\boldsymbol{\beta}}_{IS}$ from the method using IS-SCAD for variable selection.

We set the size of the feature space to $p = 150$. The sure screening process in the first-step estimation identifies $\lfloor m/(4 \times \log(m)) \rfloor$ variables for estimation process for $\boldsymbol{\beta}$, $\lfloor m_0/(2 \times \log(m_0)) \rfloor$ for $\boldsymbol{\eta_0}$ and $\lfloor m_1/(2 \times \log(m_1)) \rfloor$ for $\boldsymbol{\eta_1}$. In the second step estimation, the sure screening process selects $\lfloor N_1/(4 \times \log(N_1)) \rfloor$, $N_1 = \min(m, n - m)$, variables in the variable selection process. Tables 3.41-3.43 report the numerical results for **Example 1**.

Table 3.41 reveals that when the correlation between the covariates is smaller (when $\rho = 0, 0.2$), the two methods do not significantly differ in the accuracy of their estimates, and $\hat{\boldsymbol{\beta}}$s are slightly more precise than $\hat{\boldsymbol{\beta}}_{IS}$s. The performance of both methods deteriorates when the covariates in the dataset are more correlated (when $\rho = 0.5$). However, the accuracy of the IS-SCAD estimates noticeably lags behind that of the SCAD estimates when the covariates are highly correlated. Consistent with the

findings in Fan and Lv (2008), our simulation study demonstrates the limitations of IS in dealing with insignificant covariates that are highly correlated with significant ones.

Table B.7 presents the ESEs of estimates for non-zero coefficients of $\boldsymbol{\beta}$ from both methods and the true values, SDs, which can also be found in Appendix B.

Table 3.42 reveals the variable selection performance of methods utilising different variable selection approaches. In detail, when the correlation parameter ρ is low ($\rho = 0, 0.2$), both methods using SCAD and IS-SCAD are able to identify all the significant features of $\boldsymbol{\beta}$. This is reflected in the fact that there are either no incorrect zero estimates or only a very small number of them. When $\rho = 0$ or $\rho = 0.2$, the CPs of the non-zero coefficients from the two methods are comparable across different cases. However, when the correlation of covariates becomes large ($\rho = 0.5$), both methods show limitations. In the 100-time repeated experiments, they fail to identify some important features, leading to incorrect zero estimates. In particular, the IS-SCAD method exhibits nearly four times as many incorrect zero estimates compared to the SCAD-only method. In this example, the IS-SCAD method demonstrates better performance in excluding irrelevant features compared to the SCAD-only method. This is evident from the lower numbers of incorrect zero estimates and higher CPs for the zero coefficients in all cases. These results indicate that IS-SCAD has an advantage in generating more interpretable models by effectively identifying and excluding irrelevant features.

Table 3.43 indicates that classifiers derived from SCAD or IS-SCAD have similar efficiency when the correlation between covariates is low ($\rho = 0, 0.2$), and both classifiers perform nearly as well as the Bayes classifier, as evidenced by the high excess risks. However, when the correlation between covariates increases, the efficiency of the classifiers obtained by IS-SCAD diminishes more dramatically than that of the SCAD-only method. Regarding computation time and memory utilisation for both methods, there is no discernible pattern. Except for the correlation parameter $\rho = 0$, IS-SCAD requires considerably more time to converge. The computing time could also be affected by the settings used for tuning parameter searches, as discussed in detail in Section 3.4.

Table 3.41: AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\beta}}$s and $\hat{\boldsymbol{\beta}}_{\text{IS}}$s

$m = 300$, ρ	0	0.2	0.5
$\hat{\boldsymbol{\beta}}$	0.624(0.1737)	0.651(0.1969)	1.174(0.2608)
$\hat{\boldsymbol{\beta}}_{\text{IS}}$	0.690(0.3016)	0.510(0.1875)	3.269(4.3943)

* All values recorded in this table multiplied by 10^{-3} are true values.

Table 3.42: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$

$m = 300$, ρ		0	0.2	0.5
non0 coeff.	$\hat{\boldsymbol{\beta}}$	0(0.880000)	0(0.883333)	0.12(0.816667)
	$\hat{\boldsymbol{\beta}}_{\text{IS}}$	0(0.863333)	0.02(0.880000)	0.5(0.450000)
0 coeff.	$\hat{\boldsymbol{\beta}}$	11.95(0.918844)	14.29(0.902857)	12.26(0.917075)
	$\hat{\boldsymbol{\beta}}_{\text{IS}}$	7.97(0.946531)	8.30(0.943605)	6.31(0.957075)

* $\boldsymbol{\beta}_{150\times 1}$ has 3 non-zero coeffients.

Table 3.43: Excess risks of $C_{\hat{\boldsymbol{\beta}}}$s and $C_{\hat{\boldsymbol{\beta}}_{\text{IS}}}$s, and computing time and memory utlisations (in brackets)

$m = 300$, ρ	0	0.2	0.5
$\text{ER}(C_{\hat{\boldsymbol{\beta}}})$	0.961398(12:49:48,8.46 GB)	0.964249(07:21:00,8.46 GB)	0.921988(06:08:13,8.47 GB)
$\text{ER}(C_{\hat{\boldsymbol{\beta}}_{\text{IS}}})$	0.939227 (25:48:36,8.45 GB)	0.963347(07:10:40,8.45 GB)	0.786987(05:42:32,8.02 GB)

* The misclassification rates of the Bayes classifiers are 18.422% for the case of $\rho = 0$, 20.66% for the case of $\rho = 0.2$, and 25.352% for the case of $\rho = 0.5$.

* We implemented each experiment with 100 independent trials using C++ and ran them on 2 *Intel Xeon 6138 CPUs*. For each experiment, we requested 38 cores.

Example 2: In **Example 2**, we study sparser data compared to **Example 1**, considering $p = 350$ and the resampled datasets with a size of $m = 300$. Due to the high dimensionality of data where $m < p$, SCAD method is not applicable for the first-step estimation. Hence, we utilise the IS-SCAD method to analyse both corrupted datasets and the datasets with all correct labels. The estimates from the corrupted datasets are denoted as $\hat{\boldsymbol{\beta}}_{\text{IS}}$s and those from the correctly labelled datasets are denoted as $\hat{\boldsymbol{\beta}}^*_{\text{IS}}$s. Because in practice, it is unlikely to have a perfectly labelled dataset, $\hat{\boldsymbol{\beta}}^*_{\text{IS}}$ serves as a benchmark for evaluating the performance of $\hat{\boldsymbol{\beta}}_{\text{IS}}$. The numerical simulation results for **Example 2** are presented in Tables 3.44-3.46.

It should be noted that when datasets are corrupted with unknown mislabelling, the method integrating IS-SCAD, as elaborated in Section 3.9.1, follows a two-step estimation process. In the case of analysing datasets with all correct labels, there is only one unknown parameter to be estimated, which is $\boldsymbol{\beta}$ in the model (3.7.1). The estimation process for this case involves a one-step estimation, which is identical to the procedure described in **Step 1** of Section 3.9.1. During the sure screening process, for the former method, in the first-step estimation, $\lfloor m/(4 \times \log(m)) \rfloor$ variables are chosen for estimating $\boldsymbol{\beta}$, $\lfloor m_0/(2 \times \log(m_0)) \rfloor$ for $\boldsymbol{\eta_0}$, and $\lfloor m_1/(2 \times \log(m_1)) \rfloor$ for $\boldsymbol{\eta_1}$. In the second step estimation, the sure screening process chooses $\lfloor N_1/(4 \times \log(N_1)) \rfloor$, $N_1 = min(m, n - m)$, features in the variable selection process. Whereas, for the latter method, $\lfloor n/(10 \times \log(n)) \rfloor$ variables are chosen in the screening process for the estimation process of $\boldsymbol{\beta}$.

The AMRSEs and MMRSEs in Table 3.44 mirror the findings from **Example 1**. Specifically, our IS-SCAD method demonstrates effective performance in analysing corrupted datasets with low covariate correlation. The numerical results obtained are comparable to those obtained from the analysis of perfectly labelled datasets. Nonetheless, the accuracy of the estimates significantly deteriorates when the correlation of covariates increases ($\rho = 0.5$).

Table B.8 illustrates the ESEs' performance from the sandwich formula (3.5) and standard deviations. Detailed findings can be explored in Appendix B.

In Table 3.45, the results reveal the numbers of incorrectly estimated zero and non-zero coefficients and CPs for non-zero and zero coefficients of $\boldsymbol{\beta}$. When integrating IS-SCAD into the algorithms, the method analysing the corrupted datasets exhibits inferiority in finding important features, particularly when the covariates are highly correlated ($\rho = 0.5$). Nevertheless, this method shows a similar performance to the one examining perfectly labelled data when $\rho = 0$ and $\rho = 0.2$. The two-step estimation method studying mislabelled data delivers a more accurate estimation of zero coefficients than the method analysing perfectly labelled datasets, indicated by fewer incorrect estimates for zero coefficients across different scenarios. This advantage is attributable to the two-step estimation algorithm, which involves more variable selection processes and hence a higher likelihood of excluding unimportant features.

Finally, the excess risk in Table 3.46 follows the trend observed in the previous

example. Using the two-step IS-SCAD estimation method to analyse mislabelled datasets, we observe that the classifiers perform more efficiently when trained on data that consists of less correlated covariates. Its efficiency is comparable to the Bayes classifier and the classifiers trained by perfectly labelled datasets. However, efficiency notably deteriorates as the covariates become more correlated.

Table 3.44: AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\beta}}_{\mathrm{IS}}$s and $\hat{\boldsymbol{\beta}}^{*}_{\mathrm{IS}}$s

$m = 300$, ρ	0	0.2	0.5
$\hat{\boldsymbol{\beta}}_{\mathrm{IS}}$	0.282(0.101)	0.377(0.232)	1.415(1.913)
$\hat{\boldsymbol{\beta}}^{*}_{\mathrm{IS}}$	0.146(0.081)	0.227(0.103)	0.291(0.100)

* All values recorded in this table multiplied by 10^{-3} are true values.

Table 3.45: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$

$m = 300$, ρ		0	0.2	0.5
non0 coeff.	$\hat{\boldsymbol{\beta}}_{\mathrm{IS}}$	0(0.870000)	0.03(0.833333)	0.52(0.433333)
	$\hat{\boldsymbol{\beta}}^{*}_{\mathrm{IS}}$	0(0.913333)	0(0.870000)	0(0.866667)
0 coeff.	$\hat{\boldsymbol{\beta}}_{\mathrm{IS}}$	8.56(0.975389)	9.20(0.973718)	7.32(0.978905)
	$\hat{\boldsymbol{\beta}}^{*}_{\mathrm{IS}}$	20.13(0.943631)	20.55(0.942882)	20.28(0.944611)

* $\boldsymbol{\beta}_{350\times 1}$ has 3 non-zero coefficients.

Table 3.46: Excess risks of $C_{\hat{\boldsymbol{\beta}}_{\mathrm{IS}}}$s and $C_{\hat{\boldsymbol{\beta}}^{*}_{\mathrm{IS}}}$s

$m = 300$, ρ	0	0.2	0.5
$ER(C_{\hat{\boldsymbol{\beta}}_{\mathrm{IS}}})$	0.952435	0.921959	0.781360
$ER(C_{\hat{\boldsymbol{\beta}}^{*}_{\mathrm{IS}}})$	0.951157	0.965134	0.961250

* The misclassification rates of the Bayes classifiers are 18.422% for the case of $\rho = 0$, 20.816% for the case of $\rho = 0.2$, and 25.352% for the case of $\rho = 0.5$.

To summarise, the two simulated examples underscore the effectiveness of the two-step estimation methods in conjunction with SCAD and IS-SCAD. These methods have demonstrated their capabilities in various facets, including accurate estimation, variable selection, and generating efficient classifiers for low-dimensional sparse mislabelled datasets.

Our simulation study confirms that the method using SCAD is effective for low-dimensional sparse mislabelled datasets. However, when applied to high-dimensional data, SCAD shows limitations and the algorithm fails to converge. IS offers an effective dimension reduction technique. This approach alleviates the limitations of traditional regularisation methods such as LASSO, SCAD, DS, and so forth, when applied to high-dimensional data. Our study integrates the IS-SCAD approach into a two-step estimation process for mislabelled datasets using resampling methods. This approach yields promising results, evidenced by accurate estimation, performance comparable to the method using IS-SCAD on correctly labelled datasets, and efficiency close to that of the Bayes classifier.

Nevertheless, our method using IS has some limitations when applied to corrupted datasets with highly correlated covariates. In such scenarios, it tends to miss important features during the variable selection process. Although IS-SCAD has the advantage of producing parsimonious models, it is not recommended for datasets with highly correlated covariates. In such cases, the method can overlook important variables that are essential for accurate results and meaningful interpretation. Therefore, the use of IS-SCAD should be critically considered, especially in scenarios with highly correlated variables.

3.10 Estimation with the Iterative Independence Screening (IIS) method

In this section, we explore the application of the IIS method, an extension of IS, within the two-step estimation framework to analyse mislabelled data.

IS, while beneficial in many cases, has been recognised to have limitations in scenarios where important features are correlated with responses but not marginally related to them, or where unimportant features are highly marginally related to some important features, or where the correlation between predictors is high (Fan and Lv, 2008). Furthermore, the simulation study in Section 3.9.2 showcases the limitations of IS in dealing with datasets with highly correlated predictors.

IIS, developed to improve the shortcomings of IS, incorporates joint covariate information and utilises an iterative feature selection process to enhance performance (Fan and Lv, 2008, Fan and Song, 2010, Saldana and Feng, 2018). In this section, we

elucidate our two-step estimation method that employs IIS for mislabelled datasets and investigate the efficacy of IIS. The details of the algorithm are provided in the subsequent section, followed by a simulation study aimed at evaluating the performance of the method and comparing it with other alternative methods.

3.10.1 Methodology

This section presents a penalised two-step estimation method incorporating IIS to study datasets with mislabelling. The data is from the model (3.1) and has the same assumption as explained in the previous section. The index set of non-zero coefficients of a sparse model is denoted as $\mathcal{M}^* = \{1 \leqslant j \leqslant p : a_j \neq 0\}$ with the size of $s = |\mathcal{M}^*|$, where $\boldsymbol{a}$ represents any p-vector and $\boldsymbol{a} = (a_1, \cdots, a_p)^\intercal$. The other $p - s$ variables are less important and assigned values of 0.

As introduced in Section 3.1, after the dataset has been resampled, the true labels of the observations are known and only known in this subsample, which is denoted as $\{(X_i, z_i, y_i)\}$, where $i \in \mathcal{D}$ and $|\mathcal{D}| = m$. Here, z_i represents the perfect label, and y_i corresponds to the observation, which can be incorrect. $\mathcal{D}_{z_0}$ is the set of the is such that $i \in \mathcal{D}$ and $z_i = 0$, and $\mathcal{D}_{z_1} = \mathcal{D} - \mathcal{D}_{z_0}$. The index of the unresampled labels is in the set $\mathcal{D}^c$. We uphold the assumption of an intercept in the model throughout this section.

Step 1: Estimation with IIS using resampled data only.

The first step estimation focuses on analysing the data in the resampled dataset. In this step, the three parameters are estimated independently. We aim to identify the most important features for each parameter: $\boldsymbol{\beta}$ with size d_m (where $1 \leqslant d_m < p$), $\boldsymbol{\eta_0}$ with size d_{m_0} (where $1 \leqslant d_{m_0} < p$), and $\boldsymbol{\eta_1}$ with size d_{m_1} (where $1 \leqslant d_{m_1} < p$).

We first calculate MMLEs of the three parameters in (3.4) using a similar approach to the IS-based estimation method described in Section 3.9.1. The MMLEs have the same forms as shown in (3.26). However, unlike the IS method, IIS employs conditional marginal regressions. We provide the following detailed explanation of the estimation process for $\boldsymbol{\beta}$ in the first-step estimation while omitting the process for $\boldsymbol{\eta_0}$ and $\boldsymbol{\eta_1}$ due to their similarity.

Specifically, for the estimation process of $\boldsymbol{\beta}$, we start by ranking the absolute values of the $(p - 1)$ MMLEs of $\boldsymbol{\beta}$, excluding the intercept. Then, we select the top k_0

variables based on the absolute values of their corresponding MMLEs. Here, k_0 is a predetermined value, and $1 \leqslant k_0 < d_m$. The index set of the selected predictors is denoted as $\hat{\mathcal{M}}_1 = \{2 \leqslant j \leqslant p : |\hat{\beta}_j^M| > \sigma_{k_0}\}$, where σ_{k_0} is a positive constant acting as the threshold. We then compute the conditional MMLEs for the predictors whose index is not in $\hat{\mathcal{M}}_1$. The conditional MMLEs have the following forms

$$\hat{\boldsymbol{\beta}}_j^{CM} = (\hat{\beta}_{j,1}^{CM}, \hat{\beta}_j^{CM}) = \underset{\beta_{j,1}^{CM}, \beta_j^{CM}, \boldsymbol{\beta}_{\hat{\mathcal{M}}_1}}{\arg\max} \sum_{i \in \mathcal{D}} \ell(\beta_1 + \beta_j x_{ij} + \mathbf{X}_{i,\hat{\mathcal{M}}_1}^T \boldsymbol{\beta}_{\hat{\mathcal{M}}_1}, z_i), \tag{3.29}$$

where $j \in \hat{\mathcal{M}}_1^c$, $\mathbf{X}_{i,\hat{\mathcal{M}}_1} = (x_{ij'})_{j' \in \hat{\mathcal{M}}_1}$ and $\boldsymbol{\beta}_{\hat{\mathcal{M}}_1} = (\beta_{j'})_{j' \in \hat{\mathcal{M}}_1}$. In this function, the log-likelihood function follows

$$\begin{aligned} &\ell(\beta_1 + \beta_j x_{ij} + \mathbf{X}_{i,\hat{\mathcal{M}}_1}^T \boldsymbol{\beta}_{\hat{\mathcal{M}}_1}, z_i) \\ &= \sum_{i \in D} \left\{ z_i \text{logit}(\pi_z((\mathbf{X}_i)_j^{CM}, \mathbf{X}_{i,\hat{\mathcal{M}}_1})) + \log(1 - \pi_z((\mathbf{X}_i)_j^{CM}, \mathbf{X}_{i,\hat{\mathcal{M}}_1})) \right\}, \end{aligned} \tag{3.30}$$

where $(\mathbf{X}_i)_j^{CM} = (1, x_{ij})$ and $\text{logit}(\pi_z((\mathbf{X}_i)_j^{CM}, \mathbf{X}_{i,\hat{\mathcal{M}}_1})) = \beta_1 + \beta_j x_{ij} + \mathbf{X}_{i,\hat{\mathcal{M}}_1}^T \boldsymbol{\beta}_{\hat{\mathcal{M}}_1}$. By ranking the absolute values of these conditional MMLEs, we select the top k_1 conditional MMLEs and have a subset index denoted as $\mathcal{A}_1 = \{j \in \hat{\mathcal{M}}_1^c : |\hat{\beta}_j^{CM}| > \sigma_{k_1}\}$, where σ_{k_1} is a constant as a threshold. Applying the penalised likelihood estimation on the set $\mathcal{A}_1 \cup \hat{\mathcal{M}}_1$, we obtain a new set $\hat{\mathcal{M}}_2$ of the significant features. For the predictors in $\hat{\mathcal{M}}_2^c$, we again compute the conditional marginal regression as in equation (3.29) to obtain a subset of the top-ranked k_2 covariates, denoted as $\mathcal{A}_2$. The penalised likelihood estimation is applied on the set $\mathcal{A}_2 \cup \hat{\mathcal{M}}_2$ to find the significant features in a set denoted as $\hat{\mathcal{M}}_3$. The process is repeated until $|\hat{\mathcal{M}}_l| = d$, or until $\hat{\mathcal{M}}_{l-1} = \hat{\mathcal{M}}_l$, l is the time of the iteration and $l = 2, \cdots$. Then the estimate of $\boldsymbol{\beta}$ is achieved, denoted as $\boldsymbol{\beta}^{(0)}$.

Step 2: Estimation of combined data using IIS In the second step of estimation, we analyse the combined data, which includes both the corrected labels and the unchecked labels that can be imperfect. Our goal is to find the MLEs for $\boldsymbol{\beta}$, $\boldsymbol{\eta_0}$, and $\boldsymbol{\eta_1}$ in (3.1).

In the $k-$th, $k = 1, \cdots$, iteration, the following details are carried out.

(1) We utilise IIS in the penalised method to find the maximiser of $L(\boldsymbol{\beta}^{(k-1)}, \boldsymbol{\eta_0}, \boldsymbol{\eta_1}^{(k-1)})$ with respect to $\boldsymbol{\eta_0}$, denoted as $\boldsymbol{\eta_0}^{(k)}$.

(2) We utilise IIS in the penalised method to find the maximiser of $L(\boldsymbol{\beta}^{(k-1)}, \boldsymbol{\eta}_0^{(k)}, \boldsymbol{\eta}_1)$ with respect to $\boldsymbol{\eta}_1$ and denote it as $\boldsymbol{\eta}_1^{(k)}$.

(3) We utilise IIS in the penalised method to find the maximiser of $L(\boldsymbol{\beta}, \boldsymbol{\eta}_0^{(k)}, \boldsymbol{\eta}_1^{(k)})$ with respect to $\boldsymbol{\beta}$ and denote it as $\boldsymbol{\beta}^{(k)}$.

We continue the iteration described above until convergence is achieved, and we view the converged $(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\eta}}_0, \hat{\boldsymbol{\eta}}_1)$ as the maximisers of (3.1).

Here, we provide detailed explanations of the estimation process described in step (1) for the above iterations. However, since the estimation process for $\boldsymbol{\eta}_1$ in step (2) and $\boldsymbol{\beta}$ in step (3) is similar, we omit the details of those processes.

At the k-th iteration, we begin by applying IS to the penalised method, as described in **Step 2** of Section 3.9.1, in order to identify the top k_0 predictors with the highest absolute-value MMLEs for $\boldsymbol{\eta}_0$. The resulting index set $\hat{\mathcal{M}}_1 = \{2 \leqslant j \leqslant p : |(\hat{\eta}_0)^{\mathcal{M}} j| \geqslant \sigma_{k_0}\}$ is obtained, where σ_{k_0} is a predefined constant threshold. We then proceed to compute the conditional marginal regressions for the predictors of $\boldsymbol{\eta}_0$, with the following form

$$\begin{aligned}(\hat{\boldsymbol{\eta}}_0)_j^{CM} &= ((\hat{\eta}_0)_{j,1}^{CM}, (\hat{\eta}_0)_j^{CM}) \\ &= \underset{(\eta_0)_{j,1}^{CM}, (\eta_0)_j^{CM}, (\eta_0)_{\hat{\mathcal{M}}_1}}{\arg\max} \Bigg\{ \sum_{i \in \mathcal{D}_0} \ell((\eta_0)_1 + (\eta_0)_j^{CM} x_{ij} + \mathbf{X}_{i,\hat{\mathcal{M}}_1}^T (\boldsymbol{\eta}_0)_{\hat{\mathcal{M}}_1}, y_{i|0}) \\ &+ \sum_{i \in \mathcal{D}^c} \ell\big((\eta_0)_1 + (\eta_0)_j^{CM} x_{ij} + \mathbf{X}_{i,\hat{\mathcal{M}}_1}^T (\boldsymbol{\eta}_0)_{\hat{\mathcal{M}}_1}, y_i\big) \Bigg\},\end{aligned} \tag{3.31}$$

where $j \in \hat{\mathcal{M}}_1^c$, $\mathbf{X}_{i,\hat{\mathcal{M}}_1} = (x_{ij'})_{j' \in \hat{\mathcal{M}}_1}$ and $(\boldsymbol{\eta}_0)_{\hat{\mathcal{M}}_1} = ((\eta_0)_{j'})_{j' \in \hat{\mathcal{M}}_1}$. When $i \in \mathcal{D}_0$, $\ell(\cdot)$ has the same form as the unpenalised log-likelihood function of $L(\boldsymbol{\eta}_0)$ in (3.4), and when $i \in \mathcal{D}^c$ the log-likelihood function has the following form,

$$\begin{aligned}&\ell\big((\eta_0)_1 + (\eta_0)_j^{CM} x_{ij} + \mathbf{X}_{i,\hat{\mathcal{M}}_1}^T (\boldsymbol{\eta}_0)_{\hat{\mathcal{M}}_1}, y_i\big) \\ &= \sum_{i \in \mathcal{D}^c} \big\{ y_i \operatorname{logit}(\pi((\mathbf{X}_i)_j^{CM}, \mathbf{X}_{i,\hat{\mathcal{M}}_1})) + \log(1 - \pi((\mathbf{X}_i)_j^{CM}, \mathbf{X}_{i,\hat{\mathcal{M}}_1})) \big\},\end{aligned} \tag{3.32}$$

where $(\mathbf{X}_i)_j^{CM} = (1, x_{ij})$ for $j \in \mathcal{M}_1^c$, and

$$\begin{aligned}\pi((\mathbf{X}_i)_j^{CM}, \mathbf{X}_{i,\hat{\mathcal{M}}_1}) = \pi_{y|0}((\mathbf{X}_i)_j^{CM}, \mathbf{X}_{i,\hat{\mathcal{M}}_1}) + \\ \big[\pi_{y|1}((\mathbf{X}_i)_j^{CM}, \mathbf{X}_{i,\hat{\mathcal{M}}_1}) - \pi_{y|0}((\mathbf{X}_i)_j^{CM}, \mathbf{X}_{i,\hat{\mathcal{M}}_1})\big] \pi_z((\mathbf{X}_i)_j^{CM}, \mathbf{X}_{i,\hat{\mathcal{M}}_1}).\end{aligned}$$

Specifically, we have

$$\begin{aligned}
\text{logit}(\pi_z((\mathbf{X}_i)_j^{CM}, \mathbf{X}_{i,\hat{\mathcal{M}}_1}) &= \beta_1^{(k-1)} + \beta_j^{(k-1)} x_{ij} + \mathbf{X}_{i,\hat{\mathcal{M}}_1}^T \boldsymbol{\beta}_{\hat{\mathcal{M}}_1}^{(k-1)}, \\
\text{logit}(\pi_{y|0}((\mathbf{X}_i)_j^{CM}, \mathbf{X}_{i,\hat{\mathcal{M}}_1}) &= (\eta_0)_1 + (\eta_0)_j x_{ij} + \mathbf{X}_{i,\hat{\mathcal{M}}_1}^T (\boldsymbol{\eta_0})_{\hat{\mathcal{M}}_1}, \\
\text{logit}(\pi_{y|1}((\mathbf{X}_i)_j^{CM}, \mathbf{X}_{i,\hat{\mathcal{M}}_1}) &= (\eta_1)_1^{(k-1)} + (\eta_1)_j^{(k-1)} x_{ij} + \mathbf{X}_{i,\hat{\mathcal{M}}_1}^T (\boldsymbol{\eta_1})_{\hat{\mathcal{M}}_1}^{(k-1)}.
\end{aligned}$$

We select the k_1 variables from the index set $\mathcal{M}_1^c$ based on their corresponding highest absolute-value conditional MMLEs, which are calculated using the formula shown in (3.31). We define the index set of the selected predictors as $\mathcal{A}_1 = j \in \mathcal{M}_1^c : |(\hat{\eta}_0)_j^{CM}| > \sigma_{k_1}$, where σ_{k_1} is a positive constant threshold. Next, we apply the Newton-Raphson method to find the maximiser of the following penalised log-likelihood function with respect to $(\boldsymbol{\eta_0})_{\mathcal{A}_1 \cup \hat{\mathcal{M}}_1}$:

$$\begin{aligned}
& L(\boldsymbol{\beta}_{\mathcal{A}_1 \cup \hat{\mathcal{M}}_1}^{(k-1)}, (\boldsymbol{\eta_0})_{\mathcal{A}_1 \cup \hat{\mathcal{M}}_1}, (\boldsymbol{\eta_1})_{\mathcal{A}_1 \cup \hat{\mathcal{M}}_1}^{(k-1)}) \\
= & \sum_{i \in D^c} \Big\{ y_i \text{logit}(\pi(\mathbf{X}_{i,\mathcal{A}_1 \cup \hat{\mathcal{M}}_1})) + \log(1 - \pi(\mathbf{X}_{i,\mathcal{A}_1 \cup \hat{\mathcal{M}}_1})) \Big\} \\
+ & \sum_{i \in D} (1 - z_i) \Big\{ y_i (\mathbf{X}_{i,\mathcal{A}_1 \cup \hat{\mathcal{M}}_1}^T \boldsymbol{\eta_0}) + \log(1 - \pi_{y|0}(\mathbf{X}_{i,\mathcal{A}_1 \cup \hat{\mathcal{M}}_1})) \\
& \qquad + (1 - z_i) \log(1 - \pi_z(\mathbf{X}_{i,\mathcal{A}_1 \cup \hat{\mathcal{M}}_1})) \Big\} \\
+ & \sum_{i \in D} z_i \Big\{ y_i (\mathbf{X}_{i,\mathcal{A}_1 \cup \hat{\mathcal{M}}_1}^T \boldsymbol{\eta}_1) + \log(1 - \pi_{y|1}(\mathbf{X}_{i,\mathcal{A}_1 \cup \hat{\mathcal{M}}_1})) \\
& \qquad + z_i \log(\pi_z(\mathbf{X}_{i,\mathcal{A}_1 \cup \hat{\mathcal{M}}_1})) \Big\} - \mathbf{P}_{\lambda_{\boldsymbol{\eta_0}}}(\|(\boldsymbol{\eta_0})_{\mathcal{A}_1 \cup \hat{\mathcal{M}}_1}\|_1),
\end{aligned}$$

where

$$\pi(\mathbf{X}_{i,\mathcal{A}_1 \cup \hat{\mathcal{M}}_1}) = \pi_{y|0}(\mathbf{X}_{i,\mathcal{A}_1 \cup \hat{\mathcal{M}}_1}) + \big[\pi_{y|1}(\mathbf{X}_{i,\mathcal{A}_1 \cup \hat{\mathcal{M}}_1}) - \pi_{y|0}(\mathbf{X}_{i,\mathcal{A}_1 \cup \hat{\mathcal{M}}_1})\big] \pi_z(\mathbf{X}_{i,\mathcal{A}_1 \cup \hat{\mathcal{M}}_1}),$$

and $\mathbf{P}_\lambda(\cdot)$ is a penalty function with the tuning parameter λ which is updated iteratively. The indices of the non-zero estimates are kept in $\mathcal{M}_2$, and the index set $\mathcal{M}_1$ is updated. This process is repeated until the l-th, where $l = 1, \ldots$, iteration, $|\mathcal{M}_l| = d$, or until $\mathcal{M}_{l-1} = \mathcal{M}_l$. Finally, we obtain the maximiser of the penalised log-likelihood function (3.1) with respect to $\boldsymbol{\eta_0}$ for the k-th, $k = 1, \cdots$, iteration, which is denoted as $\hat{\boldsymbol{\eta}}_{\mathbf{0}}^{(k-1)}$.

3.10.2 Simulation study

In this section, two simulated examples of high-dimensional mislabelled data are presented. The first example compares the performance of estimates obtained by the two-step estimation method utilising IS-SCAD and IIS-SCAD respectively. In the second example, we contrast the performance of the two-step method employing IIS-SCAD to examine the corrupted datasets with the method utilising IIS-SCAD to analyse datasets with entirely corrected labels.

The settings for the two examples of parameters of the model (3.1) are as follows

$$\begin{aligned}
&\boldsymbol{\beta} = (0, \beta_2, \beta_3, \beta_4, \cdots, 0)^{\mathsf{T}}_{p\times 1}, && \beta_2 = -1.3, \beta_3 = 1.8, \beta_5 = -1.3;\\
&\boldsymbol{\eta_0} = (0, \eta_{0,2}, \eta_{0,3}, \cdots, 0)^{\mathsf{T}}_{p\times 1}, && \eta_{0,2} = -1.5, \eta_{0,3} = 1.2;\\
&\boldsymbol{\eta_1} = (0, \eta_{1,2}, \eta_{1,3}, \cdots, 0)^{\mathsf{T}}_{p\times 1}, && \eta_{1,2} = 1.15, \eta_{1,3} = -0.45.
\end{aligned}$$

Example 1: In this example, we compare the performance of the two-step estimation method deploying IS-SCAD and IIS-SCAD respectively. For convenience, we denote the estimates from the two methods as $\hat{\boldsymbol{\beta}}_{\mathrm{IS}}$ and $\hat{\boldsymbol{\beta}}_{\mathrm{IIS}}$ respectively.

Datasets with different dimensions and sizes of resampled datasets are discussed. Specifically, we have datasets with dimension $p = 1010$ and resampled datasets with a size of $m = 400$, as well as datasets with dimension $p = 3000$ and $m = 300$. Considering the correlation parameter of the covariates, we take $\rho = 0, 0.2, 0.5$, and the covariates $x_{j_1 i}$ and $x_{j_2 i}$ for $i = 1, \cdots, N$, and $j_{1,2} = 2, \cdots, p$, possess correlation as $\rho^{|j_1 - j_2|}$. From the $N = 15000$ generated data, we randomly select $n = 1000$ data to build the training dataset.

During the sure screening process, for the first step estimation, we select $\lfloor m/(4 \times \log(m)) \rfloor$ variables for the estimation process of $\boldsymbol{\beta}$, $\lfloor m_0/(2 \times \log(m_0)) \rfloor$ features for $\boldsymbol{\eta_0}$ and $\lfloor m_1/(2 \times \log(m_1)) \rfloor$ features for $\boldsymbol{\eta_1}$. For the second step estimation, we select $\lfloor N_1/(4 \times \log(N_1)) \rfloor$, where $N_1 = \min\{m, n - m\}$, features. The simulation results of **Example 1** are recorded in Table 3.47 - Table 3.49.

The AMRSE and the MMRSE in Table 3.47 indicate that when the correlation between the covariates is small, the estimates obtained by IS-SCAD and IIS-SCAD are very close, as revealed by the close results of AMRSEs and MMRSEs. However, when the correlation parameter is larger, when $\rho = 0.5$, the accuracy of the estimates

obtained by the method using IS-SCAD is greatly lower than that of IIS-SCAD. Especially in the case of increased dimensionality, specifically $p = 3000$ and $\rho = 0.5$ in this example, the estimates $\hat{\boldsymbol{\beta}}_{\text{ISIS}}$s demonstrate noticeably higher accuracy compared to $\hat{\boldsymbol{\beta}}_{\text{SIS}}$s. This reaffirms the conclusion that IS cannot perform optimally for mislabelled datasets with highly correlated covariates, while IIS can effectively address this drawback. The estimates $\hat{\boldsymbol{\beta}}_{\text{IIS}}$ consistently exhibit stable performance and closely approximate the true parameter $\boldsymbol{\beta}$ across different cases with varying covariate correlations.

The ESEs and SDs of the estimates from the two methods are cataloged in Table B.9, located in Appendix B.

The results in Table 3.48 display the variable selection performance of the two methods, recording the numbers of incorrect non-zero and zero estimates for $\boldsymbol{\beta}$ and CPs for non-zero and zero coefficients of $\boldsymbol{\beta}$. Implementing IIS, almost no significant features are missed, except in scenarios where the training datasets have $p = 3000$ covariates and $m = 300$ resampled data, wherein a few significant features are missed in the 100 trails. However, for the two-step estimation method utilising IS-SCAD, the results align with those displayed in Table 3.42 and Table 3.45. That is, when the covariates are less correlated ($\rho = 0, 0.2$), almost all important features can be identified. However, many important features fail to be detected when the correlation between the covariates is higher ($\rho = 0.5$). The CPs from the method employing IS-SCAD are lower than the method using IIS-SCAD for both non-zero and zero coefficients. Additionally, the method incorporating IIS exhibits fewer incorrect non-zero estimates compared to the method employing IS. This also indicates that IIS is a more effective approach for generating parsimonious models, even in the presence of unknown mislabelling in the datasets.

The excess risks in Table 3.49 demonstrate that the efficiency of the classifiers obtained using the two-step method with IS-SCAD and IIS-SCAD is close when $p = 1010$ and $m = 400$ while the covariates are less correlated ($\rho = 0, 0.2$). However, when highly correlated covariates are studied, such as in the case of $p = 1010$ and $m = 400$, or when sparser models are studied, such as in the case of $p = 3000$ and $m = 300$, the classifiers obtained by IS-SCAD demonstrate significantly lower efficiency. The performance of the classifier obtained by IIS-SCAD is more stable and is close to that of the Bayes classifier in all scenarios. In this example, under the same

settings for tuning parameter λ selections, the computational time of the method using IIS-SCAD is less than the method using IS-SCAD across different cases. It is also noticeable that the method incorporating IIS-SCAD requires fewer computing resources than the one incorporating IS-SCAD, as shown by lower memory utilisation across different cases.

Table 3.47: AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\beta}}_{\mathrm{IS}}$s and $\hat{\boldsymbol{\beta}}_{\mathrm{IIS}}$s

	ρ	0	0.2	0.5
$p = 1010$, $m = 400$	$\hat{\boldsymbol{\beta}}_{\mathrm{IS}}$	0.053(0.015)	0.091(0.015)	0.518(0.665)
	$\hat{\boldsymbol{\beta}}_{\mathrm{IIS}}$	0.055(0.018)	0.104(0.018)	0.217(0.041)
$p = 3000$, $m = 300$	$\hat{\boldsymbol{\beta}}_{\mathrm{IS}}$	0.067(0.047)	0.078(0.062)	0.265(0.308)
	$\hat{\boldsymbol{\beta}}_{\mathrm{IIS}}$	0.029(0.007)	0.032(0.008)	0.067(0.014)

* All values recorded in this table multiplied by 10^{-3} are true values.

Table 3.48: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$

	ρ		0	0.2	0.5
$p = 1010$, $m = 400$	non0 coeff.	$\hat{\boldsymbol{\beta}}_{\mathrm{IS}}$	0(0.900000)	0(0.850000)	0.43(0.396667)
		$\hat{\boldsymbol{\beta}}_{\mathrm{IIS}}$	0(0.913333)	0(0.860000)	0(0.740000)
	0 coeff.	$\hat{\boldsymbol{\beta}}_{\mathrm{IS}}$	12.01(0.988083)	12.69(0.987398)	10.65(0.989424)
		$\hat{\boldsymbol{\beta}}_{\mathrm{IIS}}$	10.35(0.989722)	10.91(0.989166)	9.92(0.990149)
$p = 3000$, $m = 300$	non0 coeff.	$\hat{\boldsymbol{\beta}}_{\mathrm{IS}}$	0(0.853333)	0.05(0.823333)	0.87(0.166667)
		$\hat{\boldsymbol{\beta}}_{\mathrm{IIS}}$	0(0.866667)	0(0.860000)	0.04(0.766667)
	0 coeff.	$\hat{\boldsymbol{\beta}}_{\mathrm{IS}}$	9.25(0.997020)	9.87(0.996753)	10.13(0.996690)
		$\hat{\boldsymbol{\beta}}_{\mathrm{IIS}}$	7.82(0.997391)	8.66(0.997110)	7.73(0.997421)

* $\boldsymbol{\beta}_{p\times 1}$ has 3 non-zero coefficients for both cases where $p = 1010$ and $p = 3000$.

Table 3.49: Excess risks of $C_{\hat{\beta}_{IS}}$s and $C_{\hat{\beta}_{IIS}}$s, and computing time and memory utilisations (in brackets)

	ρ	0	0.2	0.5
$p = 1010$,	$ER(C_{\hat{\beta}_{IS}})$	0.972959(09:04:04,8.25 GB)	0.967466(08:06:06,8.00 GB)	0.789438(07:37:57,9.04 GB)
$m = 400$	$ER(C_{\hat{\beta}_{IIS}})$	0.971522(08:10:27,5.54 GB)	0.964686(08:56:39,6.08 GB)	0.931237(06:18:25,6.01 GB)
$p = 3000$,	$ER(C_{\hat{\beta}_{IS}})$	0.888578(40:53:24,10.40 GB)	0.859171(41:11:04,9.13 GB)	0.641953(48:02:25,7.80 GB)
$m = 300$	$ER(C_{\hat{\beta}_{IIS}})$	0.954409(37:43:45,4.39 GB)	0.957410(39:27:56,5.17 GB)	0.914113(40:18:52,5.75 GB)

* The misclassification rates of the Bayes classifiers are 18.422% for the case of $\rho = 0$, 20.816% for the case of $\rho = 0.2$, and 25.352% for the case of $\rho = 0.5$.

* We implemented each experiment with 100 independent trials using C++ and ran them on 2 *Intel Xeon 6138 CPUs*. For each experiment, we requested 38 cores.

Example 2: In this example, we compare the performance of the two methods using IIS-SCAD, one studying mislabelled datasets and another studying datasets having all corrected labels. The estimate from the former method is denoted as $\hat{\boldsymbol{\beta}}_{IIS}$ and the latter one is denoted as $\hat{\boldsymbol{\beta}}^*_{IIS}$. We evaluate the effectiveness of our proposed method by comparing it with a method that assumes all labels are corrected. Since it is often impractical and costly to obtain all labels corrected in real-world problems, we use $\hat{\boldsymbol{\beta}}^*_{IIS}$ as a benchmark in this example.

Datasets with dimensions $p = 350, 1500, 3000$ are considered and the correlation parameter is set to $\rho = 0.5$. The covariates $x_{j_1 i}$ and $x_{j_2 i}$ exhibit correlation as $0.5^{|j_1 - j_2|}$ for $j_{1,2} = 2, \cdots, p$, and $i = 1, \cdots, N$. The training datasets have a fixed size of $n = 1000$ and $m = 300$ for resampled datasets.

For the mislabelled data and considering resampling, we employ the two-step estimation method described in Section 3.10.1. For the perfectly labelled datasets, only $\boldsymbol{\beta}$ in the model (3.7.1) needs to be estimated, and the estimation process is the same as the **Step 1** in the estimation process described in Section 3.10.1. During the screening processes, for the two-step estimation process, $\lfloor m/(4 \times \log(m)) \rfloor$ variables are selected for the estimation process of $\boldsymbol{\beta}$, $\lfloor m_0/(2 \times \log(m_0)) \rfloor$ features for $\boldsymbol{\eta_0}$ and $\lfloor m_1/(2 \times \log(m_1)) \rfloor$ features for $\boldsymbol{\eta_1}$. For the second step estimation, we select $\lfloor N_1/(4 \times \log(N_1)) \rfloor$, where $N_1 = \min\{m, n - m\}$, features. For the method analysing all perfect labels, $\lfloor n/(10 \times \log(n)) \rfloor$ variables are selected for the estimation process across different cases. The simulation results of **Example 2** are recorded in Table 3.50 - Table 3.52.

Table 3.50 shows the AMRSE and MMRSE for the estimates trained by the cor-

rupted datasets with the two-step estimation method and by the perfectly labelled datasets. As expected, with all labels corrected, $\hat{\boldsymbol{\beta}}^*_{\text{IIS}}$s are more accurate than $\hat{\boldsymbol{\beta}}_{\text{IIS}}$s in all cases, but $\hat{\boldsymbol{\beta}}_{\text{IIS}}$s still exhibit stable performance with low AMRSEs and MMRSEs in all cases, which are close to the true values of $\boldsymbol{\beta}$.

The ESEs of the estimates obtained by following the sandwich formula (3.5) and the true value SDs are presented in Table B.10 and in Appendix B, which exhibits how well the sandwich formula performs with estimated values close to the true values.

The performance of the variable selection is displayed in Table 3.51. Both methods incorporating IIS-SCAD can almost identify all the significant features, with very few missed from the two-step estimation method studying mislabelled data. Both methods maintain stable performance with the increase in the dimensions of the datasets, demonstrated by identical CPs and close numbers of incorrect non-zero estimates and CPs for both non-zero and zero coefficients of $\boldsymbol{\beta}$. Intriguingly, our two-step estimation method yields fewer incorrect non-zero estimates than the method studying all perfect labels. This can be explained by the algorithm used in the two-step estimation method, which performs variable selection more times, increasing the likelihood of screening out the insignificant features.

In Table 3.52, the excess risks are recorded. It is evident that classifiers trained by all perfect labels perform closer to the Bayes classifiers, although it is typically challenging to have all data perfectly labelled. Our two-step estimation method using IIS-SCAD exhibits stable performance across different cases. Despite being trained by the corrupted dataset and with less than half of the data resampled, the efficiency of classifiers is relatively commendable compared to the Bayes classifiers, with approximately one-third of the test data being misclassified.

Table 3.50: AMRSEs and MMRSEs (in brackets) of $\hat{\boldsymbol{\beta}}_{\text{IIS}}$s and $\hat{\boldsymbol{\beta}}^*_{\text{IIS}}$s

p	350	1500	3000
$\hat{\boldsymbol{\beta}}_{\text{IIS}}$	0.614(0.366)	0.165(0.091)	0.067(0.014)
$\hat{\boldsymbol{\beta}}^*_{\text{IIS}}$	0.276(0.017)	0.059(0.004)	0.033(0.003)

* All values recorded in this table multiplied by 10^{-3} are true values.

Table 3.51: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$

	p	350	1500	3000
non0 coeff.	$\hat{\boldsymbol{\beta}}_{\text{IIS}}$	0.02(0.776667)	0.02(0.766667)	0.04(0.766667)
	$\hat{\boldsymbol{\beta}}^*_{\text{IIS}}$	0(0.866667)	0(0.876667)	0(0.866667)
non0 coeff.	$\hat{\boldsymbol{\beta}}_{\text{IIS}}$	7.69(0.977867)	7.58(0.994937)	7.73(0.997421)
	$\hat{\boldsymbol{\beta}}^*_{\text{IIS}}$	11.05(0.968156)	11.31(0.992445)	11.49(0.996166)

* $\boldsymbol{\beta}_{\text{p}\times 1}$ has 3 non-zero coefficients in all cases.

Table 3.52: Excess risks of $\text{C}_{\hat{\boldsymbol{\beta}}_{\text{IIS}}}$s and $\text{C}_{\hat{\boldsymbol{\beta}}^*_{\text{IIS}}}$s

p	350	1500	3000
$\text{ER}(\text{C}_{\hat{\boldsymbol{\beta}}_{\text{IIS}}})$	0.906076	0.886124	0.914113
$\text{ER}(\text{C}_{\hat{\boldsymbol{\beta}}^*_{\text{IIS}}})$	0.980963	0.981646	0.973654

* For all cases where $\text{p} = 350,\ 1500,\ 3000$, the Bayes classifiers have the misclassification rate as 25.352% in each experiment.

In the two examples presented in this section, we evaluate the performance of our proposed two-step estimation method using IIS-SCAD. We have compared it with the two-step estimation method using IS-SCAD, as well as the method using IIS-SCAD with all perfectly labelled datasets. The numerical results provide strong evidence in favor of our two-step estimation method using IIS-SCAD. The IIS-SCAD method significantly outperforms the IS-SCAD method in terms of stability across varying corrupted datasets with different covariate correlations. It consistently generates more accurate estimation results, identifies almost all important variables, and constructs more efficient classifiers.

Compared with the method using IIS-SCAD that analyses all perfectly labelled data, the two-step IIS-SCAD method maintained competitive performance even when dealing with corrupted datasets. Specifically, it still exhibits good performance in terms of accurate estimation results, effective variable selection, and the efficiency of classifiers. In real-world scenarios, correcting all labels is often very challenging due to the substantial time and cost associated with the process. Moreover, the existing techniques are not capable of identifying all the noisy labels, as discussed in Section 2.5. In light of these considerations, our proposed method offers a promising solution for handling high-dimensional classification problems with mislabelled data. Our

proposed method offers several advantages, including its ability to maintain good performance despite high-dimensional mislabelled data, its applicability in situations where label correction is not feasible, and it is potential for reduced expenses, time, and resource requirements.

3.11 Real data analysis

In this section, we apply our proposed method as well as the alternative methods introduced in the previous sections to analyse a real dataset. The dataset used in this study is the Framingham Heart Study (FHS) dataset, which provides predictions for coronary heart disease risk over a 10-year period. The dataset is available for free download on the Kaggle website: Framingham Heart Study.

In 1948, FHS was initiated with the aim of advancing researchers' understanding of the epidemiology of coronary heart disease in the United States. For over 70 years, FHS has been the subject of ongoing investigation, with researchers tirelessly exploring its extensive dataset. This long-term study has provided valuable insights into the epidemiology of cardiovascular disease and its related risk factors. A concise historical overview of selected contributions from previous work, which discusses the influence of FHS, can be found in the work of Mahmood et al. (2014).

Given the extensive analysis and the widespread use of the Framingham Heart Study (FHS) dataset by researchers over the years, it is reasonable to presume all perfect labels in this specific case. Logistic regression (LR) has been a common method of analysis for the FHS, as shown in previous studies such as Abbott (1985), Wilson et al. (1987), Ambrish et al. (2022), among others.

In real-world scenarios, acquiring a dataset with completely accurate labels, especially with large sample sizes, is often challenging. The label noise could be due to various factors including manual recording errors, unsuitable data collection instructions, and missing or misleading predictor information, among other reasons. Considering more general cases, label noise is both class- and feature-dependent, which complicates the task of label correction. Moreover, pinpointing mislabelled instances within the dataset can be unfeasible, particularly when the noisy labels behave similarly to the true labels rather than as outliers (Li et al., 2017). In contrast, commissioning experts to verify and correct a subset of the dataset might be a more

feasible and cost-effective approach. However, to the best of our knowledge, there is limited research addressing the challenge of mislabelled datasets in this context.

Motivated by these challenges, we aim to explore the application of our two-step estimation method, incorporating resampling and using LR, to analyse the FHS dataset with mislabelling that originates from both class- and feature-dependent label noise. In this section, we start by presenting an overview of the dataset. We then introduce predetermined parameters associated with the flipping probabilities of labels, which we use to introduce label noise into the FHS dataset. The process of generating mislabelling follows the same methodology discussed in Section 3.3. Subsequently, in the simulation study, we examine various parameter settings that can introduce different levels of incorrect labels into the dataset.

3.11.1 Description of the dataset and how to reclassify the perfect labels with noise

After the exclusion of missing values, we obtain a dataset consisting of 3656 records and 15 attributes, which are considered potential risk factors. These include demographic variables, behavioral variables, and historical as well as current medical treatments. We presume an intercept term to be present in the model. The covariates are denoted by $X_i = (x_{1i}, \cdots, x_{18i})^T$, where $i = 1, \cdots, 3656$, with the intercept defined as $x_{1i} \equiv 1$. A detailed description of these factors is provided in Table 3.53.

The collected response variables are denoted as $z_i = \{0, 1\}$, $i = 1, \cdots, 3656$, with z_is being perfectly labelled. If the expert deems the individual observed to be at risk of coronary heart disease (CHD), the response variable is labelled as 1, otherwise, it is 0. All continuous variables in Table 3.53 are standardised prior to the analysis. We use the logistic regression method to analyse this dataset and assume

$$\text{logit}\{\pi_z(X)\} = \log\frac{\pi_z(X)}{1-\pi_z(X)} = X^T\boldsymbol{\beta},$$

where $\boldsymbol{\beta} = (\beta_1, \beta_2, \cdots, \beta_{18})^T$ is the parameter we are interested in and aim to estimate. For any observation X, we have

$$C(X) = \begin{cases} 1 & \text{if } \pi_z(X) \geqslant 0.5 \\ 0 & \text{otherwise,} \end{cases} \tag{3.33}$$

where $C(\cdot)$ represents any classifier.

In our study, we select 80% of the observed data as training data, amounting to a size of $n = 2925$, while the remaining 731 data points are utilised for out-of-sample testing. Given the parameters $\boldsymbol{\eta_0}$ and $\boldsymbol{\eta_1}$ in the model (3.1), we can introduce mislabelling to the dataset following the algorithm described in Section 3.3. The new dataset is comprised of $\{(X_i, z_i, y_i)\}$, $i = 1, \cdots, n$, where X_is and y_is are observed, y_is may be incorrect, and the perfect labels z_is are treated as unknown. In the subsequent tables, the mislabelling rate obtained from the resampled data is denoted as $\hat{M}$, the mislabelling rate for label 0 as $\hat{M}_{y|0}$, and the mislabelling rate for label 1 as $\hat{M}_{y|1}$. We define the misclassification rate for any classifier C on the testing dataset as

$$\text{MIS}_C = \sum_{t=1}^{731} \frac{\mathbf{1}(C(X_t))}{731},$$

where $t = 1, \ldots, 731$, and $\mathbf{1}$ is the indicator function defined as

$$\mathbf{1}(C(X_t)) = \begin{cases} 1 & \text{if } C(X_t) \neq z_t \\ 0 & \text{otherwise.} \end{cases}$$

Table 3.53: Description of the dataset

Factor Name	Description	Note
Intercept	Equals to 1	x_{1i}
Sex	Nominal variable: 1 for male, 0 for female	x_{2i}
Education Level	Categorical variable: - "Graduate or professional degree": reference group - "Less than or completed high school" (education_1) - "High School Diploma/GED" (education_2) - "College" (education_3)	 x_{3i} x_{4i} x_{5i}
CurrentSmoker	Nominal variable: 1 for current smoker, 0 for non-smoker	x_{6i}
BPMeds	Nominal variable: 1 for anti-hypertensive medication user, 0 for non-user	x_{7i}
PrevalentStroke	Nominal variable: 1 for prevalent stroke, 0 for no stroke	x_{8i}
PrevalentHyp	Nominal variable: 1 for prevalent hypertension, 0 for no hypertension	x_{9i}
Diabetes	Nominal variable: 1 for diabetes, 0 otherwise	x_{10i}
CigsPerDay	Continuous variable: number of cigarettes smoked per day (integer values)	x_{11i}
Age	Continuous variable: age of the subjects (integer values)	x_{12i}
TotChol	Continuous variable: total cholesterol level (measured in *mg/dL*)	x_{13i}
SysBP	Continuous variable: systolic blood pressure (measured in *mmHg*)	x_{14i}
DiaBP	Continuous variable: diastolic blood pressure (measured in *mmHg*)	x_{15i}
BMI	Continuous variable: body mass index (weight in *kg* divided by height in *meters* squared)	x_{16i}
HeartRate	Continuous variable: heart rate (measured in beats per minute)	x_{17i}
Glucose	Continuous variable: blood glucose level (measured in *mg/dL*)	x_{18i}

* This table provides a concise description of the potential risk factors denoted as x_{ji}, where $i = 1, \cdots, 3656$, represents the number of observations in the training dataset, and $j = 1, \cdots, 18$, denotes the index.

3.11.2 Detailed analysis

In this section, we investigate the Framingham Heart Study (FHS) dataset in two scenarios under different mislabelling conditions. To evaluate the performance of our proposed two-step estimation method, as described in Section 3.1, we compare it with several alternative methods in this section. These methods include:

- The method that analyses only the subsampled data, with the corresponding estimates denoted as $\boldsymbol{\beta}^{(0)}$.
- The method discussed in Section 3.7.2, which follows the approach proposed by Cannings et al. (2020) to handle mislabelling using the raw data, and the estimate results are represented as $\hat{\boldsymbol{\beta}}_{\mathrm{R}}$.
- The method described in Section 3.7.3, which focuses on correcting the labels in the resampled dataset without considering the flipping probabilities during estimation, and the estimate is denoted as $\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$.

- The method outlined in Section 3.8.1, which estimates the flipping probabilities using the known mislabelling ratios from the subsample, and the estimates are represented as $\hat{\boldsymbol{\beta}}_{|\hat{M}_{y|0},\hat{M}_{y|1}}$.

- The method described in Section 3.7.1 is discussed, which scrutinises the dataset with all corrected labels. The estimate derived from this method is denoted as $\hat{\boldsymbol{\beta}}^*$ and is used as a benchmark for comparison with other estimates.

In the first scenario, we examine the impact of varying amounts of resampled data on the performance of the tested methods using the same corrupted dataset. The second scenario explores a different setup where the FHS dataset is corrupted using a different set of parameters $\boldsymbol{\eta_0}$ and $\boldsymbol{\eta_1}$, which are related to flipping probabilities. This allows us to evaluate the effectiveness and robustness of these methods under different conditions.

In this section, we also perform the Wald test for each estimate of the coefficient to test the null hypothesis $H_0 : \beta_j = 0$ against the alternative hypothesis $H_1 : \beta_j \neq 0$, where $j = 1, \cdots, 18$. The significance level for the test is set at $\alpha = 0.05$. The resulting p-values are presented in the subsequent content.

Scenario 1: In **Scenario 1**, we set the parameters $\boldsymbol{\eta_0}$ and $\boldsymbol{\eta_1}$ related to the flipping probabilities as

$$
\begin{aligned}
&\boldsymbol{\eta_0} = (0, \cdots, \eta_{0,5}, \cdots, \eta_{0,8}, \eta_{0,9}, \eta_{0,10}, 0, \eta_{0,12}, \eta_{0,13}, \cdots, \eta_{0,16}, \cdots, 0)^T_{p\times 1},\\
&\eta_{0,5} = \eta_{0,8} = \eta_{0,9} = \eta_{0,10} = 1,\ \eta_{0,12} = \eta_{0,13} = -0.55,\ \eta_{0,16} = 0.55;\\
&\boldsymbol{\eta_1} = (0, \cdots, \eta_{1,3}, \eta_{1,4}, 0, \eta_{0,6}, \eta_{0,7}, \cdots, \eta_{0,12}, 0, \eta_{0,14}, \eta_{0,15}, \cdots, 0)^T_{p\times 1},\\
&\eta_{1,3} = \eta_{1,4} = \eta_{0,6} = \eta_{0,7} = 1,\ \eta_{0,12} = -0.8,\ \eta_{0,14} = -0.65,\ \eta_{0,15} = 0.8,
\end{aligned}
$$

where $p = 18$.

We then analyse different percentages of the training data resampled, ranging from 10% to 50% in increments of 10%. The corresponding sizes are $m = 292, 585, 877, 1170, 1462$. It is worth noting that when $m = 0$, the estimate $\hat{\boldsymbol{\beta}}_R$ is derived using the corrupted dataset without any modification, and when $m = n = 3656$, the estimate $\hat{\boldsymbol{\beta}}^*$ can be obtained using the method that corrects all labels in the full training dataset. Tables 3.54-Table 3.60 showcase the results of **Scenario 1**.

Table 3.54 lists the mislabelling ratios obtained from various resampled datasets.

Interestingly, we observe a close correspondence between the mislabelling rates obtained from the resampled dataset and those from the full-size dataset, even when the resampled data is small. This is indicated by the close values observed between the resampled datasets with varying sizes and the dataset resampled with a size of $\mathfrak{m} = \mathfrak{n} = 3656$.

The efficiency of classifiers achieved by different methods is recorded in Table 3.55. It is clear that when directly using the corrupted datasets without any resampling (when $\mathfrak{m} = 0$), the misclassification rates of $\mathsf{C}_{\hat{\boldsymbol{\beta}}_{\mathrm{R}}}$ exceeds 50%, indicating that the LR classifier performs only as good as random guesses or potentially worse. Incorporating a resampling approach, the efficiency of all other classifiers $\mathsf{C}_{\boldsymbol{\beta}^{(0)}}$, $\mathsf{C}_{\hat{\boldsymbol{\beta}}}$, $\mathsf{C}_{\hat{\boldsymbol{\beta}}_{|\hat{\mathrm{M}}_{\mathrm{y}|0},\hat{\mathrm{M}}_{\mathrm{y}|1}}}$ and $\mathsf{C}_{\hat{\boldsymbol{\beta}}_{\mathrm{CD}}}$ improves and the misclassification rates approach that of $\mathsf{C}_{\hat{\boldsymbol{\beta}}^*}$. However, for classifier $\mathsf{C}_{\hat{\boldsymbol{\beta}}_{\mathrm{CD}}}$, when less than 30% of training data is resampled ($\mathfrak{m} = 292$, 585, and 877), the classifier performs poorly, essentially matching random guess, despite improvements as $\mathfrak{m}$ increases. Our classifier $\mathsf{C}_{\hat{\boldsymbol{\beta}}}$ performs best and maintains efficiency closest to $\mathsf{C}_{\hat{\boldsymbol{\beta}}^*}$ across different cases. This result further emphasises the effectiveness of our approach in leveraging the dataset with mislabelling to enhance the performance of classifiers trained on small clean datasets. These findings are consistent with the results of the simulation study discussed in Section 3.7.4. Additionally, $\mathsf{C}_{\hat{\boldsymbol{\beta}}_{|\hat{\mathrm{M}}_{\mathrm{y}|0},\hat{\mathrm{M}}_{\mathrm{y}|1}}}$ has more misclassified labels than ours when studying the same datasets.

Tables 3.56 to 3.60 provide comprehensive results, including estimates for each predictor, corresponding p-values, and the absolute difference between the estimates and $\hat{\boldsymbol{\beta}}^*$. The latter serves as a benchmark for comparing the accuracy of the different estimates. From the results obtained for $\hat{\boldsymbol{\beta}}^*$, it is worth noting that the variables "Sex" (indicating male gender), "cigsPerDay", "age", "totChol", "sysBP", "diaBP", "BMI", "heartRate", and "glucose" are identified as important features. Among these, "totChol", "diaBP", "BMI", and "heartRate" exhibit relatively small numerical values, suggesting that they may have a weaker predictive power for assessing the risk of coronary heart disease.

The estimate $\hat{\boldsymbol{\beta}}_{\mathrm{R}}$, obtained from the method using the raw corrupted data, is not affected by the varying sizes of the resampled data. However, it is evident that this method, which does not take mislabelling into account, produces less accurate estimates, most of which are close to 0. As a result, the resulting model becomes challenging to interpret. The method that corrects only the labels in the resampled

dataset, while ignoring the noise of labels during estimation, having the estimate as $\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$ which performs similarly to $\hat{\boldsymbol{\beta}}_{\mathrm{R}}$, with estimates close to 0 across different cases. The estimate of the method using mislabelling ratios as flipping probabilities also deviates largely from the method using all corrected data and deems most predictors insignificant, shown by $\hat{\boldsymbol{\beta}}_{|\hat{\mathrm{M}}_{\mathrm{y}|0},\hat{\mathrm{M}}_{\mathrm{y}|1}}$s having most zero estimates.

As more data is resampled and more labels are corrected, our method's estimates ($\hat{\boldsymbol{\beta}}$s) become closer to $\hat{\boldsymbol{\beta}}^{*}$s. This can be observed by the closer performance in variable selection and the trend of decreasing mean absolute difference between $\hat{\boldsymbol{\beta}}$ and $\hat{\boldsymbol{\beta}}^{*}$. It is also noteworthy that in all cases, the mean absolute difference between $\hat{\boldsymbol{\beta}}$ and $\hat{\boldsymbol{\beta}}^{*}$ is the smallest among other estimates, further demonstrating the effectiveness of our method.

Table 3.54: mislabelling ratios of the resampled datasets with varying sizes.

m	292	585	877	1170	1462	$3656(\mathrm{m}=\mathrm{n})$
$\hat{\mathrm{M}}_{\mathrm{y}\mid 0}$	0.573222	0.597938	0.604396	0.602434	0.595488	0.600965
$\hat{\mathrm{M}}_{\mathrm{y}\mid 1}$	0.339623	0.370000	0.348993	0.342391	0.348416	0.357631
$\hat{\mathrm{M}}$	0.530822	0.558974	0.561003	0.561538	0.558140	0.564444

* This table presents the mislabelling ratios for different sizes of resampled datasets. Each column in the table represents the mislabelling ratios obtained from resampled data. The percentages indicate the amount of training data that is resampled, ranging from 10% to 50% in increments of 10%. The last column corresponds to the mislabelling ratios when the entire training dataset, where $\mathrm{m}=\mathrm{n}$, is resampled.

Table 3.55: Misclassification rates of various classifiers (in $\%$).

m	0	292	585	877	1170	1462	$3656(\mathrm{m}=\mathrm{n})$
$\mathrm{Mis}_{\mathrm{C}_{\hat{\boldsymbol{\beta}}^{(0)}}}$	62.9275	16.6895	16.1423	16.1423	16.1423	15.8687	15.5951
$\mathrm{Mis}_{\mathrm{C}_{\hat{\boldsymbol{\beta}}}}$	62.9275	16.4159	16.1423	15.8687	15.8687	15.5951	15.5951
$\mathrm{Mis}_{\mathrm{C}_{\hat{\boldsymbol{\beta}}_{\mid\hat{\mathrm{M}}_{\mathrm{y}\mid 0},\hat{\mathrm{M}}_{\mathrm{y}\mid 1}}}}$	62.9275	17.9207	16.2791	16.2791	16.1423	15.7319	15.5951
$\mathrm{Mis}_{\mathrm{C}_{\hat{\boldsymbol{\beta}}_{\mathrm{CD}}}}$	62.9275	66.2107	60.6019	48.974	22.2982	16.1423	15.5951

* The same out-of-sample testing dataset consisting of 731 observations is used for all cases.

* In the table, when $\mathrm{m}=0$, indicating no resampling, all the classifiers are equivalent to $\mathrm{C}_{\hat{\boldsymbol{\beta}}_{\mathrm{R}}}$, while when $\mathrm{m}=\mathrm{n}=3656$, indicating that all labels are correct, all the classifiers behave the same as $\mathrm{C}_{\hat{\boldsymbol{\beta}}^{*}}$. Therefore, the misclassification rates are identical for all classifiers in these scenarios.

Table 3.56: Comparisons of different estimates under the setting of resampled data with a size of $\mathfrak{m} = 292$

	$\hat{\boldsymbol{\beta}}$			$\boldsymbol{\beta}^{(0)}$			$\hat{\boldsymbol{\beta}}_{\mathrm{R}}$		
	Estimates	p values	$\lvert\hat{\beta}_j^* - (\hat{\beta}_j)\rvert$	Estimates	p values	$\lvert\hat{\beta}_j^* - (\beta_j)^{(0)}\rvert$	Estimates	p values	$\lvert\hat{\beta}_j^* - ((\hat{\beta}_j)_{\mathrm{R}}\rvert$
Intercept	**-2.211460**	**0.000014**	0.107180	**-2.087410**	< **0.000001**	0.231230	**0.001739**	< **0.000001**	2.320379
sex	0.368468	0.357504	0.066415	**0.001245**	**0.000001**	0.433638	**1.2035e-05**	< **0.000001**	0.434871
$\mathrm{education}_1$	0.349259	0.342181	0.208465	0	1.000000	0.140794	0	1.000000	0.140794
$\mathrm{education}_2$	0	1.000000	0	**-0.000026**	< **0.000001**	0.000026	0	1.000000	0
$\mathrm{education}_3$	0.278912	0.570060	0.278912	**0.000220**	< **0.000001**	0.000220	**0.000114**	< **0.000001**	0.000114
currentSmoker	-0.534159	0.289591	0.534159	**0.000020**	< **0.000001**	0.000020	**3.20467e-05**	< **0.000001**	0.000032
BPMeds	-1.180150	0.240864	1.180150	-1.322270	0.271214	1.322270	0	1.000000	0
prevalentStroke	1.066060	0.384707	0.340873	-2.925030	0.060387	3.650217	0	1.000000	0.725187
prevalentHyp	**0.646541**	**0.043511**	0.394860	**1.061050**	**0.002709**	0.809369	**0.000307**	< **0.000001**	0.251374
diabetes	0.410849	0.443685	0.410849	0	1.000000	0	0	1.000000	0
cigsPerDay	0.311982	0.145045	0.036637	**0.006057**	**0.017615**	0.269288	0	1.000000	0.275345
age	**0.493298**	**0.002495**	0.008395	**0.475600**	**0.008313**	0.026093	**-0.008695**	< **0.000001**	0.510388
totChol	**0.467694**	**0.003430**	0.464350	**0.419572**	**0.012061**	0.416228	**-0.000688**	< **0.000001**	0.004032
sysBP	**0.000109**	< **0.000001**	0.328768	**0.000034**	**0.001982**	0.328843	0	1.000000	0.328877
diaBP	0	1.000000	0.000021	**0.003061**	**0.001223**	0.003082	**0.000178**	< **0.000001**	0.000198
BMI	0	1.000000	0.000253	**0.000020**	**0.023640**	0.000233	**0.001383**	< **0.000001**	0.001130
heartRate	0	1.000000	0.000235	**-0.000998**	**0.013277**	0.000762	0	1.000000	0.000235
glucose	**9.85909e-05**	< **0.000001**	0.213921	**0.000357**	**0.016576**	0.213663	0	1.000000	0.214020

	$\hat{\boldsymbol{\beta}}_{\lvert\hat{\mathrm{M}}_{\mathrm{y}\lvert 0},\hat{\mathrm{M}}_{\mathrm{y}\lvert 1}}$			$\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$			$\hat{\boldsymbol{\beta}}^*$	
	Estimates	p values	$\lvert\hat{\beta}_j^* - (\hat{\beta}_j)_{\lvert\hat{\mathrm{M}}_{\mathrm{y}\lvert 0},\hat{\mathrm{M}}_{\mathrm{y}\lvert 1}}\rvert$	Estimates	p values	$\lvert\hat{\beta}_j^* - (\hat{\beta}_j)_{\mathrm{CD}}\rvert$	Estimates	p values
Intercept	**-2.141820**	< **0.000001**	0.176820	0	1.000000	2.318640	**-2.318640**	< **0.000001**
sex	0	1.000000	0.434883	0	1.000000	0.434883	**0.434883**	**0.000437**
$\mathrm{education}_1$	0	1.000000	0.140794	0	1.000000	0.140794	0.140794	0.215942
$\mathrm{education}_2$	0	1.000000	0	**-1.97378e-05**	< **0.000001**	0.000020	0	1.000000
$\mathrm{education}_3$	0	1.000000	0	**0.673898**	< **0.000001**	0.673898	0	1.000000
currentSmoker	0	1.000000	0	**-9.77636e-05**	< **0.000001**	0.000098	0	1.000000
BPMeds	-1.772940	0.101122	1.772940	**1.85491e-05**	< **0.000001**	0.000019	0	1.000000
prevalentStroke	**-7.582760**	**0.000017**	8.307947	0	1.000000	0.725187	0.725187	0.194834
prevalentHyp	0	1.000000	0.251681	**0.621115**	< **0.000001**	0.369434	0.251681	0.100156
diabetes	0	1.000000	0	**1.03355e-05**	< **0.000001**	0.000010	0	1.000000
cigsPerDay	0	1.000000	0.275345	**-0.000127**	< **0.000001**	0.275472	**0.275345**	< **0.000001**
age	**1.150930**	< **0.000001**	0.649237	**-0.551358**	< **0.000001**	1.053051	**0.501693**	< **0.000001**
totChol	**0.565858**	**0.000038**	0.562514	**-0.026330**	< **0.000001**	0.029674	**0.003344**	**0.023402**
sysBP	0	1.000000	0.328877	0	1.000000	0.328877	**0.328877**	**0.000004**
diaBP	0	1.000000	0.000021	**0.002551**	< **0.000001**	0.002572	**-0.000021**	**0.000527**
BMI	0	1.000000	0.000253	**0.380223**	< **0.000001**	0.379970	**0.000253**	**0.019268**
heartRate	0	1.000000	0.000235	**-1.52015e-05**	< **0.000001**	0.000220	**-0.000235**	**0.014509**
glucose	0	1.000000	0.214020	0	1.000000	0.214020	**0.214020**	**0.000006**

* The mean absolute difference between $\hat{\boldsymbol{\beta}}^*$ and $\hat{\boldsymbol{\beta}}$ is 0.254136, between $\boldsymbol{\beta}^{(0)}$ is 0.435888, between $\hat{\boldsymbol{\beta}}_{\mathrm{R}}$ is 0.289277, between $\hat{\boldsymbol{\beta}}_{\lvert\hat{\mathrm{M}}_{\mathrm{y}\lvert 0},\hat{\mathrm{M}}_{\mathrm{y}\lvert 1}}$ is 0.728643, and between $\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$ is 0.385935.

Table 3.57: Comparisons of different estimates under the setting of resampled data with a size of $\mathfrak{m} = 585$

	$\hat{\boldsymbol{\beta}}$			$\boldsymbol{\beta}^{(0)}$			$\hat{\boldsymbol{\beta}}_{\mathrm{R}}$		
	Estimates	p values	$\lvert\hat{\beta}_j^* - (\hat{\beta}_j)\rvert$	Estimates	p values	$\lvert\hat{\beta}_j^* - (\beta_j)^{(0)}\rvert$	Estimates	p values	$\lvert\hat{\beta}_j^* - ((\hat{\beta}_j)_{\mathrm{R}}\rvert$
Intercept	**-2.478080**	< **0.000001**	0.159440	**-1.741820**	< **0.000001**	0.576820	**0.001739**	< **0.000001**	2.320379
sex	0.453991	0.111959	0.019108	**0.000255**	< **0.000001**	0.434628	**1.2035e-05**	< **0.000001**	0.434871
$\mathrm{education}_1$	0.264500	0.351890	0.123706	0	1.000000	0.140794	0	1.000000	0.140794
$\mathrm{education}_2$	0	1.000000	0	0	1.000000	0	0	1.000000	0
$\mathrm{education}_3$	0.526044	0.147483	0.526044	0	1.000000	0	**0.000114**	< **0.000001**	0.000114
currentSmoker	0	1.000000	0	0	1.000000	0	**3.20467e-05**	< **0.000001**	0.000032
BPMeds	-0.616418	0.402651	0.616418	0	1.000000	0	0	1.000000	0
prevalentStroke	-1.649980	0.376160	2.375167	**-4.126510**	< **0.000001**	4.851697	0	1.000000	0.725187
prevalentHyp	0.421953	0.195845	0.170272	**0.002600**	< **0.000001**	0.249081	**0.000307**	< **0.000001**	0.251374
diabetes	0.137252	0.819258	0.137252	0	1.000000	0	0	1.000000	0
cigsPerDay	**0.358140**	**0.003439**	0.082795	**0.008897**	**0.000011**	0.266448	0	1.000000	0.275345
age	**0.652764**	**0.000008**	0.151071	**0.703095**	< **0.000001**	0.201402	**-0.008695**	< **0.000001**	0.510388
totChol	**0.437044**	**0.000698**	0.433700	**0.025055**	**0.000098**	0.021711	**-0.000688**	< **0.000001**	0.004032
sysBP	0.207689	0.157456	0.121188	**0.056038**	**0.000008**	0.272839	0	1.000000	0.328877
diaBP	0	1.000000	0.000021	**0.003301**	**0.000021**	0.003321	**0.000178**	< **0.000001**	0.000198
BMI	0.161120	0.188876	0.160867	**0.003471**	**0.000051**	0.003218	**0.001383**	< **0.000001**	0.001130
heartRate	**-0.000732**	< **0.000001**	0.000496	0	1.000000	0.000235	0	1.000000	0.000235
glucose	0	1.000000	0.214020	0	1.000000	0.214020	0	1.000000	0.214020

	$\hat{\boldsymbol{\beta}}_{\lvert\hat{\mathrm{M}}_{\mathrm{y}\lvert 0},\hat{\mathrm{M}}_{\mathrm{y}\lvert 1}}$			$\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$			$\hat{\boldsymbol{\beta}}^*$	
	Estimates	p values	$\lvert\hat{\beta}_j^* - (\hat{\beta}_j)_{\lvert\hat{\mathrm{M}}_{\mathrm{y}\lvert 0},\hat{\mathrm{M}}_{\mathrm{y}\lvert 1}}\rvert$	Estimates	p values	$\lvert\hat{\beta}_j^* - ((\hat{\beta}_j)_{\mathrm{CD}}\rvert$	Estimates	p values
Intercept	**-1.898450**	< **0.000001**	0.420190	**-0.007813**	**0.000018**	2.310827	**-2.318640**	< **0.000001**
sex	0	1.000000	0.434883	**-0.002010**	**0.000001**	0.436893	**0.434883**	**0.000437**
$\mathrm{education}_1$	0	1.000000	0.140794	**-0.000119**	< **0.000001**	0.140913	0.140794	0.215942
$\mathrm{education}_2$	0	1.000000	0	**-0.039413**	**0.000001**	0.039413	0	1.000000
$\mathrm{education}_3$	0	1.000000	0	**0.514151**	< **0.000001**	0.514151	0	1.000000
currentSmoker	0	1.000000	0	**-0.009510**	**0.000001**	0.009510	0	1.000000
BPMeds	0	1.000000	0	0.228630	0.358405	0.228630	0	1.000000
prevalentStroke	-7.665740	0.231450	8.390927	-0.566465	0.347402	1.291652	0.725187	0.194834
prevalentHyp	0	1.000000	0.251681	**0.257896**	**0.003806**	0.006215	0.251681	0.100156
diabetes	0	1.000000	0	0.250032	0.342450	0.250032	0	1.000000
cigsPerDay	1.27308e-05	1.000000	0.275332	**-0.000231**	**0.003878**	0.275576	**0.275345**	< **0.000001**
age	**1.024450**	< **0.000001**	0.522757	**-0.366415**	< **0.000001**	0.868108	**0.501693**	< **0.000001**
totChol	**0.000107**	< **0.000001**	0.003237	**-0.174139**	**0.000033**	0.177483	**0.003344**	**0.023402**
sysBP	0	1.000000	0.328877	**-7.12042e-05**	< **0.000001**	0.328948	**0.328877**	**0.000004**
diaBP	0	1.000000	0.000021	**0.140292**	**0.004312**	0.140313	**-0.000021**	**0.000527**
BMI	0	1.000000	0.000253	**0.358322**	< **0.000001**	0.358069	**0.000253**	**0.019268**
heartRate	0	1.000000	0.000235	**-0.001832**	**0.002821**	0.001597	**-0.000235**	**0.014509**
glucose	0	0.214020	0.214020	**-0.001546**	**0.000606**	0.215566	**0.214020**	**0.000006**

* The mean absolute difference between $\hat{\boldsymbol{\beta}}^*$ and $\hat{\boldsymbol{\beta}}$ is 0.293976, between $\boldsymbol{\beta}^{(0)}$ is 0.402012, between $\hat{\boldsymbol{\beta}}_{\mathrm{R}}$ is 0.289277, between $\hat{\boldsymbol{\beta}}_{\lvert\hat{\mathrm{M}}_{\mathrm{y}\lvert 0},\hat{\mathrm{M}}_{\mathrm{y}\lvert 1}}$ is 0.610178, and between $\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$ is 0.421883.

Table 3.58: Comparisons of different estimates under the setting of resampled data with a size of $\mathfrak{m} = 877$

	$\hat{\boldsymbol{\beta}}$			$\boldsymbol{\beta}^{(0)}$			$\hat{\boldsymbol{\beta}}_{\mathrm{R}}$		
	Estimates	p values	$\lvert\hat{\beta}_j^* - (\hat{\beta}_j)\rvert$	Estimates	p values	$\lvert\hat{\beta}_j^* - (\beta_j)^{(0)}\rvert$	Estimates	p values	$\lvert\hat{\beta}_j^* - ((\hat{\beta}_j)_{\mathrm{R}}\rvert$
Intercept	**-2.164560**	< **0.000001**	0.154080	**-1.736330**	< **0.000001**	0.582310	**0.001739**	< **0.000001**	2.320379
sex	0.378506	0.091350	0.056377	**0.00178333**	< **0.000001**	0.433100	**1.2035e-05**	< **0.000001**	0.434871
$\mathrm{education}_1$	0	1.000000	0.140794	0	1.000000	0.140794	0	1.000000	0.140794
$\mathrm{education}_2$	0	1.000000	0	0	1.000000	0	0	1.000000	0
$\mathrm{education}_3$	0.276016	0.291961	0.276016	**2.55513e-05**	< **0.000001**	0.000026	**0.000114**	< **0.000001**	0.000114
currentSmoker	0	1.000000	0	**0.000558**	< **0.000001**	0.000558	**3.20467e-05**	< **0.000001**	0.000032
BPMeds	-0.400312	0.435766	0.400312	0	1.000000	0	0	1.000000	0
prevalentStroke	0	1.000000	0.725187	0	1.000000	0.725187	0	1.000000	0.725187
prevalentHyp	0.391514	0.140840	0.139833	**0.007785**	< **0.000001**	0.243896	**0.000307**	< **0.000001**	0.251374
diabetes	0	1.000000	0	0	1.000000	0	0	1.000000	0
cigsPerDay	**0.289124**	**0.003991**	0.013779	**0.032504**	**0.000012**	0.242841	0	1.000000	0.275345
age	**0.539559**	**0.000001**	0.037866	**0.560815**	< **0.000001**	0.059122	**-0.008695**	< **0.000001**	0.510388
totChol	0.208652*	0.059147*	0.205308	**0.004265**	**0.000154**	0.000921	**-0.000688**	< **0.000001**	0.004032
sysBP	**0.269873**	**0.019704**	0.059004	**0.141940**	**0.004379**	0.186937	0	1.000000	0.328877
diaBP	0	1.000000	0.000021	**0.001568**	< **0.000001**	0.001588	**0.000178**	< **0.000001**	0.000198
BMI	**9.03052e-05**	< **0.000001**	0.000163	**0.008607**	**0.000183**	0.008354	**0.001383**	< **0.000001**	0.001130
heartRate	**-4.85764e-05**	< **0.000001**	0.000187	**-9.68837e-05**	**0.000021**	0.000139	0	1.000000	0.000235
glucose	**0.001610**	< **0.000001**	0.212410	**0.005399**	**0.000047**	0.208621	0	1.000000	0.214020

	$\hat{\boldsymbol{\beta}}_{\lvert\hat{\mathrm{M}}_{\mathrm{y}\lvert 0},\hat{\mathrm{M}}_{\mathrm{y}\lvert 1}}$			$\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$			$\hat{\boldsymbol{\beta}}^*$	
	Estimates	p values	$\lvert\hat{\beta}_j^* - (\hat{\beta}_j)_{\lvert\hat{\mathrm{M}}_{\mathrm{y}\lvert 0},\hat{\mathrm{M}}_{\mathrm{y}\lvert 1}}\rvert$	Estimates	p values	$\lvert\hat{\beta}_j^* - ((\hat{\beta}_j)_{\mathrm{CD}}\rvert$	Estimates	p values
Intercept	**-1.821960**	< **0.000001**	0.496680	**-0.000210**	< **0.000001**	2.318430	**-2.318640**	< **0.000001**
sex	0	1.000000	0.434883	0	1.000000	0.434883	**0.434883**	**0.000437**
$\mathrm{education}_1$	0	1.000000	0.140794	0	1.000000	0.140794	0.140794	0.215942
$\mathrm{education}_2$	0	1.000000	0	**-1.55477e-05**	< **0.000001**	0.000016	0	1.000000
$\mathrm{education}_3$	0	1.000000	0	**2.75924e-05**	< **0.000001**	0.000028	0	1.000000
currentSmoker	0	1.000000	0	**-1.28153e-05**	< **0.000001**	0.000013	0	1.000000
BPMeds	0	1.000000	0	0	1.000000	0	0	1.000000
prevalentStroke	0	1.000000	0.725187	0	1.000000	0.725187	0.725187	0.194834
prevalentHyp	0	1.000000	0.251681	**6.80266e-05**	< **0.000001**	0.251613	0.251681	0.100156
diabetes	0	1.000000	0	0	1.000000	0	0	1.000000
cigsPerDay	**0.000182**	< **0.000001**	0.275163	0	1.000000	0.275345	**0.275345**	< **0.000001**
age	**0.820463**	< **0.000001**	0.318770	**-0.002465**	< **0.000001**	0.504158	**0.501693**	< **0.000001**
totChol	0	1.000000	0.003344	**-0.000238**	< **0.000001**	0.003582	**0.003344**	**0.023402**
sysBP	**0.000396**	< **0.000001**	0.328481	**7.27173e-05**	< **0.000001**	0.328804	**0.328877**	**0.000004**
diaBP	0	1.000000	0.000021	**0.001130**	< **0.000001**	0.001151	**-0.000021**	**0.000527**
BMI	0	1.000000	0.000253	**0.006565**	< **0.000001**	0.006312	**0.000253**	**0.019268**
heartRate	0	1.000000	0.000235	0	1.000000	0.000235	**-0.000235**	**0.014509**
glucose	0	1.000000	0.214020	0	1.000000	0.214020	**0.214020**	**0.000006**

* The mean absolute difference between $\hat{\boldsymbol{\beta}}^*$ and $\hat{\boldsymbol{\beta}}$ is 0.134519, between $\boldsymbol{\beta}^{(0)}$ is 0.157466, between $\hat{\boldsymbol{\beta}}_{\mathrm{R}}$ is 0.289277, between $\hat{\boldsymbol{\beta}}_{\lvert\hat{\mathrm{M}}_{\mathrm{y}\lvert 0},\hat{\mathrm{M}}_{\mathrm{y}\lvert 1}}$ is 0.177195, and between $\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$ is 0.289143.

Table 3.59: Comparisons of different estimates under the setting of resampled data with a size of $\mathfrak{m} = 1170$

	$\hat{\boldsymbol{\beta}}$			$\boldsymbol{\beta}^{(0)}$			$\hat{\boldsymbol{\beta}}_{\mathrm{R}}$		
	Estimates	p values	$\lvert\hat{\beta}_j^* - (\hat{\beta}_j)\rvert$	Estimates	p values	$\lvert\hat{\beta}_j^* - (\beta_j)^{(0)}\rvert$	Estimates	p values	$\lvert\hat{\beta}_j^* - ((\hat{\beta}_j)_{\mathrm{R}}\rvert$
Intercept	**-2.218800**	< **0.000001**	0.099840	**-1.687260**	< **0.000001**	0.631380	**0.001739**	< **0.000001**	2.320379
sex	**0.389353**	**0.037147**	0.045530	**0.000139**	< **0.000001**	0.434744	**1.2035e-05**	< **0.000001**	0.434871
$\mathrm{education}_1$	0	1.000000	0.140794	**3.53468e-05**	< **0.000001**	0.140759	0	1.000000	0.140794
$\mathrm{education}_2$	0	1.000000	0	**-3.63295e-05**	< **0.000001**	0.000036	0	1.000000	0
$\mathrm{education}_3$	0	1.000000	0	0	1.000000	0	**0.000114**	< **0.000001**	0.000114
currentSmoker	0	1.000000	0	**3.08897e-05**	< **0.000001**	0.000031	**3.20467e-05**	< **0.000001**	0.000032
BPMeds	-0.323604	0.467661	0.323604	0	1.000000	0	0	1.000000	0
prevalentStroke	0.602492	0.341943	0.122695	0	1.000000	0.725187	0	1.000000	0.725187
prevalentHyp	0.285392	0.202582	0.033711	**0.002462**	< **0.000001**	0.249219	**0.000307**	< **0.000001**	0.251374
diabetes	0.495262	0.164316	0.495262	0	1.000000	0	0	1.000000	0
cigsPerDay	**0.242178**	**0.004091**	0.033167	**0.002030**	< **0.000001**	0.273315	0	1.000000	0.275345
age	**0.479056**	< **0.000001**	0.022637	**0.065963**	< **0.000001**	0.435731	**-0.008695**	< **0.000001**	0.510388
totChol	**0.225585**	**0.007793**	0.222241	**0.004317**	< **0.000001**	0.000973	**-0.000688**	< **0.000001**	0.004032
sysBP	**0.294279**	**0.002168**	0.034598	**0.0659473**	< **0.000001**	0.262930	0	1.000000	0.328877
diaBP	0	1.000000	0.000021	**0.004994**	< **0.000001**	0.005014	**0.000178**	< **0.000001**	0.000198
BMI	**1.34427e-05**	< **0.000001**	0.000239	**0.002991**	< **0.000001**	0.002738	**0.001383**	< **0.000001**	0.001130
heartRate	**-0.000788**	< **0.000001**	0.000553	0	1.000000	0.000235	0	1.000000	0.000235
glucose	0	1.000000	0.214020	**0.000771**	< **0.000001**	0.213249	0	1.000000	0.214020

	$\hat{\boldsymbol{\beta}}_{\lvert\hat{\mathrm{M}}_{\mathrm{y}\lvert 0},\hat{\mathrm{M}}_{\mathrm{y}\lvert 1}}$			$\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$			$\hat{\boldsymbol{\beta}}^*$	
	Estimates	p values	$\lvert\hat{\beta}_j^* - (\hat{\beta}_j)_{\lvert\hat{\mathrm{M}}_{\mathrm{y}\lvert 0},\hat{\mathrm{M}}_{\mathrm{y}\lvert 1}}\rvert$	Estimates	p values	$\lvert\hat{\beta}_j^* - ((\hat{\beta}_j)_{\mathrm{CD}}\rvert$	Estimates	p values
Intercept	**-1.884990**	< **0.000001**	0.433650	**-0.005358**	< **0.000001**	2.313282	**-2.318640**	< **0.000001**
sex	0	1.000000	0.434883	**-2.48346e-05**	< **0.000001**	0.434908	**0.434883**	**0.000437**
$\mathrm{education}_1$	0	1.000000	0.140794	**-2.95232e-05**	< **0.000001**	0.140824	0.140794	0.215942
$\mathrm{education}_2$	0	1.000000	0	**-0.000141**	< **0.000001**	0.000141	0	1.000000
$\mathrm{education}_3$	0	1.000000	0	0	1.000000	0	0	1.000000
currentSmoker	0	1.000000	0	**-6.34966e-05**	< **0.000001**	0.000063	0	1.000000
BPMeds	0	1.000000	0	0	1.000000	0	0	1.000000
prevalentStroke	0	1.000000	0.725187	0	1.000000	0.725187	0.725187	0.194834
prevalentHyp	0	1.000000	0.251681	0	1.000000	0.251681	0.251681	0.100156
diabetes	0	1.000000	0	0	1.000000	0	0	1.000000
cigsPerDay	0	1.000000	0.275345	**1.0515e-05**	< **0.000001**	0.275334	**0.275345**	< **0.000001**
age	**0.749713**	< **0.000001**	0.248020	**-0.000592**	< **0.000001**	0.502285	**0.501693**	< **0.000001**
totChol	**2.72758e-05**	< **0.000001**	0.003317	**-5.36505e-05**	< **0.000001**	0.003398	**0.003344**	**0.023402**
sysBP	**0.002222**	< **0.000001**	0.326655	0	1.000000	0.328877	**0.328877**	**0.000004**
diaBP	0	1.000000	0.000021	**0.000396**	< **0.000001**	0.000417	**-0.000021**	**0.000527**
BMI	0	1.000000	0.000253	**0.004197**	< **0.000001**	0.003944	**0.000253**	**0.019268**
heartRate	0	1.000000	0.000235	0	1.000000	0.000235	**-0.000235**	**0.014509**
glucose	0	1.000000	0.214020	0	1.000000	0.214020	**0.214020**	**0.000006**

* The mean absolute difference between $\hat{\boldsymbol{\beta}}^*$ and $\hat{\boldsymbol{\beta}}$ is 0.099384, between $\boldsymbol{\beta}^{(0)}$ is 0.187530, between $\hat{\boldsymbol{\beta}}_{\mathrm{R}}$ is 0.289277, between $\hat{\boldsymbol{\beta}}_{\lvert\hat{\mathrm{M}}_{\mathrm{y}\lvert 0},\hat{\mathrm{M}}_{\mathrm{y}\lvert 1}}$ is 0.169670, and between $\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$ is 0.288589.

Table 3.60: Comparisons of different estimates under the setting of resampled data with a size of $\mathfrak{m} = 1462$

	$\hat{\boldsymbol{\beta}}$			$\boldsymbol{\beta}^{(0)}$			$\hat{\boldsymbol{\beta}}_{\mathrm{R}}$		
	Estimates	p values	$\lvert\hat{\beta}_j^* - (\hat{\beta}_j)\rvert$	Estimates	p values	$\lvert\hat{\beta}_j^* - (\beta_j)^{(0)}\rvert$	Estimates	p values	$\lvert\hat{\beta}_j^* - ((\hat{\beta}_j)_{\mathrm{R}}\rvert$
Intercept	**-2.219140**	< **0.000001**	0.099500	**-1.852040**	< **0.000001**	0.466600	**0.001739**	< **0.000001**	2.320379
sex	**0.351653**	**0.035817**	0.083230	**0.000185**	< **0.000001**	0.434698	**1.2035e-05**	< **0.000001**	0.434871
$\mathrm{education}_1$	0	1.000000	0.140794	0	1.000000	0.140794	0	1.000000	0.140794
$\mathrm{education}_2$	0	1.000000	0	0	1.000000	0	0	1.000000	0
$\mathrm{education}_3$	0	1.000000	0	0	1.000000	0	**0.000114**	< **0.000001**	0.000114
currentSmoker	0	1.000000	0	**0.000040**	< **0.000001**	0.000040	**3.20467e-05**	< **0.000001**	0.000032
BPMeds	0	1.000000	0	0	1.000000	0	0	1.000000	0
prevalentStroke	0.779235	0.187353	0.054048	0.923963	0.281815	0.198776	0	1.000000	0.725187
prevalentHyp	0.282824	0.179188	0.031143	**0.000438**	< **0.000001**	0.251243	**0.000307**	< **0.000001**	0.251374
diabetes	**0.932592**	**0.011117**	0.932592	0	1.000000	0	0	1.000000	0
cigsPerDay	**0.236242**	**0.001608**	0.039103	**0.002084**	< **0.000001**	0.273261	0	1.000000	0.275345
age	**0.502532**	< **0.000001**	0.000839	**0.557206**	< **0.000001**	0.055513	**-0.008695**	< **0.000001**	0.510388
totChol	**0.002526**	< **0.000001**	0.000818	**0.000439**	< **0.000001**	0.002905	**-0.000688**	< **0.000001**	0.004032
sysBP	**0.295916**	**0.001131**	0.032961	**0.110056**	< **0.000001**	0.218821	0	1.000000	0.328877
diaBP	0	1.000000	0.000021	**0.001478**	< **0.000001**	0.001499	**0.000178**	< **0.000001**	0.000198
BMI	**0.000019**	< **0.000001**	0.000234	**0.000725**	< **0.000001**	0.000472	**0.001383**	< **0.000001**	0.001130
heartRate	**-0.000572**	< **0.000001**	0.000337	0	1.000000	0.000235	0	1.000000	0.000235
glucose	**0.000922**	< **0.000001**	0.213098	**0.006676**	**0.000004**	0.207344	0	1.000000	0.214020

	$\hat{\boldsymbol{\beta}}_{\lvert\hat{\mathrm{M}}_{\mathrm{y}\lvert 0},\hat{\mathrm{M}}_{\mathrm{y}\lvert 1}}$			$\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$			$\hat{\boldsymbol{\beta}}^*$	
	Estimates	p values	$\lvert\hat{\beta}_j^* - (\hat{\beta}_j)_{\lvert\hat{\mathrm{M}}_{\mathrm{y}\lvert 0},\hat{\mathrm{M}}_{\mathrm{y}\lvert 1}}\rvert$	Estimates	p values	$\lvert\hat{\beta}_j^* - ((\hat{\beta}_j)_{\mathrm{CD}}\rvert$	Estimates	p values
Intercept	**-1.911890**	< **0.000001**	0.406750	**-0.010916**	< **0.000001**	2.307724	**-2.318640**	< **0.000001**
sex	0	1.000000	0.434883	**-0.000036**	< **0.000001**	0.434919	**0.434883**	**0.000437**
$\mathrm{education}_1$	0	1.000000	0.140794	0	1.000000	0.140794	0.140794	0.215942
$\mathrm{education}_2$	0	1.000000	0	**-0.000056**	< **0.000001**	0.000056	0	1.000000
$\mathrm{education}_3$	0	1.000000	0	0	1.000000	0	0	1.000000
currentSmoker	0	1.000000	0	**-0.000115**	< **0.000001**	0.000115	0	1.000000
BPMeds	0	1.000000	0	0	1.000000	0	0	1.000000
prevalentStroke	0.738087	0.383907	0.012900	0	1.000000	0.725187	0.725187	0.194834
prevalentHyp	0	1.000000	0.251681	0	1.000000	0.251681	0.251681	0.100156
diabetes	0	1.000000	0	0	1.000000	0	0	1.000000
cigsPerDay	**0.000012**	< **0.000001**	0.275333	0	1.000000	0.275345	**0.275345**	< **0.000001**
age	**0.705260**	< **0.000001**	0.203567	**-0.000017**	< **0.000001**	0.501710	**0.501693**	< **0.000001**
totChol	0	1.000000	0.003344	0	1.000000	0.003344	**0.003344**	**0.023402**
sysBP	**0.000876**	< **0.000001**	0.328001	**0.000056**	< **0.000001**	0.328821	**0.328877**	**0.000004**
diaBP	0	1.000000	0.000021	**0.000039**	< **0.000001**	0.000060	**-0.000021**	**0.000527**
BMI	0	1.000000	0.000253	**0.000957**	< **0.000001**	0.000704	**0.000253**	**0.019268**
heartRate	0	1.000000	0.000235	0	1.000000	0.000235	**-0.000235**	**0.014509**
glucose	**0.000032**	< **0.000001**	0.213988	0	1.000000	0.214020	**0.214020**	**0.000006**

* The mean absolute difference between $\hat{\boldsymbol{\beta}}^*$ and $\hat{\boldsymbol{\beta}}$ is 0.090484, between $\boldsymbol{\beta}^{(0)}$ is 0.125122, between $\hat{\boldsymbol{\beta}}_{\mathrm{R}}$ is 0.289277, between $\hat{\boldsymbol{\beta}}_{\lvert\hat{\mathrm{M}}_{\mathrm{y}\lvert 0},\hat{\mathrm{M}}_{\mathrm{y}\lvert 1}}$ is 0.126208, and between $\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$ is 0.288040.

Scenario 2: In this scenario, we test the methods mentioned at the beginning of this section in different scenarios involving various mislabelled data. Specifically, we preset different settings for parameters $\boldsymbol{\eta_0}$ and $\boldsymbol{\eta_1}$ as outlined in **CASE 1-CASE 3**, which corrupt the FHS dataset differently.

CASE 1:

$$\boldsymbol{\eta_0} = (0, \cdots, \eta_{0,4}, \eta_{0,5}, \cdots, \eta_{0,10}, \cdots, \eta_{0,13}, \cdots, \eta_{0,17}, \eta_{0,18})^{\mathsf{T}}_{\mathrm{p}\times 1},$$

$$\eta_{0,4} = \eta_{0,5} = \eta_{0,10} = 1,\ \eta_{0,13} = 0.55,\ \eta_{0,17} = \eta_{0,18} = -0.55;$$

$$\boldsymbol{\eta_1} = (0, \cdots, \eta_{1,12}, \cdots, \eta_{1,15}, \eta_{1,16}, \cdots, 0)^{\mathsf{T}}_{\mathrm{p}\times 1},$$

$$\eta_{1,12} = -0.5,\ \eta_{0,15} = -0.35,\ \eta_{1,16} = 0.65;$$

CASE 2:

$$\boldsymbol{\eta_0} = (0, 0, \eta_{0,3}, \cdots, \eta_{0,6}, \eta_{0,7}, \cdots, 0)^{\mathsf{T}}_{\mathfrak{p}\times 1},$$
$$\eta_{0,3} = -0.65,\ \eta_{0,6} = 0.5,\ \eta_{0,7} = -0.55;$$
$$\boldsymbol{\eta_1} = (0, \cdots, \eta_{1,13}, \eta_{1,14}, \cdots, 0)^{\mathsf{T}}_{\mathfrak{p}\times 1},$$
$$\eta_{1,13} = 0.5,\ \eta_{1,14} = 0.85;$$

CASE 3:

$$\boldsymbol{\eta_0} = (0, \eta_{0,[2:4]}, \cdots, \eta_{0,8}, \cdots, \eta_{0,12}, \cdots, \eta_{0,15}, \cdots, \eta_{0,18})^{\mathsf{T}}_{\mathfrak{p}\times 1},$$
$$\eta_{0,2} = \eta_{0,3} = \eta_{0,4} = 1, \eta_{0,12} = 0.65,\ \eta_{0,15} = -0.75,\ \eta_{0,18} = 0.5;$$
$$\boldsymbol{\eta_1} = (0, \cdots, \eta_{1,4}, \cdots \eta_{1,8}, \cdots, \eta_{1,12}, \eta_{1,16}, \cdots, 0)^{\mathsf{T}}_{\mathfrak{p}\times 1},$$
$$\eta_{1,4} = \eta_{1,8} = 1,\ \eta_{1,12} = 0.65,\ \eta_{1,16} = -0.85,$$

where $\mathfrak{p} = 18$. We fix the resampled datasets as $\mathfrak{m} = 1170$ for each case, representing 40% of the training data resampled. Table 3.61-Table 3.65 display the results of **Scenario 2**.

As shown in Table 3.62, the mislabelling ratios of the resampled datasets with 40% of the training data checked are very close to those from the resampled datasets of the same size as the training datasets.

Table 3.62 showcases the performance of the classifiers from different methods. As shown in the table, our classifier $C_{\hat{\boldsymbol{\beta}}}$s have the closest performance to classifier $C_{\hat{\boldsymbol{\beta}}^*}$ and even the same misclassification rate in **CASE 2**, outperforming other classifiers in all cases. It is noted that $C_{\hat{\boldsymbol{\beta}}_R}$, trained by the raw corrupted data, performs worst, with more than half of the testing data misclassified. Another noteworthy point is the unstable performance of $C_{\hat{\boldsymbol{\beta}}_{CD}}$s across different cases, derived from the method that corrects the labels in the resampled dataset while ignoring noisy labels during estimation. Our classifier clearly outperforms both $C_{\boldsymbol{\beta}^{(0)}}$ and $C_{\hat{\boldsymbol{\beta}}_{|\hat{M}_Y|0,\hat{M}_Y|1}}$, with fewer testing data misclassified.

Table 3.63-Table 3.65 display the details of the estimates, p-values from the Wald test, and the difference between the estimates from different methods compared to the method using all corrected data. Consistent with the performance in **Scenario 1**, other methods generate estimates, namely $\hat{\boldsymbol{\beta}}_R$s, $\hat{\boldsymbol{\beta}}_{|\hat{M}_Y|0,\hat{M}_Y|1}$ and $\hat{\boldsymbol{\beta}}_{CD}$, close to 0, rendering the models less interpretable. On the other hand, $\hat{\boldsymbol{\beta}}$s closely resemble $\hat{\boldsymbol{\beta}}^*$

across different cases, performing well in variable selection and having the least mean absolute difference.

Table 3.61: Averaged mislabelling ratios of full-size generated and resampled datasets

	resample size	$\mathfrak{m} = 3656 \quad (\mathfrak{m} = \mathfrak{n})$	$\mathfrak{m} = 1170$
CASE 1	$\hat{\mathrm{M}}_{\mathfrak{y}\|0}$	0.617056	0.611562
	$\hat{\mathrm{M}}_{\mathfrak{y}\|1}$	0.555809	0.543478
	$\hat{\mathrm{M}}$	0.607863	0.600855
CASE 2	$\hat{\mathrm{M}}_{\mathfrak{y}\|0}$	0.497586	0.492901
	$\hat{\mathrm{M}}_{\mathfrak{y}\|1}$	0.398633	0.396739
	$\hat{\mathrm{M}}$	0.482735	0.477778
CASE 3	$\hat{\mathrm{M}}_{\mathfrak{y}\|0}$	0.706758	0.701826
	$\hat{\mathrm{M}}_{\mathfrak{y}\|1}$	0.405467	0.380435
	$\hat{\mathrm{M}}$	0.661538	0.651282

* This table displays the mislabelling ratios for two sets of resampled datasets. The first set consists of datasets with the same size as the training dataset, which is $\mathfrak{m} = \mathfrak{n} = 3656$. The second set includes datasets with a size of $\mathfrak{m} = 1170$, representing 40% of the training dataset.

Table 3.62: Misclassification rates of various classifiers (in $\%$)

$\mathfrak{m} = 1170$	$\mathrm{Mis}_{\mathrm{C}_{\hat{\boldsymbol{\beta}}^{(0)}}}$	$\mathrm{Mis}_{\mathrm{C}_{\hat{\boldsymbol{\beta}}}}$	$\mathrm{Mis}_{\mathrm{C}_{\hat{\boldsymbol{\beta}}_{\|\hat{\mathrm{M}}_{\mathfrak{y}\|0},\hat{\mathrm{M}}_{\mathfrak{y}\|1}}}}$	$\mathrm{Mis}_{\mathrm{C}_{\hat{\boldsymbol{\beta}}_{\mathrm{CD}}}}$	$\mathrm{Mis}_{\mathrm{C}_{\hat{\boldsymbol{\beta}}_{\mathrm{R}}}}$	$\mathrm{Mis}_{\mathrm{C}_{\hat{\boldsymbol{\beta}}^{*}}}$
CASE 1	16.1423	15.7319	16.1423	17.3735	64.2955	15.5951
CASE 2	16.1423	15.5951	16.1423	16.1423	59.5075	15.5951
CASE 3	16.1423	16.0055	16.1423	39.8085	74.4186	15.5951

* The same out-of-sample testing dataset consisting of 731 observations is used for all cases.

Table 3.63: Comparisons of different estimates of **CASE 1** under the setting of resampled data with a size of $\mathfrak{m} = 1170$

	$\hat{\boldsymbol{\beta}}$			$\boldsymbol{\beta}^{(0)}$			$\hat{\boldsymbol{\beta}}_{\mathrm{R}}$		
	Estimates	p values	$\lvert\hat{\beta}_j^* - (\hat{\beta}_j)\rvert$	Estimates	p values	$\lvert\hat{\beta}_j^* - (\beta_j)^{(0)}\rvert$	Estimates	p values	$\lvert\hat{\beta}_j^* - ((\hat{\beta}_j)_{\mathrm{R}}\rvert$
Intercept	**-2.175710**	< **0.000001**	0.142930	**-1.687370**	< **0.000001**	0.631270	**0.000237**	< **0.000001**	2.318877
sex	0.353969*	0.069658*	0.080914	**0.000141**	< **0.000001**	0.434742	**-0.000049**	< **0.000001**	0.434932
$\mathrm{education}_1$	0	1.000000	0.140794	**0.000035**	< **0.000001**	0.140759	0	1.000000	0.140794
$\mathrm{education}_2$	0	1.000000	0	**-0.000037**	< **0.000001**	0.000037	**0.787064**	< **0.000001**	0.787064
$\mathrm{education}_3$	0	1.000000	0	0	1.000000	0	**0.862666**	< **0.000001**	0.862666
currentSmoker	**0.000013**	< **0.000001**	0.000013	**0.000031**	< **0.000001**	0.000031	0	1.000000	0
BPMeds	**-0.000013**	< **0.000001**	0.000013	0	1.000000	0	0	1.000000	0
prevalentStroke	0.405505	0.739303	0.319682	0	1.000000	0.725187	0	1.000000	0.725187
prevalentHyp	0.288291	0.210789	0.036610	**0.002476**	< **0.000001**	0.249205	**-0.000219**	< **0.000001**	0.251900
diabetes	0	1.000000	0	0	1.000000	0	0	1.000000	0
cigsPerDay	**0.268030**	**0.002448**	0.007315	**0.002050**	< **0.000001**	0.273295	**-0.000158**	< **0.000001**	0.275503
age	**0.510372**	< **0.000001**	0.008679	**0.066495**	< **0.000001**	0.435198	**-0.008797**	< **0.000001**	0.510490
totChol	**0.209548**	**0.021641**	0.206204	**0.004340**	< **0.000001**	0.000996	**0.393077**	< **0.000001**	0.389733
sysBP	**0.272608**	**0.008622**	0.056269	**0.066453**	< **0.000001**	0.262424	**-0.000901**	< **0.000001**	0.329778
diaBP	**-0.000016**	< **0.000001**	0.000005	**0.005003**	< **0.000001**	0.005023	**-0.000027**	< **0.000001**	0.000006
BMI	**0.000904**	< **0.000001**	0.000651	**0.003006**	< **0.000001**	0.002753	**0.000466**	< **0.000001**	0.000213
heartRate	**-0.000312**	< **0.000001**	0.000077	0	1.000000	0.000235	**-0.458471**	< **0.000001**	0.458236
glucose	**0.000264**	< **0.000001**	0.213756	**0.000776**	< **0.000001**	0.213244	**-0.086379**	< **0.000001**	0.300399

	$\hat{\boldsymbol{\beta}}_{\lvert\hat{\mathrm{M}}_{\mathrm{y}\lvert 0},\hat{\mathrm{M}}_{\mathrm{y}\lvert 1}}$			$\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$			$\hat{\boldsymbol{\beta}}^*$	
	Estimates	p values	$\lvert\hat{\beta}_j^* - (\hat{\beta}_j)_{\lvert\hat{\mathrm{M}}_{\mathrm{y}\lvert 0},\hat{\mathrm{M}}_{\mathrm{y}\lvert 1}}\rvert$	Estimates	p values	$\lvert\hat{\beta}_j^* - ((\hat{\beta}_j)_{\mathrm{CD}}\rvert$	Estimates	p values
Intercept	**-1.675890**	< **0.000001**	0.642750	**-0.006872**	< **0.000001**	2.311768	**-2.318640**	< **0.000001**
sex	0	1.000000	0.434883	**-0.000167**	< **0.000001**	0.435050	**0.434883**	**0.000437**
$\mathrm{education}_1$	0	1.000000	0.140794	**-0.000209**	< **0.000001**	0.141003	0.140794	0.215942
$\mathrm{education}_2$	0	1.000000	0	**-0.000025**	< **0.000001**	0.000025	0	1.000000
$\mathrm{education}_3$	0	1.000000	0	0	1.000000	0	0	1.000000
currentSmoker	0	1.000000	0	**-0.000055**	< **0.000001**	0.000055	0	1.000000
BPMeds	0	1.000000	0	0	1.000000	0	0	1.000000
prevalentStroke	0	1.000000	0.725187	0	1.000000	0.725187	0.725187	0.194834
prevalentHyp	0	1.000000	0.251681	**-0.000024**	< **0.000001**	0.251705	0.251681	0.100156
diabetes	0	1.000000	0	0	1.000000	0	0	1.000000
cigsPerDay	0	1.000000	0.275345	0	1.000000	0.275345	**0.275345**	< **0.000001**
age	**0.004722**	< **0.000001**	0.496971	0	1.000000	0.501693	**0.501693**	< **0.000001**
totChol	0	1.000000	0.003344	**0.003217**	< **0.000001**	0.000127	**0.003344**	**0.023402**
sysBP	**0.004694**	< **0.000001**	0.324183	0	1.000000	0.328877	**0.328877**	**0.000004**
diaBP	**0.000023**	< **0.000001**	0.000044	0	1.000000	0.000021	**-0.000021**	**0.000527**
BMI	0	1.000000	0.000253	**0.000069**	< **0.000001**	0.000183	**0.000253**	**0.019268**
heartRate	0	1.000000	0.000235	**-0.002674**	< **0.000001**	0.002438	**-0.000235**	**0.014509**
glucose	0	1.000000	0.214020	**-0.000099**	< **0.000001**	0.214119	**0.214020**	**0.000006**

* The mean absolute difference between $\hat{\boldsymbol{\beta}}^*$ and $\hat{\boldsymbol{\beta}}$ is 0.067440, between $\boldsymbol{\beta}^{(0)}$ is 0.187467, between $\hat{\boldsymbol{\beta}}_{\mathrm{R}}$ is 0.43254, between $\hat{\boldsymbol{\beta}}_{\lvert\hat{\mathrm{M}}_{\mathrm{y}\lvert 0},\hat{\mathrm{M}}_{\mathrm{y}\lvert 1}}$ is 0.194983, and between $\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$ is 0.288200.

Table 3.64: Comparisons of different estimates of **CASE 2** under the setting of resampled data with a size of $\mathfrak{m} = 1170$

	$\hat{\boldsymbol{\beta}}$			$\boldsymbol{\beta}^{(0)}$			$\hat{\boldsymbol{\beta}}_{\mathrm{R}}$		
	Estimates	p values	$\lvert\hat{\beta}_j^* - (\hat{\beta}_j)\rvert$	Estimates	p values	$\lvert\hat{\beta}_j^* - (\beta_j)^{(0)}\rvert$	Estimates	p values	$\lvert\hat{\beta}_j^* - ((\hat{\beta}_j)_{\mathrm{R}}\rvert$
Intercept	**-2.152810**	< **0.000001**	0.165830	**-1.684130**	< **0.000001**	0.634510	0.075844	0.939544	2.394484
sex	0.329969*	0.064563*	0.104914	**0.000097**	< **0.000001**	0.434786	0.000132	0.999895	0.434751
$\mathrm{education}_1$	0	1.000000	0.140794	**0.000029**	< **0.000001**	0.140765	-0.511621	0.608916	0.652415
$\mathrm{education}_2$	**-0.000052**	< **0.000001**	0.000052	**-0.000029**	< **0.000001**	0.000029	0.000422	0.999664	0.000422
$\mathrm{education}_3$	0	1.000000	0	0	1.000000	0	0.000212	0.999831	0.000212
currentSmoker	0	1.000000	0	**0.000020**	< **0.000001**	0.000020	0.239774	0.810505	0.239774
BPMeds	**-0.000029**	< **0.000001**	0.000029	0	1.000000	0	-0.000031	0.999975	0.000031
prevalentStroke	-0.190789	0.817819	0.915976	0	1.000000	0.725187	0	1.000000	0.725187
prevalentHyp	0.276169	0.191386	0.024488	**0.001953**	< **0.000001**	0.249728	0.000528	0.999579	0.251153
diabetes	0	1.000000	0	0	1.000000	0	0	1.000000	0
cigsPerDay	**0.264991**	**0.001289**	0.010354	**0.001378**	< **0.000001**	0.273967	0.000103	0.999918	0.275242
age	**0.454063**	< **0.000001**	0.047630	**0.048939**	< **0.000001**	0.452755	-0.000021	0.999983	0.501714
totChol	**0.250813**	**0.002257**	0.247469	**0.003424**	< **0.000001**	0.000080	0.003050	0.997567	0.000294
sysBP	**0.278914**	**0.003344**	0.049963	**0.049088**	< **0.000001**	0.279789	0.003312	0.997357	0.325565
diaBP	0	1.000000	0.000021	**0.004475**	< **0.000001**	0.004495	0.013260	0.989420	0.013281
BMI	**0.001127**	< **0.000001**	0.000874	**0.002413**	< **0.000001**	0.002160	0.000024	0.999980	0.000228
heartRate	**-0.001607**	< **0.000001**	0.001371	0	1.000000	0.000235	0.000028	0.999977	0.000264
glucose	**0.011130**	< **0.000001**	0.202890	**0.000605**	< **0.000001**	0.213415	0.000316	0.999748	0.213704

	$\hat{\boldsymbol{\beta}}_{\lvert\hat{\mathrm{M}}_{\mathrm{y}\lvert 0},\hat{\mathrm{M}}_{\mathrm{y}\lvert 1}}$			$\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$			$\hat{\boldsymbol{\beta}}^*$	
	Estimates	p values	$\lvert\hat{\beta}_j^* - (\hat{\beta}_j)_{\lvert\hat{\mathrm{M}}_{\mathrm{y}\lvert 0},\hat{\mathrm{M}}_{\mathrm{y}\lvert 1}}\rvert$	Estimates	p values	$\lvert\hat{\beta}_j^* - ((\hat{\beta}_j)_{\mathrm{CD}}\rvert$	Estimates	p values
Intercept	**-1.753630**	< **0.000001**	0.565010	**-0.008818**	< **0.000001**	2.309822	**-2.318640**	< **0.000001**
sex	0	1.000000	0.434883	**-0.000122**	< **0.000001**	0.435005	**0.434883**	**0.000437**
$\mathrm{education}_1$	0	1.000000	0.140794	**-0.002420**	< **0.000001**	0.143214	0.140794	0.215942
$\mathrm{education}_2$	0	1.000000	0	**-0.000089**	< **0.000001**	0.000089	0	1.000000
$\mathrm{education}_3$	0	1.000000	0	0	1.000000	0	0	1.000000
currentSmoker	0	1.000000	0	**-0.000245**	< **0.000001**	0.000245	0	1.000000
BPMeds	0	1.000000	0	0	1.000000	0	0	1.000000
prevalentStroke	0	1.000000	0.725187	0	1.000000	0.725187	0.725187	0.194834
prevalentHyp	0	1.000000	0.251681	**-0.000013**	< **0.000001**	0.251694	0.251681	0.100156
diabetes	0	1.000000	0	0	1.000000	0	0	1.000000
cigsPerDay	0	1.000000	0.275345	**0.000056**	< **0.000001**	0.275289	**0.275345**	< **0.000001**
age	**0.000843**	< **0.000001**	0.500850	0	1.000000	0.501693	**0.501693**	< **0.000001**
totChol	**0.000014**	< **0.000001**	0.003330	**0.000045**	< **0.000001**	0.003299	**0.003344**	**0.023402**
sysBP	**0.006068**	< **0.000001**	0.322809	**0.000053**	< **0.000001**	0.328824	**0.328877**	**0.000004**
diaBP	**0.000180**	< **0.000001**	0.000200	**0.000047**	< **0.000001**	0.000068	**-0.000021**	**0.000527**
BMI	**0.000023**	< **0.000001**	0.000230	0	1.000000	0.000253	**0.000253**	**0.019268**
heartRate	0	1.000000	0.000235	0	1.000000	0.000235	**-0.000235**	**0.014509**
glucose	0	1.000000	0.214020	0	1.000000	0.214020	**0.214020**	**0.000006**

* The mean absolute difference between $\hat{\boldsymbol{\beta}}^*$ and $\hat{\boldsymbol{\beta}}$ is 0.106259, between $\boldsymbol{\beta}^{(0)}$ is 0.189551, between $\hat{\boldsymbol{\beta}}_{\mathrm{R}}$ is 0.334929, between $\hat{\boldsymbol{\beta}}_{\lvert\hat{\mathrm{M}}_{\mathrm{y}\lvert 0},\hat{\mathrm{M}}_{\mathrm{y}\lvert 1}}$ is 0.190810, and between $\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$ is 0.288274.

Table 3.65: Comparisons of different estimates of **CASE 3** under the setting of resampled data with a size of $\mathfrak{m} = 1170$

	$\hat{\boldsymbol{\beta}}$			$\boldsymbol{\beta}^{(0)}$			$\hat{\boldsymbol{\beta}}_{\mathrm{R}}$		
	Estimates	p values	$\lvert\hat{\beta}_j^* - (\hat{\beta}_j)\rvert$	Estimates	p values	$\lvert\hat{\beta}_j^* - (\beta_j)^{(0)}\rvert$	Estimates	p values	$\lvert\hat{\beta}_j^* - ((\hat{\beta}_j)_{\mathrm{R}}\rvert$
Intercept	**-2.264560**	< **0.000001**	0.054080	**-1.684130**	< **0.000001**	0.634510	**0.000840**	< **0.000001**	2.319480
sex	0.326881	0.114527	0.108002	**0.000097**	< **0.000001**	0.434786	**0.574608**	< **0.000001**	0.139725
$\mathrm{education}_1$	0	1.000000	0.140794	**0.000029**	< **0.000001**	0.140765	**0.682697**	< **0.000001**	0.541903
$\mathrm{education}_2$	0	1.000000	0	**-0.000029**	< **0.000001**	0.000029	**0.974995**	< **0.000001**	0.974995
$\mathrm{education}_3$	0.196783	0.408666	0.196783	0	1.000000	0	0	1.000000	0
currentSmoker	0.173442	0.512922	0.173442	**0.000020**	< **0.000001**	0.000020	0	1.000000	0
BPMeds	0	1.000000	0	0	1.000000	0	0	1.000000	0
prevalentStroke	-0.909521	0.363250	1.634708	0	1.000000	0.725187	0	1.000000	0.725187
prevalentHyp	0.266303	0.272021	0.014622	**0.001953**	< **0.000001**	0.249728	**-0.001543**	< **0.000001**	0.253224
diabetes	0	1.000000	0	0	1.000000	0	0	1.000000	0
cigsPerDay	0.226704	0.082813	0.048641	**0.001378**	< **0.000001**	0.273967	**-0.000219**	< **0.000001**	0.275564
age	**0.504207**	< **0.000001**	0.002514	**0.048939**	< **0.000001**	0.452755	**0.452599**	< **0.000001**	0.049094
totChol	**0.200689**	**0.035571**	0.197345	**0.003424**	< **0.000001**	0.000080	0	1.000000	0.003344
sysBP	**0.304353**	**0.003379**	0.024524	**0.049088**	< **0.000001**	0.279789	**-0.049510**	< **0.000001**	0.378387
diaBP	**-0.000155**	< **0.000001**	0.000134	**0.004475**	< **0.000001**	0.004495	**-0.172146**	< **0.000001**	0.172125
BMI	**0.000513**	< **0.000001**	0.000260	**0.002413**	< **0.000001**	0.002160	**-0.014739**	< **0.000001**	0.014992
heartRate	**-0.000242**	< **0.000001**	0.000006	0	1.000000	0.000235	**-0.000023**	< **0.000001**	0.000212
glucose	**0.000863**	< **0.000001**	0.213157	**0.000605**	< **0.000001**	0.213415	**0.000825**	< **0.000001**	0.213195

	$\hat{\boldsymbol{\beta}}_{\lvert\hat{\mathrm{M}}_{\mathrm{y}\lvert 0},\hat{\mathrm{M}}_{\mathrm{y}\lvert 1}}$			$\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$			$\hat{\boldsymbol{\beta}}^*$	
	Estimates	p values	$\lvert\hat{\beta}_j^* - (\hat{\beta}_j)_{\lvert\hat{\mathrm{M}}_{\mathrm{y}\lvert 0},\hat{\mathrm{M}}_{\mathrm{y}\lvert 1}}\rvert$	Estimates	p values	$\lvert\hat{\beta}_j^* - ((\hat{\beta}_j)_{\mathrm{CD}}\rvert$	Estimates	p values
Intercept	**-1.700110**	< **0.000001**	0.618530	**-0.000361**	< **0.000001**	2.318279	**-2.318640**	< **0.000001**
sex	0	1.000000	0.434883	0	1.000000	0.434883	**0.434883**	**0.000437**
$\mathrm{education}_1$	0	1.000000	0.140794	0	1.000000	0.140794	0.140794	0.215942
$\mathrm{education}_2$	0	1.000000	0	0	1.000000	0	0	1.000000
$\mathrm{education}_3$	0	1.000000	0	0	1.000000	0	0	1.000000
currentSmoker	0	1.000000	0	0	1.000000	0	0	1.000000
BPMeds	0	1.000000	0	0	1.000000	0	0	1.000000
prevalentStroke	0	1.000000	0.725187	0	1.000000	0.725187	0.725187	0.194834
prevalentHyp	0	1.000000	0.251681	0	1.000000	0.251681	0.251681	0.100156
diabetes	0	1.000000	0	0	1.000000	0	0	1.000000
cigsPerDay	0	1.000000	0.275345	0	1.000000	0.275345	**0.275345**	< **0.000001**
age	**0.002970**	< **0.000001**	0.498723	**0.009925**	< **0.000001**	0.491768	**0.501693**	< **0.000001**
totChol	**0.000016**	< **0.000001**	0.003328	**0.000099**	< **0.000001**	0.003245	**0.003344**	**0.023402**
sysBP	**0.002629**	< **0.000001**	0.326248	**-0.000032**	< **0.000001**	0.328909	**0.328877**	**0.000004**
diaBP	**0.000039**	< **0.000001**	0.000060	**-0.000244**	< **0.000001**	0.000223	**-0.000021**	**0.000527**
BMI	**0.000010**	< **0.000001**	0.000242	0	1.000000	0.000253	**0.000253**	**0.019268**
heartRate	0	1.000000	0.000235	**-0.000039**	< **0.000001**	0.000196	**-0.000235**	**0.014509**
glucose	0	1.000000	0.214020	**0.000052**	< **0.000001**	0.213968	**0.214020**	**0.000006**

* The mean absolute difference between $\hat{\boldsymbol{\beta}}^*$ and $\hat{\boldsymbol{\beta}}$ is 0.156056, between $\boldsymbol{\beta}^{(0)}$ is 0.189551, between $\hat{\boldsymbol{\beta}}_{\mathrm{R}}$ is 0.336746, between $\hat{\boldsymbol{\beta}}_{\lvert\hat{\mathrm{M}}_{\mathrm{y}\lvert 0},\hat{\mathrm{M}}_{\mathrm{y}\lvert 1}}$ is 0.193849, and between $\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$ is 0.288040.

To conclude, from the analysis of the FHS dataset in this section, we consistently observe that mislabelling can have a substantial impact on the performance of the LR classifier, confirming the findings from the simulation study discussed in Section 3.7.4. Ignoring them can have severe consequences, with the trained classifier performing no better than a random guess. Correcting a subset of imperfect labels without addressing mislabelling during estimation has limited impact on improving the classifier's performance. Particularly, when the size of the resampled data is small, the classifier tends to perform at a level comparable to random guessing. We also made several important findings when comparing our approach with alternative methods. First, by considering resampling and using our proposed method, we can still utilise the corrupted dataset and extract valuable information. Resampling is

practical and requires less time and cost for researchers to scrutinise a subsample of the dataset and the combined datasets. Leveraging the small clean dataset with the unresampled, mislabelled data can improve the classifier's efficiency. Second, although the mislabelling ratios of the resampled dataset are close to those obtained from the full-size training dataset, the method that estimates flipping probabilities by these ratios generates coefficient estimates close to 0, which is less interpretable. This approach treats the class- and feature-dependent label noise as class-dependent only, making the model less credible. Lastly, our approach demonstrates strong performance with low misclassification rates, similar to those of the classifier trained with perfectly labelled data. Our approach is more practical and efficient, saving more time and costs in real-world applications.

4 Online algorithms for streaming data

This chapter centres around the discussion of online algorithms for sparse streaming data. We propose online methods tailored for scrutinising streaming data in the context of sparse GLMs, which include both low-dimensional sparse streaming data and high-dimensional streaming data. The architecture of this chapter is articulated as follows.

Initially, in Section 4.1, we provide a detailed description of the model that we study for this research topic. Commencing with the introduction of the offline penalised likelihood approach, we then offer comprehensive descriptions of our proposed incremental algorithm for sparse low-dimensional streaming data in Section 4.2. Subsequently, we present the procedure for defining the tuning parameter of the penalty function in Section 4.3, which encompasses two distinct algorithms for selecting the tuning parameter, and various simulation studies are executed. The results have demonstrated our proposed method's stability and competitive performance. Specifically, in the simulation study in Section 4.3.4, we compare our method with Luo and Song (2020)'s approach, which disregards variable selections. The comparison demonstrates that their technique experiences convergence issues when applied to sparse datasets, while our method performs more effectively.

Section 4.4 introduces a method that incorporates an iteratively updated penalty term during the estimation process. This method shows comparable performance to the method introduced in Section 4.2, which keeps the penalty term constant, but it demonstrates superiority in feature selection by effectively identifying and excluding insignificant features. As a result, it generates more parsimonious models.

Building on the methodology delineated in Section 4.2 for low-dimensional sparse streaming data, we extend our approach to cater to high-dimensional streaming scenarios. To realise this, we incorporate the Sure Independent Screening (SIS) and Iterative Sure Independent Screening (ISIS) techniques (Fan and Song, 2010, Fan et al., 2009), which we subsequently term as 'Independence Screening' (IS) and 'Iterative Independence Screening' (IIS) respectively. These integrated methods, designed especially for streaming data, are detailed in Sections 4.5 and 4.6. We supplement our explanations with a variety of simulation studies, underscoring the efficacy of our proposed approaches.

Section 4.7 is devoted to the analysis of real-world data, specifically the National Automotive Sampling System Crashworthiness Data, wherein the monthly crash incidents are treated as streaming data. In this section, we perform a comparative evaluation of the performance between our proposed methods described in Section 4.2 and Section 4.4, and the traditional offline estimation method. We also assess the efficiency of the method suggested by Luo and Song (2020), which excludes variable selection procedures, using the same dataset they have analysed. By comparing the outcomes obtained from alternative methods with the performance of our proposed methods, we can confidently conclude that our methods offer advantages in terms of estimation accuracy, model interpretability, and resource efficiency for streaming data analyses.

4.1 Model specification

Streaming data, in our study, means that the data is generated continuously and can be observed sequentially. We have a sample $(X_i,\ y_i)$, $i = 1,\ \cdots,\ N_b$, where $X_i = (x_{1i}, x_{2i}, \cdots, x_{pi})^T$ is a random p-dimensional vector and y_i is the response variable. The subset of observations in a sequence is stored in different batches, denoted as $\mathcal{B}_1 = \{\mathbf{X}_1,\ \mathbf{y}_1\},\ \cdots,\ \mathcal{B}_l = \{\mathbf{X}_l,\ \mathbf{y}_l\},\ \cdots,\ \mathcal{B}_b = \{\mathbf{X}_b,\ \mathbf{y}_b\}$, where $1 \leqslant l \leqslant b$ and b, $b = 2,\ \cdots$, is the number of observed data batches, and $\mathcal{B}_l$ has a size of n_l and the total size of all the observed data is $N_b = \sum_{l=1}^{b} n_l$. $\mathbf{X}_l$ and $\mathbf{y}_l$ are the notations of the observed covariates and the response variables in batch $\mathcal{B}_l$, and $\mathbf{X} = X_1,\ \cdots,\ X_{N_b}$, and $\mathbf{y} = y_1,\ \cdots,\ y_{N_b}$, are the notations of all observations.

In our streaming data analysis, we hold the assumption that the data either follows a consistent distribution or exhibits only minor variations between different distributions. Delving deeper into our methodology, the simulation studies in the subsequent sections generate i.i.d. data from a single distribution, or in cases where they differ, the covariates X_i, $i = 1, \cdots, N_b$, display varied correlations.

Linear models or statistics based on linear functions of data can be easily updated or decomposed in the context of streaming data (Luo and Song, 2020). While we discuss a more general case the MLE, with a non-linear function of data. We assume that the observations are independently generated according to the distribution $f(y; \mathbf{X}, \boldsymbol{\beta}_0, \phi_0)$, where $\boldsymbol{\beta}_0 \in \mathcal{R}^p$ represents the true parameter of interest and ϕ_0 is the

true value of a dispersion parameter. With reference to Jorgensen (1997), the GLM has the log-likelihood function in the form of an exponential dispersion model,

$$\ell(\boldsymbol{\beta},\ \phi) = \sum_{i=1}^{N_b} \left[\frac{y_i\theta_i - b(\theta_i)}{a_i(\phi)} + c(y_i, \phi) \right],$$

where $\boldsymbol{\beta}$ is the unknown parameter to be estimated, ϕ is the dispersion parameter, θ_i is the canonical parameter for observation i and $\theta_i = g(\mu_i) = X_i\boldsymbol{\beta}$, where $g(\cdot)$ is a known link function and $\mu_i = E(y_i|X_i)$ refers to the mean parameter. Additionally, $a_i(\cdot)$, $b_i(\cdot)$ and $c_i(\cdot)$ are all known functions and notably, $c(y_i, \phi)$ is a known function that only depends on y_i and ϕ.

Under the assumption of sparsity, we partition $\boldsymbol{\beta}$ into $\boldsymbol{\beta_1} \in \mathbb{R}^d$ and $\boldsymbol{\beta_2} \in \mathbb{R}^{p-d}$, where $\boldsymbol{\beta_1}$ consists of d, $1 \leqslant d \leqslant p$, non-zero coefficients and $\boldsymbol{\beta_2}$ consists of zero coefficients. Our goal is to obtain the MLE of the unknown p-dimensional parameter vector $\boldsymbol{\beta}$ by fitting a penalised log-likelihood function with the following form,

$$\mathcal{L}(\mathbf{X},\ \mathbf{y};\ \boldsymbol{\beta}) = \ell(\mathbf{X},\ \mathbf{y};\ \boldsymbol{\beta}) - \mathbf{P}_\lambda(\|\boldsymbol{\beta}\|_1),$$

where $\ell(\mathbf{X},\ \mathbf{y};\ \boldsymbol{\beta})$ is a log-likelihood function and $\ell(X,\ \mathbf{y};\ \boldsymbol{\beta}) = \sum_{i=1}^{N_b} \ell(y_i, X_i; \boldsymbol{\beta})$. $\mathbf{P}_\lambda(\cdot)$ is a penalty function and $\|\boldsymbol{\beta}\|_1 = \sum_{j=1}^{p} |\beta_j|$, and λ is a threshold of the penalty function. The above equation is abbreviated as equation (4.1) for simplicity.

$$\mathcal{L}(\boldsymbol{\beta}) = \ell(\boldsymbol{\beta}) - \mathbf{P}_\lambda(\|\boldsymbol{\beta}\|_1). \tag{4.1}$$

In the following context of this study, we use the tilde symbol to denote a quantity obtained by the incremental algorithm. For example, $\tilde{\boldsymbol{\beta}}_b$ is the renewable MLE of $\boldsymbol{\beta}$, obtained using the incremental algorithm with new arrivals in $\mathcal{B}_b$ and historical statistical inferences. The hat symbol denotes estimates obtained using the traditional offline maximum likelihood estimation method. We use $\hat{\boldsymbol{\beta}}_b$ to denote the MLE of $\boldsymbol{\beta}$ obtained from the data only in batch $\mathcal{B}_b$, and $\hat{\boldsymbol{\beta}}^*_b$ to denote the MLE obtained from all observations in the b collected data batches $\mathcal{B}_l,\ l = 1, \ldots, b$, using the traditional offline maximum likelihood estimation method. In the following context, the terms "renewable estimates" and "renewable MLE" are used mutually.

4.2 Methodology

In this section, we provide a detailed description of our method for studying streaming data generated from sparse models. We begin by explaining the offline algorithm for penalised maximum likelihood estimation. Next, we introduce our incremental algorithms with penalty functions. Finally, we present a simulation study that compares the performance of our incremental method with the offline method in two examples. The first example involves a different full-size training dataset, while the second example considers varying sizes of streaming data.

4.2.1 Offline penalised maximum likelihood estimation algorithm

The offline algorithm analyses all observed data all at once. For streaming data observed in different batches, $\mathcal{B}_l$, $l = 1, \cdots, b$, and $b \geqslant 2$, it retains each observation and waits for the collection of N_b observations to be completed before starting the analysis. However, this approach requires large amounts of storage space, and when new data joins, the analysis must be restarted using previously trained data with the new data, which can be both computationally time-consuming and expensive.

We employ the Newton-Raphson method along with LQA to derive the MLE of $\boldsymbol{\beta}$ in (4.1). In detail, at each iteration k, where $k = 1, \cdots$, the estimator takes the form of a non-zero d-dimensional vector, where $1 \leqslant d \leqslant p$. Specifically,

$$\hat{\boldsymbol{\beta}}^{(k)} = \hat{\boldsymbol{\beta}}^{(k-1)} - \Big(\frac{\partial^2 \mathcal{L}(\boldsymbol{\beta})}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^T}\Big)^{-1}\Big|_{\boldsymbol{\beta}=\hat{\boldsymbol{\beta}}^{(k-1)}} \frac{\partial \mathcal{L}(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}}\Big|_{\boldsymbol{\beta}=\hat{\boldsymbol{\beta}}^{(k-1)}},$$

which can be expressed in the form

$$\hat{\boldsymbol{\beta}}^{(k)} = \hat{\boldsymbol{\beta}}^{(k-1)} - \Big\{\mathbf{H}(\hat{\boldsymbol{\beta}}^{(k-1)}) - N_b \mathbf{P}''_\lambda(\|\hat{\boldsymbol{\beta}}^{(k-1)}\|_1)\Big\}^{-1} \Big[\mathbf{S}(\hat{\boldsymbol{\beta}}^{(k-1)}) - N_b \mathbf{P}'_\lambda(\|\hat{\boldsymbol{\beta}}^{(k-1)}\|_1)\Big], \tag{4.2}$$

when the iterations converge, we get the MLE of $\boldsymbol{\beta}$ denoted as $\hat{\boldsymbol{\beta}}^*_b$. Here, $\mathbf{S}(\boldsymbol{\beta})$ and $\mathbf{H}(\boldsymbol{\beta})$ are respectively the unit score function and the Hessian matrix of the log-likelihood function $\ell(\boldsymbol{\beta})$ in (4.1). Specifically,

$$\mathbf{S}(\boldsymbol{\beta}) = \frac{\partial \ell(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}}, \quad \mathbf{H}(\boldsymbol{\beta}) = \nabla \mathbf{S}(\boldsymbol{\beta}) = \frac{\partial^2 \ell(\boldsymbol{\beta})}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^T}.$$

Furthermore, employing the LQA, the second derivative of the penalty function can

be expressed as:

$$\mathbf{P}''_\lambda(\|\hat{\boldsymbol{\beta}}^{(k-1)}\|_1) = \Sigma_\lambda(\hat{\boldsymbol{\beta}}^{(k-1)}) = \mathrm{diag}\{p'_\lambda(|\hat{\beta}_1^{(k-1)}|)/|\hat{\beta}_1^{(k-1)}|, \cdots, p'_\lambda(|\hat{\beta}_d^{(k-1)}|)/|\hat{\beta}_d^{(k-1)}|\}, \tag{4.3}$$

and

$$\mathbf{P}'_\lambda(\|\hat{\boldsymbol{\beta}}^{(k-1)}\|_1) = \mathbf{P}''_\lambda(\|\hat{\boldsymbol{\beta}}^{(k-1)}\|_1)\hat{\boldsymbol{\beta}}^{(k-1)}.$$

The derivation process, incorporating both the LQA and Newton-Raphson algorithm, serves to approximate the log-likelihood function and the penalty term. This approach is consistent with the methodology elaborated upon for the *first-step estimation* in Section A. Consequently, the iterative estimation equation presented as (A.5) mirrors that of (4.2) introduced above in this section.

For the non-zero estimates of $\boldsymbol{\beta}$, we have the estimated variance matrix as

$$\widehat{\mathrm{cov}}((\hat{\boldsymbol{\beta}}_1^*)_b) = -\left[\mathbf{H}((\hat{\boldsymbol{\beta}}_1^*)_b) - N_b\Sigma_\lambda((\hat{\boldsymbol{\beta}}_1^*)_b)\right]^{-1}, \tag{4.4}$$

where $(\hat{\boldsymbol{\beta}}_1^*)_b$ consisting of the non-zero components of $\hat{\boldsymbol{\beta}}^*$ that is obtained by the offline algorithm and $\Sigma_\lambda(\cdot)$ is the second derivative of the penalty function $\mathbf{P}_\lambda(\cdot)$ and has the form shown in (4.3).

4.2.2 Incremental algorithm for penalised maximum likelihood estimation

In this section, we outline our approach to the renewable estimation method. Specifically, we approximate the penalty function anchored around the previously established renewable MLE. This ensures that the resultant penalty function retains its consistency across all stages of the iterative estimation process. In a subsequent section, we unveil an alternative model that offers dynamic updates to the penalty function throughout the estimation iterations, further detailed in Section 4.4.1.

Building on the offline algorithm of the previous section, we now present the incremental algorithm and start with the simplest case, where the data in $\mathcal{B}_{1,2}$ are observed sequentially. When the first batch of data is collected at $\mathcal{B}_1$, the calculation algorithm is the same as the offline algorithm, and the MLE of $\boldsymbol{\beta}$ is denoted as $\hat{\boldsymbol{\beta}}_1$.

The estimation of the dispersion parameter ϕ is

$$\hat{\phi}_1 = \frac{1}{n_1 - p} \sum_{i \in \mathcal{B}_1} \frac{(y_i - \hat{\mu}_i)^2}{v(\hat{\mu}_i)},$$

where $\hat{\mu}_i$ is the estimation of the mean parameter μ_i and $v(\hat{\mu}_i),\ i \in \mathcal{B}_1$, is the estimation of the variance parameter.

When the data in $\mathcal{B}_2$ is collected, we first refer to the offline algorithm with the iterative estimation process shown in (4.2). At the k-th iteration, where $k = 1, \cdots$, the offline estimation of $\boldsymbol{\beta}$ is expressed as:

$$\hat{\boldsymbol{\beta}}_2^{(k)} = \hat{\boldsymbol{\beta}}_2^{(k-1)} - \left\{ \tilde{\mathbf{H}}_2(\hat{\boldsymbol{\beta}}_2^{(k-1)}) - N_2 \mathbf{P}''_{\lambda_2^*}(\|\hat{\boldsymbol{\beta}}_2^{(k-1)}\|_1) \right\}^{-1} \left[\tilde{\mathbf{S}}_2(\hat{\boldsymbol{\beta}}_2^{(k-1)}) - N_2 \mathbf{P}'_{\lambda_2^*}(\|\hat{\boldsymbol{\beta}}_2^{(k-1)}\|_1) \right],$$

where $\tilde{\mathbf{S}}_2(\boldsymbol{\beta}) = \mathbf{S}_1(\mathcal{B}_1; \boldsymbol{\beta}) + \mathbf{S}_2(\mathcal{B}_2; \boldsymbol{\beta})$, $\tilde{\mathbf{H}}_2(\boldsymbol{\beta}) = \mathbf{H}_1(\mathcal{B}_1; \boldsymbol{\beta}) + \mathbf{H}_2(\mathcal{B}_2; \boldsymbol{\beta})$ and $\mathbf{P}_\lambda(\cdot)$ is the penalty function with the tuning parameter λ. When the iterations converge, we obtain the MLE of $\boldsymbol{\beta}$ using the offline algorithm and it is denoted as $\hat{\boldsymbol{\beta}}_2^*$.

By virtue of the properties of MLE, the $\hat{\boldsymbol{\beta}}_1$ derived from the data in $\mathcal{B}_1$ satisfies the unit score equation for the penalised log-likelihood function, as articulated in equation (4.1):

$$\mathbf{S}_1(\mathcal{B}_1; \hat{\boldsymbol{\beta}}_1) - n_1 \mathbf{P}'_{\lambda_1}(\|\hat{\boldsymbol{\beta}}_1\|_1) = \mathbf{0}. \tag{4.5}$$

Similarly, $\hat{\boldsymbol{\beta}}_2^*$ trained by the data in both $\mathcal{B}_1$ and $\mathcal{B}_2$ also satisfies the condition, as given by

$$\mathbf{S}_1(\mathcal{B}_1; \hat{\boldsymbol{\beta}}_2^*) + \mathbf{S}_2(\mathcal{B}_2; \hat{\boldsymbol{\beta}}_2^*) - N_2 \mathbf{P}'_{\lambda_2^*}(\|\hat{\boldsymbol{\beta}}_2^*\|_1) = \mathbf{0}. \tag{4.6}$$

Drawing from the discussion in Section 4.1, we assume that the data in $\mathcal{B}_1$ and $\mathcal{B}_2$ either share the same distribution or come from distributions with slight variations. As a result, $\hat{\boldsymbol{\beta}}_1$, derived from dataset $\mathcal{B}_1$, is expected to be closely aligned with $\hat{\boldsymbol{\beta}}_2^*$, derived from both $\mathcal{B}_1$ and $\mathcal{B}_2$. Given this alignment and the smooth characteristics of the estimation function around $\hat{\boldsymbol{\beta}}_1$, we adopt the first-order Taylor approximation. This yields the expression for the first and last terms in (4.6):

$$\mathbf{S}_1(\mathcal{B}_1; \hat{\boldsymbol{\beta}}_2^*) = \mathbf{S}_1(\mathcal{B}_1; \hat{\boldsymbol{\beta}}_1) + \mathbf{H}_1(\mathcal{B}_1; \hat{\boldsymbol{\beta}}_1)(\hat{\boldsymbol{\beta}}_2^* - \hat{\boldsymbol{\beta}}_1) + \mathcal{O}_p(\|\hat{\boldsymbol{\beta}}_2^* - \hat{\boldsymbol{\beta}}_1\|^2), \tag{4.7}$$

$$\begin{aligned} N_2 \mathbf{P}'_{\lambda_2^*}(\|\hat{\boldsymbol{\beta}}_2^*\|_1) = {} & N_2 \mathbf{P}'_{\lambda_2^*}(\|\hat{\boldsymbol{\beta}}_1\|_1) + N_2 \mathbf{P}''_{\lambda_2^*}(\|\hat{\boldsymbol{\beta}}_1\|_1)(\hat{\boldsymbol{\beta}}_2^* - \hat{\boldsymbol{\beta}}_1) \\ & + \mathcal{O}_p(\|\hat{\boldsymbol{\beta}}_2^* - \hat{\boldsymbol{\beta}}_1\|^2), \end{aligned} \tag{4.8}$$

where $\mathcal{O}_p(\cdot)$ is the error term and can be asymptotically ignored, and λ_1 represents the tuning parameter of the penalty function derived from the previous data batch, whereas λ_2^* corresponds to the tuning parameter obtained from the combined data of the two batches.

Given the proximity of the streaming data in the two data batches, and of $\hat{\boldsymbol{\beta}}_1$ to $\hat{\boldsymbol{\beta}}_2^*$, equation (4.8) can be expressed as:

$$N_2 \mathbf{P}'_{\lambda_2^*}(\|\hat{\boldsymbol{\beta}}_2^*\|_1) \approx n_1 \mathbf{P}'_{\lambda_1}(\|\hat{\boldsymbol{\beta}}_1\|_1) + n_2 \mathbf{P}'_{\lambda_2^*}(\|\hat{\boldsymbol{\beta}}_1\|_1) + N_2 \mathbf{P}''_{\lambda_2^*}(\|\hat{\boldsymbol{\beta}}_1\|_1)(\hat{\boldsymbol{\beta}}_2^* - \hat{\boldsymbol{\beta}}_1). \tag{4.9}$$

Taking into account both Equation (4.7) and Equation (4.9), and after discarding the error terms from (4.7) as well as the zero term evident in (4.5), Equation (4.6) can be reformulated as:

$$\left[\mathbf{H}_1(\mathcal{B}_1; \hat{\boldsymbol{\beta}}_1) - N_2 \mathbf{P}''_{\lambda_2}(\|\hat{\boldsymbol{\beta}}_1\|_1)\right](\tilde{\boldsymbol{\beta}}_2 - \hat{\boldsymbol{\beta}}_1) + \mathbf{S}_2(\mathcal{B}_2; \tilde{\boldsymbol{\beta}}_2) - n_2 \mathbf{P}'_{\lambda_2}(\|\hat{\boldsymbol{\beta}}_1\|_1) = \mathbf{0}, \tag{4.10}$$

where $\tilde{\boldsymbol{\beta}}_2$ can be viewed as a second-order asymptotic approximation for $\hat{\boldsymbol{\beta}}_2^*$ and λ_2, obtained from $\mathcal{B}_2$, serves as an approximation for λ_2^*.

Using the unit score function given by equation (4.10), we can deploy the Newton-Raphson algorithm to determine its solution. To be sepcific, at the k-th, $k = 1, \cdots$, iteration, the estimator then takes the form as

$$\begin{aligned}
\tilde{\boldsymbol{\beta}}_2^{(k)} \;=\; \tilde{\boldsymbol{\beta}}_2^{(k-1)} &- \left[\mathbf{H}_1(\mathcal{B}_1; \hat{\boldsymbol{\beta}}_1) + \mathbf{H}_2(\mathcal{B}_2; \tilde{\boldsymbol{\beta}}_2^{(k-1)}) - N_2 \mathbf{P}''_{\lambda_2}(\|\hat{\boldsymbol{\beta}}_1\|_1)\right]^{-1} \\
&\times \Big\{\left[\mathbf{H}_1(\mathcal{B}_1; \hat{\boldsymbol{\beta}}_1) - N_2 \mathbf{P}''_{\lambda_2}(\|\hat{\boldsymbol{\beta}}_1\|_1)\right](\tilde{\boldsymbol{\beta}}_2^{(k-1)} - \hat{\boldsymbol{\beta}}_1) \\
&\quad + \mathbf{S}_2(\mathcal{B}_2; \tilde{\boldsymbol{\beta}}_2^{(k-1)}) - n_2 \mathbf{P}'_{\lambda_2}(\|\hat{\boldsymbol{\beta}}_1\|_1)\Big\},
\end{aligned}$$

and $\mathbf{H}_2(\mathcal{B}_2; \tilde{\boldsymbol{\beta}}_2^{(k-1)})$ can be replaced by $\mathbf{H}_2(\mathcal{B}_2; \hat{\boldsymbol{\beta}}_1)$. In this way, it avoids the need to update the Hessian matrix at each iteration and speed up the iterations (Luo and Song, 2020). Therefore, the estimator at the k-th, $k = 1, \cdots$, iteration has the following form:

$$\begin{aligned}
\tilde{\boldsymbol{\beta}}_2^{(k)} = \; \tilde{\boldsymbol{\beta}}_2^{(k-1)} &- \left[\sum_{l=1}^{2} \mathbf{H}_l(\hat{\boldsymbol{\beta}}_1) - N_2 \mathbf{P}''_{\lambda_2}(\|\hat{\boldsymbol{\beta}}_1\|_1)\right]^{-1} \\
&\times \Big\{\left[\mathbf{H}_1(\hat{\boldsymbol{\beta}}_1) - N_2 \mathbf{P}''_{\lambda_2}(\|\hat{\boldsymbol{\beta}}_1\|_1)\right](\tilde{\boldsymbol{\beta}}_2^{(k-1)} - \hat{\boldsymbol{\beta}}_1)
\end{aligned}$$

$$+\mathbf{S}_2(\tilde{\boldsymbol{\beta}}_2^{(k-1)}) - n_2\mathbf{P}'_{\lambda_2}(\|\hat{\boldsymbol{\beta}}_1\|_1)\Big\}. \tag{4.11}$$

After the iterations in (4.11) converge, the estimate of $\boldsymbol{\beta}$ is obtained as $\tilde{\boldsymbol{\beta}}_2$. We refer to this estimate as the renewable MLE.

To update the previous statistics, we renew $\hat{\boldsymbol{\beta}}_1$ by $\tilde{\boldsymbol{\beta}}_2$. Concurrently, the Hessian matrix is updated as $\tilde{\mathbf{H}}_2$, where $\tilde{\mathbf{H}}_2(\tilde{\boldsymbol{\beta}}_2) = \mathbf{H}_1(\hat{\boldsymbol{\beta}}_1) + \mathbf{H}_2(\tilde{\boldsymbol{\beta}}_2)$. In addition, the unbiased estimate of the variance parameter ϕ in the GLM is updated by computing $\tilde{\phi}_2$ (Luo and Song, 2020)

$$\tilde{\phi}_2 = \frac{n_1 - p}{N_2 - p}\hat{\phi}_1 + \frac{n_2 - p}{N_2 - p}\hat{\phi}_2,$$
$$\hat{\phi}_2 = \frac{1}{n_2 - p}\sum_{i\in\mathcal{B}_2}\frac{(y_i - \hat{\mu}_i)^2}{v(\hat{\mu}_i)},$$

where $N_2 = \sum_{l=1}^{2} n_l$.

The estimated covariance matrix for the non-zero estimates of $\boldsymbol{\beta}$ is

$$\widetilde{\mathrm{cov}}((\tilde{\boldsymbol{\beta}}_1)_2) = -\tilde{\phi}_2\left[\tilde{\mathbf{H}}_2((\tilde{\boldsymbol{\beta}}_1)_2) - N_2\Sigma_{\lambda_2}((\tilde{\boldsymbol{\beta}}_1)_2)\right]^{-1},$$

where $(\tilde{\boldsymbol{\beta}}_1)_2$ is the renewable estimate consisting of non-zero elements of $\tilde{\boldsymbol{\beta}}_2$ and $\Sigma_\lambda(\cdot)$ refers to the second derivative of the penalty function with the explicit form given in (4.3).

From our outlined estimation process, it is evident that only specific historical statistical quantities, namely $\hat{\boldsymbol{\beta}}_1$ and $\mathbf{H}_1(\mathcal{B}_1)$, are utilised to update the statistics as new data from $\mathcal{B}_2$ becomes available. Notably, the details of observations in $\mathcal{B}_1$ are not required.

It is important to highlight that the tuning parameter, initially denoted as λ_1 and derived from the streaming data in $\mathcal{B}_1$, is continuously updated with the introduction of new data from $\mathcal{B}_2$, and is subsequently denoted as λ_2. A comprehensive approach to updating the tuning parameter with each fresh data batch is provided in Section 4.3.

We now describe a more general situation where streaming data comes in two or more data batches, and the last observed data batch is denoted as $\mathcal{B}_b$ for $b \geqslant 2$. The previous statistical inferences have been obtained and saved using the previous

batches of data and are denoted with the subscript $b-1$. When new data in $\mathcal{B}_b$ is collected, at the k—th, $k = 1, \cdots$, iteration, the estimator takes the form of:

$$\begin{aligned}\tilde{\boldsymbol{\beta}}_b^{(k)} = \tilde{\boldsymbol{\beta}}_b^{(k-1)} &- \left[\tilde{\mathbf{H}}_b(\tilde{\boldsymbol{\beta}}_{b-1}) - N_b \mathbf{P}''_{\lambda_b}(\|\tilde{\boldsymbol{\beta}}_{b-1}\|_1)\right]^{-1} \\ &\times \left\{\left[\tilde{\mathbf{H}}_{b-1}(\tilde{\boldsymbol{\beta}}_{b-1}) - N_b \mathbf{P}''_{\lambda_b}(\|\tilde{\boldsymbol{\beta}}_{b-1}\|_1)\right](\tilde{\boldsymbol{\beta}}_b^{(k-1)} - \tilde{\boldsymbol{\beta}}_{b-1})\right. \\ &\left. + \mathbf{S}_b(\tilde{\boldsymbol{\beta}}_b^{(k-1)}) - n_b \mathbf{P}'_{\lambda_b}(\|\tilde{\boldsymbol{\beta}}_{b-1}\|_1)\right\}, \end{aligned} \tag{4.12}$$

where λ_b corresponds to the tuning parameter obtained from data in $\mathcal{B}_b$.

In this more general situation, the renewable MLE of $\boldsymbol{\beta}$ is denoted as $\tilde{\boldsymbol{\beta}}_b$, obtained by iterating the online algorithm of (4.12) until convergence is achieved. Then, $\tilde{\boldsymbol{\beta}}_{b-1}$ and $\tilde{\mathbf{H}}_{b-1}(\tilde{\boldsymbol{\beta}}_{b-1})$ are updated using $\tilde{\boldsymbol{\beta}}_b$ and $\tilde{\mathbf{H}}_b(\tilde{\boldsymbol{\beta}}_b)$ respectively. To update the unbiased estimates of ϕ with the data in $\mathcal{B}_b$, we use the following equation (Luo and Song, 2020):

$$\begin{aligned}\tilde{\phi}_b &= \frac{N_{b-1} - p}{N_b - p}\tilde{\phi}_{b-1} + \frac{n_b - p}{N_b - p}\hat{\phi}_b, \\ \hat{\phi}_b &= \frac{1}{n_b - p}\sum_{i \in \mathcal{B}_b} \frac{(y_i - \hat{\mu}_i)^2}{v(\hat{\mu}_i)}, \end{aligned} \tag{4.13}$$

where $N_b = \sum_{l=1}^{b} n_l$, $\hat{\mu}_i$ is the estimation of the mean parameter and $\mu_i = E(y_i|X_i)$ and $v(\hat{\mu}_i)$, $i \in \mathcal{B}_b$, is the estimation of the variance parameter. The estimated covariance matrix for the non-zero estimates of $\boldsymbol{\beta}$ is

$$\widetilde{\text{cov}}((\tilde{\boldsymbol{\beta}}_1)_b) = -\tilde{\phi}_b \left[\tilde{\mathbf{H}}_b((\tilde{\boldsymbol{\beta}}_1)_b) - N_b \Sigma_{\lambda_b}((\tilde{\boldsymbol{\beta}}_1)_b)\right]^{-1}, \tag{4.14}$$

where $(\tilde{\boldsymbol{\beta}}_1)_b$ is the renewable estimate consisting of non-zero elements of $\tilde{\boldsymbol{\beta}}_b$.

As is evident from our incremental algorithm, only the crucial statistics are retained and used in the estimation process instead of using and storing the detailed streaming data. These include the renewable MLE of $\boldsymbol{\beta}$, the corresponding renewed Hessian matrix, the renewable estimate of dispersion parameter ϕ, and the cumulative number of the trained observations. This distinction underscores the efficiency of our online algorithm in conserving storage space, offering a computational advantage over offline methods. Furthermore, the online inferences, particularly online confidence intervals, can be shaped using the estimated covariance matrix (4.14).

4.2.3 Simulation study setup

Model specification: For this study, we conduct 100 separate and independent runs of an experiment, with a full-size dataset consisting of N_b independent observations used for training in each run. In addition, we use an independent set of 500 observations for testing in each run. Each data point in the dataset consists of p covariates, $X_i = (x_{1i}, x_{2i}, \cdots, x_{pi})^T$. The first covariate, x_{1i}, represents the intercept term and is fixed at 1. The remaining $p-1$ covariates, denoted as $x_{[2:p]i}$, are generated from a multivariate normal distribution with mean vector $\mathbf{0}$ and a $(p-1)\times(p-1)$ compound symmetry covariance matrix, denoted as Σ_{p-1}, with correlation parameter ρ. We specify the covariance matrix Σ_{p-1} for each simulation using different examples. Unless otherwise noted, we assume that the data for each experiment are identically independently distributed (i.i.d.).

In this study, the logistic model, a specific type of GLM, is the focus of our simulation study. We assume that the binary response variable $y_i \in \{0, 1\}$, $i = 1, \cdots, N_b$, and

$$\pi_y(X_i) = P(y_i = 1|X_i), \quad \text{logit}\{\pi_y(X_i)\} = X_i^T\boldsymbol{\beta},$$

where $\boldsymbol{\beta}$ is a p-dimensional parameter vector to be estimated. We continue with the assumption of sparsity in $\boldsymbol{\beta}$ that the parameter vector $\boldsymbol{\beta}$ consists of d non-zero coefficients, where $1 \leqslant d \leqslant p$, and $p - d$ zero coefficients. The mean parameter μ_i is equal to $\pi_y(X_i)$ in the logistic model and the dispersion parameter $\phi = 1$ in the logistic model. For the i-th observation, the log-likelihood function is

$$\ell(\boldsymbol{\beta}) = y_i \text{logit}(\pi_y(X_i)) - \log(1 - \pi_y(X_i)).$$

We incorporate the SCAD penalty function into our model, and the penalty term $\mathbf{P}_\lambda(\cdot)$ in Equation (4.1) is defined by Equation (2.4), with a fixed value of $a = 3.7$. The process of determining a desirable value for λ is discussed in Section 4.3.

Additionally, we adopt the method outlined in (Rodríguez, 2007, p. 10) to handle the problem of zero-one data that often occurs when analysing binary response variables. To achieve this, for every simulation, we transform the response variable, and the estimator of $\boldsymbol{\beta}$ is adjusted accordingly, but this transformation is only performed

for the first iteration. The transformations are as follows

$$\begin{aligned} \hat{\beta}_1^{(0)} &= (XX^T)^{-1}Xy^*, \\ \text{where,} \quad y_i^* &= \log(\frac{y_i + 0.5}{1.5 - y_i}), \quad i = 1, \cdots, N_b. \end{aligned} \tag{4.15}$$

During the penalised iterative estimation process for the unknown parameter $\boldsymbol{\beta}$, if the absolute value of an estimate at the current iteration satisfies the condition

$$|\hat{\beta}_j| < \sigma_0, \tag{4.16}$$

for $j = 1, \cdots, p$, we set the estimate to zero. This rule applies to both the offline method, denoted by $\hat{\boldsymbol{\beta}}_b^*$, and the online method, denoted by $\tilde{\boldsymbol{\beta}}_b$, where $b = 1, \cdots$ represents the batch number of the observed streaming data.

Evaluation metrics for simulation study: The criteria and metrics employed to assess performance in this chapter's simulation study are delineated in Section 3.2. In addition to procuring the Estimated Standard Errors (ESEs) within the context of *Online algorithms for Streaming Data*, we utilise (4.4) to compute the ESEs of $\hat{\boldsymbol{\beta}}$ when implementing the offline method. In contrast, for the renewable MLE, we resort to (4.14) to obtain the ESEs. Detailed explanations are provided in the corresponding sections.

4.2.4 Simulation study

In this section, we have two simulation examples designed to investigate the performance of our renewable estimation method under different experimental settings. Moreover, we compare our method with the traditional offline method to evaluate its effectiveness. The first example studies the performance of renewable estimates using different sizes of streaming data. The second example compares the renewable estimates obtained with various full-size training datasets.

For both examples, we set the parameter vector in the model to be

$$\boldsymbol{\beta} = (0, \beta_2, 0, \beta_4, \beta_5, \ 0, \cdots, \ 0)_{p\times 1}^T,$$

$$\beta_2 = 1, \ \beta_4 = -2, \ \beta_5 = 1.5.$$

The value of p is different between the following two examples and is specified separately in each case.

Example 1: Different batch sizes n_b. In the first example, we study datasets with $p = 12$ covariates, where $x_{[2:12]i}$ are distributed as $\mathcal{N}(0,\ \Sigma_{11})$, $i = 1,\ \cdots,\ N_b$, and Σ_{11} is a compound symmetry covariate matrix with correlation $\rho = 0.5$. We use streaming data of size n_b, which remain fixed for each run, and we perform trials with various batch sizes of $n_b = 200,\ 500,\ 1000$. Our method is tested on two different full-size training datasets with $N_b = 2000$ and $N_b = 10000$ in the first example. The numerical results are presented in Table 4.1 to Table 4.4. When $n_b = N_b$, the estimates are obtained using the offline method.

Table 4.1 reports the AMRSEs and MMRSEs of the 100 estimates. The table shows that, for each case with the same full-size training datasets, there is no pronounced difference between the $\tilde{\boldsymbol{\beta}}$s trained using different streaming data sizes, especially when the batch sizes are large ($n_b = 500,\ 1000$). However, when the data batch size is small ($n_b = 200$), the renewable estimates are slightly less accurate than those trained using larger streaming data sizes ($n_b = 500,\ 1000$). In each case of the same training data, the $\tilde{\boldsymbol{\beta}}$s exhibit very close performance with the $\hat{\boldsymbol{\beta}}$s, and our method produces accurate estimates for $\boldsymbol{\beta}$.

Table 4.2 displays SDs, ESE_m, and ESE_o for the non-zero coefficients of $\boldsymbol{\beta}$, which are calculated using two different methods for estimating variance. Specifically, we used (4.4) to estimate the variance of the offline estimates and (4.14) to estimate the variance of the online estimates. The close agreement between the estimated values and the true values, observed across different cases with varying streaming data sizes and full training data sizes, indicates that both variance formulas are effective in practice. The 100 estimates of the standard errors are used to construct confidence ellipsoids for the non-zero coefficients of $\boldsymbol{\beta}$ in each run, and the resulting values are recorded in Table 4.3.

Table 4.3 shows the numbers of incorrect non-zero and zero estimates, as well as the CPs for the non-zero and zero coefficients. The CP values for the non-zero and zero coefficients of $\boldsymbol{\beta}$ are similar between the renewable and offline estimates when trained on the same dataset. In this example, the offline estimates exhibit the highest CP values of 1 for zero coefficients in both cases with different sizes of training data, and the renewable estimates have values that are very close across different cases. On

the other hand, the CP values for the non-zero coefficients of $\boldsymbol{\beta}$ are similar between the renewable and offline estimates when the size of the streaming data is sufficiently large (when $n_b = 500$ or 1000). Specifically, in the case where the full-size training data is $N_b = 2000$ and the streaming data has a size of $n_b = 500$, the renewable method achieves slightly higher CP than the offline method. We also observe that in all cases with 100 replications, both our penalised renewable estimation method and the penalised offline method using SCAD do not miss any important features. It should be noted that both methods produce some incorrect non-zero estimates for the zero coefficients of $\boldsymbol{\beta}$. The offline method has the least number of incorrect estimates when $N_b = 10000$, while there are slightly more incorrect non-zero estimates of $\hat{\boldsymbol{\beta}}$s than $\tilde{\boldsymbol{\beta}}$s for different trials with various streaming data sizes when $N_b = 2000$. We provide a detailed discussion of the variable selection results, including the effects of the search intervals, search step sizes, and validation data sizes, in Section 4.3. Despite these factors, the renewable method demonstrates stable performance for variable selection, regardless of the size of the streaming data.

Table 4.4 records the excess risks of the classifiers, the computing time, and the memory utilisations. The memory utilisation shows the memory required for one experiment including 100 independent runs during its execution. From the results of excess risks of the renewable classifiers trained with different sizes of streaming data, we observe that the classifiers perform similarly. It is also noticeable that all the renewable classifiers exhibit similar performance to the Naive Bayes classifier, as evidenced by the excess risk values close to or equal to 1 in the table. It is clear that when trained by the same datasets, the renewable classifiers have almost no difference in efficiency from the traditional classifier of the offline method. Thus, the renewable estimation has obvious advantages over the offline method in terms of computing efficiency, requiring less time and memory to train on the same dataset.

To conclude, our method performs similarly regardless of batch size. When the streaming data is sufficiently large, the size of stream batches has less influence on the estimation outcomes. Our classifier performs similarly to the Bayes classifier and is almost as efficient as the offline classifier. It should be noted that our method requires much fewer computing resources and time to conduct the analysis, as revealed by the recorded results, and also requires less space to store the details of observations.

.

Table 4.1: AMRSEs and MMRSEs (in brackets) of $\tilde{\beta}_b$s and $\hat{\beta}^*$s

n_b	200	500	1000	N_b
$N_b = 2000$	0.780(0.398)	0.476(0.365)	0.617(0.452)	0.434(0.216)
$N_b = 10000$	0.131(0.064)	0.075(0.055)	0.081(0.058)	0.062(0.046)

* All values recorded in this table multiplied by 10^{-3} are true values.

* For the cases where $n_b = 200, 500, 1000$, the renewable estimation method produces estimates $\tilde{\beta}_b$s, while for the case where $n_b = N_b$, the offline method produces estimates $\hat{\beta}^*$s.

Table 4.2: SDs and ESEs for non-zero $\beta_{12\times1}$ coefficient estimates

	n_b	200	500	1000	N_b
β_2	$N_b = 2000$	0.0825(0.0798,0.1183)	0.0752(0.0799,0.1185)	0.0766(0.0802,0.1189)	0.0655(0.0784,0.1163)
	$N_b = 10000$	0.0359(0.0349,0.0518)	0.0294(0.0350,0.0519)	0.0287(0.0351,0.0521)	0.0290(0.0349,0.0517)
β_4	$N_b = 2000$	0.1321(0.0995,0.1475)	0.1204(0.1008,0.1494)	0.1218(0.1016,0.1507)	0.1139(0.0994,0.1474)
	$N_b = 10000$	0.0611(0.0440,0.0652)	0.0550(0.0442,0.0656)	0.0542(0.0444,0.0658)	0.0506(0.0441,0.0653)
β_5	$N_b = 2000$	0.1068(0.0883,0.1309)	0.0935(0.0889,0.131)	0.1072(0.0894,0.1325)	0.1152(0.0874,0.1296)
	$N_b = 10000$	0.0531(0.0388,0.0576)	0.0517(0.0391,0.0579)	0.0486(0.0392,0.0581)	0.0506(0.0390,0.0578)

* For each cell, the value outside the brackets represents SD, while the first and second numbers inside the brackets represent ESE_m and ESE_o, respectively.

* For the cases where $n_b = 200, 500, 1000$, the ESEs of the renewable estimates $\tilde{\beta}_b$s are obtained using Equation (4.14), while for the cases where $n_b = N_b$, the ESEs of $\hat{\beta}^*$ are obtained using equation Equation (4.4).

Table 4.3: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for β

	n_b	200	500	1000	N_b
$N_b = 2000$	non0 coeff.	0(0.900000)	0(0.936667)	0(0.923333)	0(0.926667)
	0 coeff.	3.18(0.976667)	2.70(0.980000)	2.84(0.968889)	3.83(1.000000)
$N_b = 10000$	non0 coeff.	0(0.856667)	0(0.910000)	0(0.903333)	0(0.926667)
	0 coeff.	2.11(0.992222)	1.75(0.995556)	1.78(0.992222)	0.95(1.000000)

* $\beta_{12\times1}$ cosists of 3 non-zero coefficients.

Table 4.4: Excess risks of $C_{\tilde{\beta}}$s and $C_{\hat{\beta}^*}$s, and computing time and memory utilisations (in brackets)

n_b	200	500	1000	N_b
$N_b = 2000$	0.998162 (00:02:45; 0.33 GB)	1.000000 (00:10:14; 0.48 GB)	1.000000 (00:33:48; 1.08 GB)	1.000260 (01:43:13; 3.37 GB)
$N_b = 10000$	0.998061 (00:06:01; 0.39 GB)	0.998413 (00:13:44; 0.56 GB)	0.998501 (00:36:23; 1.17 GB)	0.998413 (39:24:16; 75.50 GB)

* For the 100 replications, when $N_b = 2000$, the misclassification rate of the Bayes classifier is 22.940%; When $N_b = 10000$, the misclassification rate of the Bayes classifier is 22.652%.
* Excess risks are reported outside of the brackets, the first value in the brackets represents the computing time, and the second value in the brackets represents the memory utilisations.
* We implemented each experiment with 100 independent trials using C++ and ran them on 2 *Intel Xeon 6138 CPUs*. For each experiment, we requested 35 cores.

Example 2: Different training data sizes. In the second example, we set $p = 150$, which is more than 10 times the number of insignificant features in **Example 1**. We test the data with covariates $x_{j_1 i}$ and $x_{j_2 i}$, $i = 1, \cdots, N_b$, $j_{1,2} = 2, \cdots, 150$, that have a correlation of $\rho^{|j_1 - j_2|}$, where $\rho = 0.5$. For the same trail, the streaming data is observed in sequential batches of the same size n_b, where we consider two different batch sizes $n_b = \{200, 300\}$. We record the performance of renewable estimations for b-th data batches with $b = 2,\ 6,\ 20$, as well as the offline estimation analyses the same training data as the renewable estimation method in $b = 20$ stream batches.

The performance of methods can be affected by the tuning parameter search, as explained in detail in Section 4.3. In this example, to find desirable values for the tuning parameter λ in the renewable estimation process, we set the search step size for all trials as 0.15. For the case where $n_b = 200$, the search interval is $[0.05, 0.5]$, and for the case where $n_b = 300$, the search interval is $[0.05, 0.65]$. For the offline method, we have a search interval of $[0.035, 0.2]$ and a search step size of 0.2 for the case where $n_b = 200$, and a search interval of $[0.05, 0.2]$ and a search step size of 0.35 for the case where $n_b = 300$. The numerical results for this example are reported in Table 4.5 to Table 4.8.

The AMRSEs and MMRSEs of the renewable estimates, $\tilde{\boldsymbol{\beta}}_b$s, $b = 2, 6, 20$, and the offline estimates, $\hat{\boldsymbol{\beta}}^*_{20}$s, are recorded in Table 4.5. As expected, the performance of our renewable estimation method improves as more information is collected. Specifically, when $b = 2$, the accuracy of renewable estimates is the lowest. With the observation of more stream batches, the performance improves and approaches $\hat{\boldsymbol{\beta}}^*_{20}$. Notably, the

most accurate renewable estimates are obtained with $\tilde{\boldsymbol{\beta}}_{20}$s in each case.

Table 4.2 provides a comparison of the ESEs obtained using equations (4.14) and (4.4) with the true values, represented by SDs. In general, are found to be close to the true values. Specifically, for smaller batch sizes ($n_b = 200$) and fewer data batches studied ($b = 2$), the ESEs from the renewable estimation deviate more from the SDs. Thus as more data batches are observed, the ESEs get closer to the SDs.

Table 4.7 presents the performance of variable selection, including the numbers of incorrectly estimated non-zero and zero coefficients and the CPs for non-zero and zero coefficients. The results indicate that our method identifies almost all significant variables with only one missed significant variable in each experiment, across all cases with 100 repetitions. With a fixed size of streaming data, we find that the number of incorrect zero estimates remains the same while the CPs of non-zero coefficients are similar, regardless of the number of batches of streaming data. From the online penalised method explained in Section 4.2.2, the penalty term is approximated and related to the renewable estimates from the previous data batch. The zero-absorbing state of SCAD (Fan and Lv, 2010) can result in missed important features in subsequent data batches.

Under the fair settings for the tuning parameter search introduced above, the renewable estimation method produces estimates with far fewer incorrect non-zero estimates compared to the estimates from the offline method. In consequence, the number of incorrect non-zero estimates decreases as more stream data batches are studied. However, the $\hat{\boldsymbol{\beta}}^*_{20}$ estimates assign non-zero values to more than half of the zero coefficients, indicating worse interpretability of the selected models compared to our renewable estimation method. This suggests that under the same settings for computing and the same threshold for assigning a 0 value to insignificant features, the offline method performs less satisfactorily than our proposed method in terms of the interpretability of the selected model. Our online algorithm has an advantage over the offline algorithm, as it can scrutinise important features and screen out non-important ones more often with streaming data.

It can be seen in Table 4.7 that the CPs for both non-zero and zero coefficients are similar across the different numbers of stream data batches used to train the renewable estimates, with CP values for zero coefficients close to 1, indicating accurate estimation of the insignificant variables. Notably, when $n_b = 300$, the CP values for

non-zero coefficients from the renewable estimates is significantly higher than the CPs from the offline method. This suggests that our method is better at identifying the significant variables in this example.

Table 4.8 presents the excess risks of the classifiers from our renewable estimation method and the offline method, as well as the computing time and memory usage for each experiment with 100 replications. $C_{\hat{\boldsymbol{\beta}}^*_{20}}$s from the offline method have the highest efficiency, performing slightly better than the Bayes classifier. On the other hand, the efficiency of renewable classifiers becomes similar to that of the traditional classifiers trained using the same training dataset when $b = 20$. Also, the renewable classifiers approach the Bayes classifier in terms of performance, with their excess risks increasing and approaching 1. Nonetheless, the offline algorithm requires significantly more time and computing resources compared to renewable algorithms.

In conclusion, the efficiency of our renewable classifiers improves with the collection of more stream data batches, allowing for more accurate and timely classification. The results of this example demonstrate that our renewable estimation method achieves higher accuracy as more data batches are analysed, due to the continual refinement of the parameter estimates with the streaming data. Furthermore, even in the presence of sparse models with many zero coefficients, our incremental algorithm can select all the important features and outperforms the offline method in scrutinising unimportant features. This is consistent with **Example 1** in this section and shows that our renewable method outperforms the offline method, requiring much less time and storage space while being more efficient in computation.

Table 4.5: AMRSEs and MMRSEs (in brackets) of $\tilde{\boldsymbol{\beta}}_b$s and $\hat{\boldsymbol{\beta}}^*$s

	$\tilde{\boldsymbol{\beta}}_2$	$\tilde{\boldsymbol{\beta}}_6$	$\tilde{\boldsymbol{\beta}}_{20}$	$\hat{\boldsymbol{\beta}}^*_{20}$
$n_b = 200$	1.844(0.278)	1.291(0.085)	0.581(0.027)	0.022(0.017)
$n_b = 300$	1.044(0.102)	0.599(0.023)	0.320(0.007)	0.025(0.015)

* All values recorded in this table multiplied by 10^{-3} are true values.

Table 4.6: SDs and ESEs for non-zero $\boldsymbol{\beta}_{150\times 1}$ coefficient estimates

$n_b = 200$	$\tilde{\boldsymbol{\beta}}_2$	$\tilde{\boldsymbol{\beta}}_6$	$\tilde{\boldsymbol{\beta}}_{20}$	$\hat{\boldsymbol{\beta}}^*_{20}$
β_2	0.4544(0.1439,0.2133)	0.2041(0.0845,0.1253)	0.0897(0.0465,0.0690)	0.0673(0.0471,0.0698)
β_3	0.4796(0.1988,0.2948)	0.2694(0.1167,0.1731)	0.1689(0.0651,0.0966)	0.1394(0.0661,0.0980)
β_5	0.3003(0.1755,0.2602)	0.2166(0.1034,0.1533)	0.1242(0.0573,0.0850)	0.1092(0.0575,0.0852)
$n_b = 300$	$\tilde{\boldsymbol{\beta}}_2$	$\tilde{\boldsymbol{\beta}}_6$	$\tilde{\boldsymbol{\beta}}_{20}$	$\hat{\boldsymbol{\beta}}^*_{20}$
β_2	0.1436(0.1255,0.1860)	0.0739(0.0719,0.1065)	0.0501(0.0392,0.0581)	0.0605(0.0385,0.0570)
β_3	0.2212(0.1777,0.2634)	0.1375(0.1006,0.1492)	0.0659(0.0551,0.0817)	0.1300(0.0541,0.0802)
β_5	0.1813(0.1533,0.2273)	0.1159(0.0876,0.1299)	0.0599(0.0479,0.0711)	0.1111(0.0469,0.0696)

* For each cell, the value outside the brackets represents SD, while the first and second numbers inside the brackets represent $\mathrm{ESE_m}$ and $\mathrm{ESE_o}$, respectively.

Table 4.7: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$

$n_b = 200$	$\tilde{\boldsymbol{\beta}}_2$	$\tilde{\boldsymbol{\beta}}_6$	$\tilde{\boldsymbol{\beta}}_{20}$	$\hat{\boldsymbol{\beta}}^*_{20}$
non0 coeff.	0.01(0.570000)	0.01(0.563333)	0.01(0.580000)	0(0.78)
0 coeff.	19.59(0.999728)	8.46(0.999592)	2.51(0.999660)	116.97(0.999796)
$n_b = 300$	$\tilde{\boldsymbol{\beta}}_2$	$\tilde{\boldsymbol{\beta}}_6$	$\tilde{\boldsymbol{\beta}}_{20}$	$\hat{\boldsymbol{\beta}}^*_{20}$
non0 coeff.	0.01(0.806667)	0.01(0.813333)	0.01(0.816667)	0(0.693333)
0 coeff.	26.02(0.995850)	12.68(0.992653)	4.83(0.992925)	97.53(1.00000)

* $\boldsymbol{\beta}_{150\times 1}$ cosists of 3 non-zero coefficients.

Table 4.8: Excess risks of $\mathrm{C}_{\tilde{\boldsymbol{\beta}}_b}$s and $\mathrm{C}_{\hat{\boldsymbol{\beta}}^*_b}$s, and computing time and memory utilisations (in brackets)

	$\mathrm{ER}(\mathrm{C}_{\tilde{\boldsymbol{\beta}}_2})$	$\mathrm{ER}(\mathrm{C}_{\tilde{\boldsymbol{\beta}}_6})$	$\mathrm{ER}(\mathrm{C}_{\tilde{\boldsymbol{\beta}}_{20}})$	$\mathrm{ER}(\mathrm{C}_{\hat{\boldsymbol{\beta}}^*_{20}})$
$n_b = 200$	0.886210 (00:40:19; 3.42 GB)	0.909440 (00:53:47; 3.43 GB)	0.973003 (01:30:02; 3.44 GB)	1.00104 (41:15:07; 17.31 GB)
	$\mathrm{ER}(\mathrm{C}_{\tilde{\boldsymbol{\beta}}_2})$	$\mathrm{ER}(\mathrm{C}_{\tilde{\boldsymbol{\beta}}_6})$	$\mathrm{ER}(\mathrm{C}_{\tilde{\boldsymbol{\beta}}_{20}})$	$\mathrm{ER}(\mathrm{C}_{\hat{\boldsymbol{\beta}}^*_{20}})$
$n_b = 300$	0.949502 (01:56:28; 3.44 GB)	0.957117 (40:32:10; 3.44 GB)	0.985448 (40:25:45; 3.45 GB)	1.00069 (45:00:32; 33.92 GB)

* For the 100 replications, in the case where $n_b = 200$, the misclassification rate of the Bayes classifier is 22.994%, while in the case where $n_b = 300$, the misclassification rate of the Bayes classifier is 23.160%.

* We implemented each experiment with 100 independent trials using C++ and ran them on 2 *Intel Xeon 6138 CPUs*. For each experiment of renewable estimations, we requested 35 cores, while for each experiment of the offline method, we requested 40 cores.

4.3 Determining the tuning parameter for the penalty function in renewable estimation

In this section, we focus on determining an optimal value for the tuning parameter of the penalty function in the incremental algorithm. To accomplish this, we introduce the modified Leave-P-Out Cross-Validation (mLpO CV) algorithm, which aids in the selection of the appropriate tuning parameter. Subsequently, we present simulated examples in separate sections to illustrate the effectiveness of our approach.

In the first section, we introduce two algorithms with different searching processes to find the tuning parameters of the penalty function in the model and design a simulation study to compare their performance. In the second section, we compare our incremental algorithm which uses the SCAD penalty function with the method that lacks variable selection (Luo and Song, 2020) and provide a simulated example. In the last section, we discuss the use of validation datasets of varying sizes in our incremental method to find the tuning parameter for the penalty function in the model. We demonstrate the stable performance of our method to find tuning parameters by examining its performance across different sizes of observations removed from the training dataset.

4.3.1 Modified Leave-P-Out Cross-Validation (mLPOCV) for streaming data

In our study, Modified Leave-P-Out Cross-Validation (mLPOCV) process is used to find a tuning parameter of the penalty function for the penalised methods.

Similar to classical Leave-P-Out Cross-Validation (LPOCV), we randomly split the newly arrived data in B_b, $b = 1, \cdots$, into two datasets: one for training and another for validation. The validation dataset has a size of n_{cv}, where $1 \leqslant n_{cv} < n_b$. However, unlike LPOCV where the model is trained only once and tested on all validation data together, in our modified approach we validate the model on one pair of data at a time. After the k-th iteration of validation, where $k = 1, \cdots, n_{cv} - 1$, the validated pair of data (X_k, y_k) is added to the training dataset for the $(k+1)$-th validation, except when $k = n_{cv}$. This modified approach enables us to train the model multiple times, which can lead to improved accuracy in selecting the tuning parameter.

4.3.2 Two algorithms of search processes

In this section, we present two algorithms for searching the optimal tuning parameter of the penalty function in mLPOCV, and compare their performance through a simulation study.

Before commencing the computation, the search interval, denoted as $[a, b]$, which contains the desired values for the tuning parameter, and the step size for searching are predetermined.

Algorithm 1: One-Way search. We initiate the search process from one end of the specified interval and set the maximum value of (a, b) as the initial testing value, denoted as $\lambda_1 = b$. The step size for this One-Way search is represented as s_1, where $0 < s_1 < 1.00$. Additionally, we test three other neighbouring values, λ_2, λ_3, and λ_4, in addition to λ_1. The values for these neighbouring testing points are given as follows:

$$\begin{aligned} \lambda_2 &= \lambda_1 - \frac{(\lambda_1 - a)}{3.00} \times s_1, \\ \lambda_3 &= \lambda_1 - \frac{(\lambda_1 - a)}{3.00} \times 2.00 \times s_1, \\ \lambda_4 &= \lambda_1 - (\lambda_1 - a) \times s_1, \end{aligned} \tag{4.17}$$

and $a \leqslant \lambda_4 < \lambda_3 < \lambda_2 < \lambda_1 \leqslant b$.

After performing mLPOCV to validate the model using the testing point λ_ν for $\nu = 1, 2, 3, 4$, we obtain the sum of the log-likelihoods, $\sum_{k=1}^{n_{cv}} \log L(X_k, y_k, (\tilde{\boldsymbol{\beta}}_b)_k | \lambda_\nu)$, where $(\tilde{\boldsymbol{\beta}}_b)_k$ is the renewable estimate of $\boldsymbol{\beta}$ obtained from the training dataset at the k-th iteration in mLPOCV. During the search process, we update λ_1 with its neighbouring value that has the maximum sum of log-likelihoods, and then we renew the other testing points using the formula (4.17). We continue the search until either the sum of the log-likelihoods associated with λ_1 is maximised, or until $\lambda_4 = a$. The value of the tuning parameter that generates the maximum sum of log-likelihoods is selected.

Algorithm 2: Two-way search. The Two-Way search method is another way to find the optimal value of the tuning parameter λ in a given interval $[a, b]$. This method starts at two ends of the interval and searches towards the middle, using a predetermined step size which is denoted as s_2, $0 < s_2 < 1$, and two candidates to be tested denoted as λ_{T_1} and λ_{T_2}, such that $a \leqslant \lambda_{T_1} < \lambda_{T_2} \leqslant b$.

In detail, following the mLPOCV process, the Two-Way search proceeds as follows: the n_{cv} data are first validated using λ_{T_1} and λ_{T_2} as testing points. The sums of the conditional log-likelihoods are then obtained.

If $\sum_{k=1}^{n_{cv}} L(X_k, y_k, (\tilde{\boldsymbol{\beta}}_b)_k|\lambda_{T_1}) > \sum_{k=1}^{n_{cv}} L(X_k, y_k, (\tilde{\boldsymbol{\beta}}_b)_k|\lambda_{T_2})$, we update the testing points as

$$\begin{cases} \lambda_{T_1} = \lambda_{T_1}, \\ \lambda_{T_2} = \lambda_{T_2} - (\lambda_{T_2} - \lambda_{T_1}) \times s_2. \end{cases} \tag{4.18}$$

On the other hand, if $\sum_{k=1}^{n_{cv}} L(X_k, y_k, (\tilde{\boldsymbol{\beta}}_b)_k|\lambda_{T_1}) < \sum_{k=1}^{n_{cv}} L(X_k, y_k, (\tilde{\boldsymbol{\beta}}_b)_k|\lambda_{T_2})$, we update the testing points as

$$\begin{cases} \lambda_{T_1} = \lambda_{T_1} + (\lambda_{T_2} - \lambda_{T_1}) \times s_2, \\ \lambda_{T_2} = \lambda_{T_2}. \end{cases} \tag{4.19}$$

Or if the sums of likelihoods are equal and $(\lambda_{T_2} - \lambda_{T_1}) > \epsilon$, where $\epsilon > 0$ is a predetermined small constant, we have:

$$\begin{cases} \lambda_{T_1} = \lambda_{T_1} + (\lambda_{T_2} - \lambda_{T_1}) \times s_2, \\ \lambda_{T_2} = \lambda_{T_2} - (\lambda_{T_2} - \lambda_{T_1}) \times s_2. \end{cases} \tag{4.20}$$

The search continues until $(\lambda_{T_2} - \lambda_{T_1}) < \epsilon$. The optimal value of the tuning parameter λ is set to the testing point with the larger sum of the conditional log-likelihoods.

It can be seen from the algorithm that testing every point in the given interval may not be feasible, and may result in missing the optimal tuning parameter, particularly when the interval $[a, b]$ is large and so is the step size s_2. However, a smaller step size may lead to longer search times. Therefore, to balance estimation accuracy and computing efficiency, it is necessary to carefully select an appropriate search interval and step size. To mitigate this issue, we propose reducing the step size s_2 by half when either $|\lambda_{T_1} - \lambda_{T_2}| > 0.5$ or $|\lambda_{T_1} - \lambda_{T_2}| < 0.1$ during the Two-Way search process. This can prevent stepping too far and potentially missing desirable values, especially in cases where the search interval is large or the step size is too large.

4.3.3 Simulation study: Comparison of two search methods for tuning parameter selection

In the following context, we design three simulated examples and compare the performance of the two algorithms introduced above. We denote the renewable estimate from One-Way search as $\tilde{\boldsymbol{\beta}}_{\text{One-Way}}$ and the one from Two-Way search as $\tilde{\boldsymbol{\beta}}_{\text{Two-Way}}$. Note that the simulation study in this section exclusively considers online methods, and as such, the term "renewable" is occasionally omitted when referring to the estimates produced by our proposed online method.

The first example compares the performance of the algorithms under different sizes of search steps, the second example studies the effect of different search intervals, and the last example demonstrates the performance of the two search algorithms with varying sizes of streaming data. For the three examples, the true coefficient vector $\boldsymbol{\beta}$ is set as follows:

$$\boldsymbol{\beta} = (0, \beta_2, 0, \cdots, \beta_5, 0, \cdots, 0)^{\mathsf{T}}_{p\times 1}, \ \beta_2 = 1, \ \beta_5 = -1.1,$$

and $p = 15$. For all simulation experiments in this section, we use full-size training datasets consisting of $N_b = 1000$ data. The correlation between $x_{j_1 i}$ and $x_{j_2 i}$, $i = 1, \cdots, N_b$, is $\rho^{|j_1 - j_2|}$, where $j_{1,2} = 2, \cdots, 15$, and $\rho = 0.5$. The size of the validation dataset is fixed as $n_{cv} = 20$ for all tuning parameter search processes in this section.

Example 1: In the first example, we have the streaming data batches with a size of $n_b = 200$ and the given search interval is $[0.005, 0.65]$. We test different step sizes $s_{1,2} = 0.15, 0.25, 0.5$, for the two search methods separately. Table 4.9- Table 4.12 record the results for **Example 1**.

Based on Table 4.9, we can see that $\tilde{\boldsymbol{\beta}}_{\text{Two-Way}}$s consistently outperform the $\tilde{\boldsymbol{\beta}}_{\text{One-Way}}$s in terms of both lower AMRSEs and MMRSEs across all three step sizes tested. This suggests that the Two-Way search method is generally more effective in finding the optimal tuning parameter values for λ than the One-Way search method. With the same search interval, it is expected that the Two-Way search is more likely to miss a desirable value for the tuning parameter and produce less accurate estimates when using a large step size. In spite of this, the pattern of the One-Way search is difficult to identify as the step size changes. With the same search interval, when using a larger step size for searching, it is expected that the Two-Way search method can be

more likely to miss the optimal tuning parameter value, resulting in less accurate estimates. However, it is difficult to identify the pattern of the One-Way search as the step size changes.

The ESEs and the true values (SDs) of the estimates from Equation (4.14) are recorded in Table 4.10. It consistently shows the trend observed in the mean relative squared errors in Table 4.9, that the more accurate renewable estimates, $\tilde{\boldsymbol{\beta}}_{\text{Two-Way}}$s, also have more accurate ESEs. In contrast, the ESEs of $\tilde{\boldsymbol{\beta}}_{\text{One-Way}}$s are not close to the SDs.

Table 4.11 displays the CPs for the coefficients of $\boldsymbol{\beta}$ based on the confidence ellipsoids constructed using the ESEs of both the One-Way and Two-Way search methods, along with the variable selection performance. Even though the table shows that both methods do not miss any significant feature in all experiments, the One-Way search method produces the estimates with lower accuracy, as shown by the CPs, which are either equal to 0 or close to 0 across all cases, for the non-zero coefficients of $\boldsymbol{\beta}$. On the other hand, the Two-Way search method yields higher CPs for the non-zero coefficients under the given significance level of $\alpha = 0.05$. For zero coefficients, the One-Way search method achieves slightly higher CPs than the Two-Way search method and fewer incorrect numbers of non-zero estimates. In consideration of this, we note that the estimates values of the zero coefficients from the Two-Way search method are small and have less impact. Overall, considering the estimation accuracy of the significant features, which is a crucial aspect of the estimation, the One-Way search method fails to work well.

Table 4.12 presents the computing time of the experiments each with 100 independent replications, and the excess risks of the renewable classifiers from both search methods to find a tuning parameter. We observe that as the step size increases from 0.15 to 0.25 in the Two-Way search method, the computing time decreases as expected. However, when the step size is set to a large value of $s_2 = 0.5$, the experiment takes the most time. We assume that this is because setting the step size too large can result in missing the optimal value and requiring more time to converge to the best value, which in turn necessitates more iterations for estimation due to less satisfactory tuning parameters being introduced into the model. In this example, both renewable classifiers $C_{\tilde{\boldsymbol{\beta}}_{\text{One-Way}}}$ and $C_{\tilde{\boldsymbol{\beta}}_{\text{Two-Way}}}$ have efficiencies close to that of the Bayes classifier across different cases. In consequence, based on the estimation accuracy shown in

the tables above, the One-Way search method is not recommended.

Table 4.9: AMRSEs and MMRSEs (in brackets) of $\tilde{\boldsymbol{\beta}}_{\text{One-Way}}$s and $\tilde{\boldsymbol{\beta}}_{\text{Two-Way}}$s

$s_{1,2}$	0.15	0.25	0.50
$\tilde{\boldsymbol{\beta}}_{\text{One-Way}}$	25.160(24.164)	25.296(24.347)	19.727(22.505)
$\tilde{\boldsymbol{\beta}}_{\text{Two-Way}}$	3.563(1.807)	3.701(1.952)	4.233(2.290)

* All values recorded in this table are multiplied by 10^{-3} and represent true values.

Table 4.10: SDs and ESEs for non-zero $\boldsymbol{\beta}$ coefficient estimates

	$s_{1,2}$	0.15	0.25	0.50
$\tilde{\boldsymbol{\beta}}_{One-Way}$	β_2	0.9159(0.0654,0.0969)	0.9177(0.0654,0.0969)	0.8714(0.0657,0.0975)
	β_5	0.9648(0.0659,0.0977)	0.9671(0.0659,0.0977)	0.9212(0.0661,0.0980)
$\tilde{\boldsymbol{\beta}}_{Two-Way}$	β_2	0.0955(0.0921,0.1365)	0.0965(0.0919,0.1363)	0.1012(0.0923,0.1368)
	β_5	0.1255(0.0976,0.1448)	0.1176(0.0973,0.1443)	0.1252(0.0977,0.1449)

* For each cell, the value outside the brackets represents SD, while the first and second numbers inside the brackets represent ESE_m and ESE_o, respectively.

Table 4.11: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$

	$s_{1,2}$	0.15	0.25	0.50
$\tilde{\boldsymbol{\beta}}_{One-Way}$	non0 coeff.	0(0.000000)	0(0.00000)	0(0.150000)
	0 coeff.	3.11(1.000000)	3.12(1.000000)	4.05(0.996923)
$\tilde{\boldsymbol{\beta}}_{Two-Way}$	non0 coeff.	0(0.825000)	0(0.810000)	0(0.785000)
	0 coeff.	9.56(0.985385)	9.61(0.983846)	9.69(0.983077)

* $\boldsymbol{\beta}_{15\times1}$ cosists of 2 non-zero coefficients.

Table 4.12: Excess risks of $\text{C}_{\tilde{\boldsymbol{\beta}}_{\text{One-Way}}}$s and $\text{C}_{\tilde{\boldsymbol{\beta}}_{\text{Two-Way}}}$s, and computing time (in brackets)

$s_{1,2}$	0.15	0.25	0.50
$\text{ER}(\text{C}_{\tilde{\boldsymbol{\beta}}_{\text{One-Way}}})$	0.985373(00:02:56)	0.984956(00:01:55)	0.991539(00:01:35)
$\text{ER}(\text{C}_{\tilde{\boldsymbol{\beta}}_{\text{Two-Way}}})$	0.991186(00:03:38)	0.991116(00:02:09)	0.962937(00:08:41)

* For the 100 replications, the misclassification rate of the Bayes classifier is 27.89% across all cases.

* We implemented each experiment with 100 independent trials using C++ and ran them on 2 *Intel Xeon 6138 CPUs*. For each experiment of renewable estimations, we requested 35 cores.

Example 2: The simulation results for **Example 2** are presented in Tables 4.13–4.15, where we test the two search methods with different search intervals of length $[0.005,\ 0.65]$, $[0.01,\ 0.35]$, and $[0.01,\ 0.25]$, while fixing the step size as $s_{1,2} = 0.15$ and the stream batch size as $n_b = 200$.

Table 4.13 displays the AMRSEs and MMRSEs of the renewable estimates. It is evident that $\tilde{\boldsymbol{\beta}}_{\text{Two-Way}}$s are much more accurate than $\tilde{\boldsymbol{\beta}}_{\text{One-Way}}$s in all cases. The trend in this example for both methods is that smaller intervals lead to more accurate estimates. This can be explained by the algorithms of both search methods, where using reasonable search steps, but large predetermined intervals can still result in testing points that vary too much and miss the desired values of the tuning parameters.

The ESEs and SDs of the renewable estimates obtained using both search methods are presented in Table C.1, which has been included in Appendix C.1. The table shows that more accurate estimates correspond to ESEs that are closer to the true values of SDs, and this trend is similar to what was observed in Table 4.10 of **Example 1** above. Hence, we omit the details here.

Table 4.14 displays the performance of two search methods in terms of variable selection and the corresponding CPs for the coefficients of $\boldsymbol{\beta}$, obtained using varying search intervals denoted by $[a, b]$. Both methods successfully identify all important features, and decreasing search interval widths lead to increased CPs for non-zero coefficients. However, the Two-Way search method achieves much higher CPs than the One-Way search method, which remains consistently low. Specifically, when $[a, b] = [0.005, 0.65]$, the One-Way search method generates a CP of 0. We also observe that decreasing search interval width led to a larger number of incorrect non-zero estimates for $\tilde{\boldsymbol{\beta}}_{One-Way}$s. The Two-Way search method shows greater stability in the number of incorrect non-zero estimates as the search interval decreases, in comparison to the One-Way search method.

Table 4.15 indicates that both classifiers, $C_{\tilde{\boldsymbol{\beta}}_{One-Way}}$s and $C_{\tilde{\boldsymbol{\beta}}_{Two-Way}}$s, perform similarly under the same search interval setting, and both classifiers approach the performance of the Bayes classifier across all cases. As the search interval width decreases, the Two-Way search method requires less computing time, which is expected. It should be noted that the computing time pattern for the One-Way search method is not as clear, as it is more likely to stop at local optimal values that cannot be detected

prior to the search.

Table 4.13: AMRSEs and MMRSEs (in brackets) of $\tilde{\boldsymbol{\beta}}_{\text{One-Way}}$s and $\tilde{\boldsymbol{\beta}}_{\text{Two-Way}}$s

$[a, b]$	$[0.005, 0.65]$	$[0.01, 0.35]$	$[0.01, 0.25]$
$\tilde{\boldsymbol{\beta}}_{One-Way}$	25.160(24.164)	11.825(13.352)	5.698(6.089)
$\tilde{\boldsymbol{\beta}}_{Two-Way}$	3.563(1.807)	2.717(1.459)	1.983(1.334)

* All values recorded in this table multiplied by 10^{-3} are true values.

Table 4.14: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$

	$[a, b]$	$[0.005, 0.65]$	$[0.01, 0.35]$	$[0.01, 0.25]$
$\tilde{\boldsymbol{\beta}}_{One-Way}$	non0 coeff.	0(0.000000)	0(0.175000)	0(0.340000)
	0 coeff.	3.11(1.000000)	5.34(0.997692)	5.87(0.997692)
$\tilde{\boldsymbol{\beta}}_{Two-Way}$	non0 coeff.	0(0.825000)	0(0.830000)	0(0.870000)
	0 coeff.	9.56(0.985385)	9.31(0.983077)	9.51(0.982308)

* $\boldsymbol{\beta}_{15\times 1}$ cosists of 2 non-zero coefficients.

Table 4.15: Excess risks of $C_{\tilde{\boldsymbol{\beta}}_{\text{One-Way}}}$s and $C_{\tilde{\boldsymbol{\beta}}_{\text{Two-Way}}}$s, and computing time (in brackets)

$[a, b]$	$[0.005, 0.65]$	$[0.01, 0.35]$	$[0.01, 0.25]$
$ER(C_{\tilde{\boldsymbol{\beta}}_{One-Way}})$	0.985373(00:02:56)	0.994509(00:05:02)	0.995147(00:04:31)
$ER(C_{\tilde{\boldsymbol{\beta}}_{Two-Way}})$	0.991186(00:03:38)	0.992103(00:03:12)	0.992385(00:03:08)

* For the 100 replications, the misclassification rate of the Bayes classifier is 27.89% across all cases.

* We implemented each experiment with 100 independent trials using C++ and ran them on 2 *Intel Xeon 6138 CPUs*. For each experiment of renewable estimations, we requested 35 cores.

Example 3: The third example in this section examines the performance of the two search methods with different sizes of streaming data batches for the same training dataset. For this example, we use the same training dataset with a size of $N_b = 1000$, but consider streaming data of sizes $n_b = 100$, 200, and 500, which correspond to $b = 10$, 5, and 2 data batches, respectively. The search interval for all experiments is set to $[a, b] = [0.01,\ 0.35]$ with the same step size of $s_{1,2} = 0.15$. The simulation results for **Example 3** are presented in Table 4.16-Table 4.18.

Table 4.16 records the AMRSEs and MMRSEs of the estimates from the different search methods for tuning parameter λ. Consistently with the previous two examples, the results show that $\tilde{\boldsymbol{\beta}}_{\text{One-Way}}$s perform worse than $\tilde{\boldsymbol{\beta}}_{\text{Two-Way}}$s in each case. We also observe that in this example, as the size of the streaming data increases while the number of data batches decreases, which in turn leads to a decrease in the number of tuning parameter searches. This reduction in the tuning parameter search can lead to the selection of an undesirable value for λ in the analysis of the data while having less chance to calibrate it, which can ultimately affect the accuracy of the estimation results. Despite this, the numerical results clearly show that $\tilde{\boldsymbol{\beta}}_{\text{Two-Way}}$s are more stable than $\tilde{\boldsymbol{\beta}}_{\text{One-Way}}$s as the number of the stream data batches changes.

Table C.2 records the ESEs and the corresponding true values (SDs) in Appendix C.1. The performance of variable selection and the CPs for coefficients of $\boldsymbol{\beta}$ are displayed in Table 4.17. Both methods produce an identical number of incorrect zero estimates for the non-zero coefficients of $\boldsymbol{\beta}$ in all cases. Furthermore, as the size of the streaming data, denoted as n_b, increases while the number of stream data batches, denoted as b, decreases, both methods exhibit an enhancement in the CPs for the non-zero coefficients. As is the case in the previous two examples, the CPs for the non-zero coefficients obtained from the One-Way search method are much lower than those obtained from the Two-Way search method. The One-Way search method generates fewer incorrect non-zero estimates for $\boldsymbol{\beta}$ compared to the Two-Way method, while the CPs for zero coefficients obtained from both methods are very close to each other. Therefore, in this example, as the streaming data size increases, the number of streaming data batches decreases, and we observe an increase in the number of incorrect non-zero estimates for both methods.

Table 4.18 presents the computation time and excess risks of the classifiers obtained using two different search methods to find λs. In this example, as the size of the streaming data batches decreases for the same training dataset, both methods require more computation time. Additionally, as observed in the previous examples, both classifiers perform similarly and have nearly the same efficiency as the Bayes classifier in all cases. It is important to note that based on the estimation accuracy displayed in the tables above, the One-Way search method is not recommended.

Table 4.16: AMRSEs and MMRSEs (in brackets) of $\tilde{\boldsymbol{\beta}}_{\text{One-Way}}$s and $\tilde{\boldsymbol{\beta}}_{\text{Two-Way}}$s

b	10	5	2
$\tilde{\boldsymbol{\beta}}_{One-Way}$	6.560(5.948)	11.825(13.352)	35.977(45.433)
$\tilde{\boldsymbol{\beta}}_{Two-Way}$	2.703(1.508)	2.717(1.459)	4.905(1.985)

* All values recorded in this table multiplied by 10^{-3} are true values.

Table 4.17: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$

	b	10	5	2
$\tilde{\boldsymbol{\beta}}_{One-Way}$	non0 coeff.	0.01(0.155000)	0(0.175000)	0(0.200000)
	0 coeff.	4.89(1.000000)	5.34(0.997692)	5.79(0.999231)
$\tilde{\boldsymbol{\beta}}_{Two-Way}$	non0 coeff.	0.01(0.675000)	0(0.830000)	0(0.875000)
	0 coeff.	7.38(0.987692)	9.31(0.983077)	10.12(0.961538)

* $\boldsymbol{\beta}_{15\times 1}$ cosists of 2 non-zero coefficients.

Table 4.18: Excess risks and computing time (in brackets)

b	10	5	2
ER($C_{\tilde{\boldsymbol{\beta}}_{One-Way}}$)	0.996071(00:03:46)	0.994509(00:05:02)	0.965653(00:16:24)
ER($C_{\tilde{\boldsymbol{\beta}}_{Two-Way}}$)	0.989990(00:02:45)	0.992103(00:03:12)	0.986698(00:10:44)

* For the 100 replications, the misclassification rate of the Bayes classifier is 27.89%.

* We implemented each experiment with 100 independent trials using C++ and ran them on 2 *Intel Xeon 6138 CPUs*. For each experiment of renewable estimations, we requested 35 cores.

To conclude, in this subsection, we explain the two algorithms for finding desirable values of the tuning parameter λ in the penalty function of our model. We designed three simulations to investigate the factors related to the width of search intervals, the size of search steps, and the number of stream batches for a fixed training dataset. Our simulations demonstrate that the One-Way search method is more susceptible to varying search step sizes, search intervals, and stream data batches than the Two-Way search method. Although testing every point in the given intervals is not feasible for either method, both may converge to a local optimum and fail to identify the global optimum. The One-Way search method is more prone to stopping at

a local optimum and overlooking better values in the untested region. However, our designed algorithm for the Two-Way search method allows testing more points, which reduces the risk of missing the global optimum. Therefore, we recommend and employ the Two-Way search method in our study to find desirable values for the tuning parameter λ. To balance both the accuracy of estimation and computing efficiency, a fair and large search interval and a fair and small step size should be considered for conducting the analysis.

4.3.4 Simulation study: Comparing the algorithm with SCAD penalty function to the one without penalty function

In this section, we compare our proposed method using the SCAD penalty function (as introduced in Section 4.2.2) with the unpenalised method described by Luo and Song (2020). The following simulation presented in this section demonstrates that our proposed method using the SCAD penalty function (introduced in Section 4.2.2) achieves higher accuracy estimates than the unpenalised method, particularly for highly sparse models. The effect of the penalty term on the model is especially pronounced when the model sparsity is high.

For the simulation study, we use different datasets with a full size training dataset of $N_b = 1200$ and streaming data in each data batch of size $n_b = 300$. We set

$$\boldsymbol{\beta} = (0, \beta_2, 0, \beta_4, \beta_5, \ 0, \cdots, \ 0)^T_{p\times 1},$$

$$\beta_2 = 1, \ \beta_4 = -2, \ \beta_5 = 1.5.$$

We consider two values of p, $p = 15$ and $p = 50$, seperately. The correlation between $x_{j_1 i}$ and $x_{j_2 i}$, $i = 1, \cdots, N_b$, is determined by $\rho^{|j_1 - j_2|}$, where ρ takes on the values of 0, 0.2, and 0.5, respectively. We denote the renewable estimate from our algorithm as $\tilde{\boldsymbol{\beta}}$ and the renewable estimate from the algorithm not using a penalty function as $\tilde{\boldsymbol{\beta}}_{\lambda=0}$. Table 4.19-Table 4.21 present the numerical results for this example.

Table 4.19 shows the AMRSE and MMRSE of $\tilde{\boldsymbol{\beta}}$s and $\tilde{\boldsymbol{\beta}}_{\lambda=0}$s. It should be noted that in the case where $p = 50$ and $\rho = 0$ or 0.2, the unpenalised method fails to converge during the iterative estimation process in some of the 100 independent replications. In such cases, we set the estimate values to $\mathbf{0}$s. Except for these non-converging

situations, the numerical results reveal that for datasets with the same number of covariates, both $\tilde{\boldsymbol{\beta}}$s and $\tilde{\boldsymbol{\beta}}_{\lambda=0}$s become less accurate as the covariates become more correlated. Nonetheless, in all cases, the estimates obtained using our penalised renewable method are more accurate and less sensitive to changes in the correlation among covariates than those obtained using the unpenalised renewable method. Specifically, in this example, our proposed method yields considerably superior estimates compared to the ones from the unpenalised method.

Table C.3 in Appendix C.1 presents the ESEs of the renewable estimates for the non-zero coefficients, $\boldsymbol{\beta}_1$, along with their true values, SDs. To obtain the ESEs for our penalised method, we use Equation (4.14). For the unpenalised method, we estimate the covariance using the formula $\widetilde{\text{cov}}((\widetilde{\boldsymbol{\beta}_1})_b) = -\left[\widetilde{\mathbf{H}}_b((\widetilde{\boldsymbol{\beta}_1})_b))\right]^{-1}$.

Table 4.20 displays the variable selection and CPs for the coefficients of non-zero and zero coefficients of $\boldsymbol{\beta}$, comparing the performance of the penalised and unpenalised methods. The results show that the SCAD method successfully identifies all the important features for all cases with different correlated covariates when $p = 15$ and misses only a few features when $p = 50$ and $\rho = 0$ or 0.2. Our method also demonstrates good performance in recognizing most of the insignificant features, and its efficiency becomes more apparent as the number of insignificant features increases with $p = 50$. In contrast, the unpenalised method is unable to identify any insignificant feature due to its inability to select variables. While the unpenalised method does not miss any important feature, CPs for the non-zero coefficients from the unpenalised method are noticeably lower than those from our method, which demonstrates that the estimation accuracy is lower than our method. This conclusion is consistent with that from Table 4.19. Moreover, without variable selection, the generated model is difficult to understand.

Table 4.21 presents the excess risks of $C_{\tilde{\boldsymbol{\beta}}}$s and $C_{\tilde{\boldsymbol{\beta}}_{\lambda=0}}$s. Studying the same dataset, our method outperforms the unpenalised method, with a classifier that is closer to the Bayes classifier, as indicated by the numerical results closer to 1 in each case. As well as this, as the model becomes sparser, the difference between the classifiers generated by the two methods becomes larger, highlighting the superior performance of our classifier.

To summarise, the simulation results in this subsection confirm the effectiveness of our penalised incremental algorithm over the non-penalised method in the context

of sparse models. The non-penalised approach suffers from the inability to perform variable selection and may fail to converge when many insignificant features are involved, as demonstrated in this example. In contrast, our penalised renewable estimation approach produces more accurate estimates and performs effective variable selection, resulting in more interpretable models. The use of the SCAD penalty function in our model provides benefits by enabling both estimation and variable selection simultaneously. Thus, the simulation results emphasise the significance of incorporating penalty functions into incremental algorithms for studying streaming data generated from sparse models.

Table 4.19: AMRSEs and MMRSEs (in brackets) of $\tilde{\boldsymbol{\beta}}$s and $\tilde{\boldsymbol{\beta}}_{\lambda=0}$s

ρ		0	0.2	0.5
$p = 15$	$\tilde{\boldsymbol{\beta}}$	0.677(0.606)	0.680(0.564)	0.793(0.692)
	$\tilde{\boldsymbol{\beta}}_{\lambda=0}$	1.136(1.094)	1.129(1.072)	1.431(1.339)
$p = 50$	$\tilde{\boldsymbol{\beta}}$	0.319(0.1640)	0.352(0.1341)	0.656(0.215)
	$\tilde{\boldsymbol{\beta}}_{\lambda=0}$	$> 10^5$(5.015)	$> 10^5$(1.699)	3.963(1.491)

* All values recorded in this table multiplied by 10^{-3} are true values.

* When $p = 50$, the non-penalised algorithm fails to converge during the iterative processes 39 times in the case of $\rho = 0$, 20 times for $\rho = 0.2$, and 6 times for $\rho = 0.5$. When the incremental algorithm fails to converge for the l-th batch, where $1 \leqslant l < b$, our computing algorithm is designed to use $\mathbf{0}$ as the estimate and proceed to the $(l+1)$-th stream data batch. However, this approach results in a loss of information as it discards the information from the previous batch.

Table 4.20: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$

$p = 15, \rho$		0	0.2	0.5
non0 coeff.	$\tilde{\boldsymbol{\beta}}$	0(0.920000)	0(0.903300)	0(0.910000)
	$\tilde{\boldsymbol{\beta}}_{\lambda=0}$	0(0.930000)	0(0.936667)	0(0.923333)
0 coeff.	$\tilde{\boldsymbol{\beta}}$	5.43(0.980800)	5.54(0.983300)	5.58(0.977500)
	$\tilde{\boldsymbol{\beta}}_{\lambda=0}$	12(0.951667)	12(0.954167)	12(0.955000)
$p = 50, \rho$		0	0.2	0.5
non0 coeff.	$\tilde{\boldsymbol{\beta}}$	0(0.883300)	0(0.866667)	0(0.840000)
	$\tilde{\boldsymbol{\beta}}_{\lambda=0}$	0.09(0.583333)	0.09(0.650000)	0(0.780000)
0 coeff.	$\tilde{\boldsymbol{\beta}}$	9.29(0.9936)	7.65(0.9972)	11.02(0.9885)
	$\tilde{\boldsymbol{\beta}}_{\lambda=0}$	45.59(0.871277)	45.59(0.894255)	47(0.947447)

* $\boldsymbol{\beta}_{p\times 1}$ has 3 non-zero coefficients.

* When $p = 50$ and $\rho = 0,\ 0.2$, the non-penalised algorithm fails to converge for studying the last stream batch in some of the 100 repetitions. In such cases, we set the estimated coefficient vector $\tilde{\boldsymbol{\beta}}_{\lambda=0}$ to $\mathbf{0}$.

Table 4.21: Excess risks of classifiers $C_{\tilde{\boldsymbol{\beta}}}$s and $C_{\tilde{\boldsymbol{\beta}}_{\lambda=0}}$s

	ρ	0	0.2	0.5
$p = 15$	$ER(C_{\tilde{\boldsymbol{\beta}}})$	0.996839	0.991110	0.989591
	$ER(C_{\tilde{\boldsymbol{\beta}}_{\lambda=0}})$	0.985663	0.982180	0.986058
$p = 50$	$ER(C_{\tilde{\boldsymbol{\beta}}})$	0.992511	0.988613	0.986645
	$ER(C_{\tilde{\boldsymbol{\beta}}_{\lambda=0}})$	0.828486	0.864634	0.935026

* For both $p = 15$ and $p = 50$, the misclassification rates of the Bayes classifiers are 18.288% for the case of $\rho = 0$, 19.622% for the case of $\rho = 0.2$, and 23.198% for the case of $\rho = 0.5$.

4.3.5 Simulation study: Tuning parameter selection on different sizes of validation data in mLpOCV

This section tests the sensitivity of our incremental algorithm to the size of the data, partitioned from the training data, used for validation during the tuning parameter selection process, and how the size of the validation data affects the estimation results. Through two simulated examples, one with two models of different sparsity and another with two different sizes of training datasets, we examine the performance of our method under varying amounts of validation data.

In this section, all stream data batches have a fixed size of $n_b = 200$. The parameter

settings for the model are

$$\boldsymbol{\beta} = (0, \beta_2, 0, \beta_4, \beta_5, \ 0, \cdots, \ 0)^{\mathsf{T}}_{p\times 1},$$

$$\beta_2 = 0.85, \ \beta_4 = 1, \ \beta_5 = -1,$$

and the correlation between $x_{j_1 i}$ and $x_{j_2 i}$, $i = 1, \cdots, N_b$, is $0.5^{|j_1 - j_2|}$, $j_{1,2} = 2, \ \cdots, \ p$. The validation data, with $n_{cv} = 10, \ 20, \ 30, \ 40, \ 50$ observations, are tested independently in each experiment.

Example 1: In this example, we evaluate the performance of our method on two sparse models with parameter dimensions of $p = 10$ and $p = 100$, where the full-size training datasets have a size of $N_b = 4000$. The numerical results for **Example 1** are presented in Table 4.22-Table 4.24.

Table 4.22 records the AMRSEs and MMRSEs of the renewable estimates. It shows the trend that with more data extracted from the training data for validation, the renewable estimates are more accurate. Specifically, there is a noticeble improvement when the size of validation increases from 10 to 20, while the improvement is less obvious after that as the increase of the validation data. The improvements from increasing the size of the validation dataset are unlikely to be sustainable, and as the validation data is extracted from the training data, attention also needs to be paid to the size of the training dataset, which can affect the accuracy of the estimates.

ESEs and true values, SDs, for estimates of non-zero coefficients are shown in Table C.4. Confidence ellipses for non-zero and zero coefficients are constructed separately from the ESE obtained from (4.14). Table 4.23 shows the CPs for the non-zero and zero coefficients, and the numbers of incorrect non-zero and zero estimates for different sizes of validation data. As for the performance of variable selection, our method of introducing SCAD finds almost all important features, while when the data are relatively sparse ($p = 100$), few features are missed during the 100 iterations of the experiments. As more data is extracted from the training data as validation, the performance of variable selection improves in some cases. In this example, n_{cv} increases from 10 to 40, with CPs for both non-zero and zero coefficients tending to increase for two models with different numbers of covariates. However, it shows that for models with $p = 100$ covariates, the CP of the non-zero coefficient decreases

significantly as n_{cv} increases from 40 to 50.

The computational times of experiments with different sizes of validation data and the excess risks of our classifier are recorded in the Table 4.24. In these cases, for $p = 10$, the classifiers show little difference in efficiency, trained from the same training data while having different sized validation datasets to find the tuning parameter $lambdas$. Although for $p = 100$ there is an improvement it is not great when n_{cv} increases from 10 to 40, though there is a decrease when n_{cv} changes from 40 to 50. As expected, more time is needed as more data is used for validation in mLOPCV.

Table 4.22: AMRSEs and MMRSEs (in brackets) of $\tilde{\boldsymbol{\beta}}$s

n_{cv}	10	20	30	40	50
$p = 10$	1.062(0.340)	0.526(0.344)	0.484(0.340)	0.473(0.346)	0.490(0.333)
$p = 100$	1.305(0.087)	1.147(0.081)	1.001(0.081)	0.293(0.032)	0.918(0.131)

* All values recorded in this table multiplied by 10^{-3} are true values.

Table 4.23: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$

	n_{cv}	10	20	30	40	50
$p = 10$	non0 coeff.	0.01(0.8667)	0(0.8800)	0(0.9067)	0(0.9167)	0(0.9100)
	0 coeff.	1.80(0.9743)	2.00(0.9800)	2.11(0.9671)	2.23(0.9714)	2.28(0.9686)
$p = 100$	non0 coeff.	0.05(0.5033)	0.06(0.5267)	0.04(0.5167)	0(0.7033)	0.04(0.4733)
	0 coeff.	3.86(0.9975)	3.25(0.9988)	2.87(0.9991)	3.15(0.9996)	2.21(0.9996)

* $\boldsymbol{\beta}_{p\times 1}$ has 3 non-zero coefficients.

Table 4.24: Excess risks of $C_{\tilde{\boldsymbol{\beta}}}$s and computing time (in brackets)

n_{cv}	10	20	30	40	50
$p = 10$	0.9937(00:01:32)	0.9948(00:02:51)	0.9952(00:04:02)	0.9962(00:05:25)	0.9962(00:06:09)
$p = 100$	0.9666(00:29:33)	0.9710(00:42:11)	0.9753(01:00:27)	0.9981(01:19:22)	0.9781(01:36:20)

* For both $p = 10$ and $p = 100$, the misclassification rate of the Bayes classifiers is 27.56% across all cases.

* We implemented each experiment with 100 independent trials using C++ and ran them on 2 *Intel Xeon 6138 CPUs*. For each experiment of renewable estimations, we requested 35 cores.

Example 2: In the second example, we compare the performance of our method on training data of different sizes and validation data of different sizes. In detail, $\boldsymbol{\beta}_{p\times 1}$ and $p = 100$, and the performance of renewable estimates from the collected $b = 2$ and $b = 20$ stream data batches are recorded and discussed. Table 4.25-Table 4.27 record the numerical results for **Example 2** in this subsection.

Table 4.25 displays the AMRSEs and MMRSEs (in brackets) of both $\tilde{\boldsymbol{\beta}}_2$s and $\tilde{\boldsymbol{\beta}}_{20}$s. It can be seen that the performance of both the $\tilde{\boldsymbol{\beta}}_2$s and the $\tilde{\boldsymbol{\beta}}_{20}$s improves as more training data is divided up for validation.More specifically, for $b = 20$, there is a consistent improvement in the accuracy of the estimates as n_{cv} increases, while for $b = 2$, AMRSEs and MMRSEs become smaller as n_{cv} increases from 10 to 30 and larger as n_{cv} increases from 30 to 50. We can see that the estimation accuracy of our method is less susceptible to differences in the size of the validation data when there is more observed training data.

ESEs and SDs of the non-zero coefficients of $\tilde{\boldsymbol{\beta}}_2$s and $\tilde{\boldsymbol{\beta}}_{20}$s on $\boldsymbol{\beta}$ are recorded in Table C.5, which is provided in Appendix C.1. The performance of variable selection and CPs for non-zero and zero coefficients at the same preset search interval and step size settings for the search tuning parameter λs is shown in Table 4.26. Table 4.26 reveals that, except for the case where $n_{cv} = 40$, both $\tilde{\boldsymbol{\beta}}_2$s and $\tilde{\boldsymbol{\beta}}_{20}$s exhibit similar numbers of incorrect zero estimates and CPs for non-zero estimates in all cases. It should be noted, $\tilde{\boldsymbol{\beta}}_{20}$s have significantly fewer incorrect zero estimates than $\tilde{\boldsymbol{\beta}}_2$s across different cases with the same size validation data. These findings are consistent with the simulation study presented in Section 4.2.4. In addition it can be observed that the variation in the size of the validation data has a negligible impact on the performance of both $\tilde{\boldsymbol{\beta}}_2$s and $\tilde{\boldsymbol{\beta}}_{20}$s.

Table 4.27 records the excess risk and computation time for the two classifiers $C_{\tilde{\boldsymbol{\beta}}_2}$s and $C_{\tilde{\boldsymbol{\beta}}_{20}}$ for different validation data size cases, as well as the computation time. We can see that, following the same trend as in **Example 1** above, the computation time increases as n_{cv} increases, while the effectiveness of $C_{\tilde{\boldsymbol{\beta}}_2}$s or $C_{\tilde{\boldsymbol{\beta}}_{20}}$s has not changed noticeably.

Table 4.25: AMRSEs and MMRSEs (in brackets) of $\tilde{\boldsymbol{\beta}}_2$s and $\tilde{\boldsymbol{\beta}}_{20}$s

n_{cv}	10	20	30	40	50
$b = 2$	4.335(2.437)	4.114(2.044)	3.938(1.795)	4.450(3.980)	4.318(3.483)
$b = 20$	1.305(0.087)	1.147(0.081)	1.001(0.081)	0.293(0.032)	0.918(0.131)

* All results are the numbers in the table $\times 10^{-3}$.

Table 4.26: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$

	n_{cv}	10	20	30	40	50
$b = 2$	non0 coeff.	0.05(0.5100)	0.06(0.5167)	0.04(0.5200)	0.04(0.4367)	0.04(0.4433)
	0 coeff.	16.08(0.9937)	16.28(0.9974)	16.69(0.9986)	15.34(0.9995)	15.48(0.9991)
$b = 20$	non0 coeff.	0.05(0.5033)	0.06(0.5267)	0.04(0.5167)	0(0.7033)	0.04(0.4733)
	0 coeff.	3.86(0.9975)	3.25(0.9988)	2.87(0.9991)	3.15(0.9996)	2.21(0.9996)

* $\boldsymbol{\beta}_{100\times 1}$ has 3 non-zero coefficients.

Table 4.27: Excess risks of $C_{\tilde{\boldsymbol{\beta}}_b}$s and computing time (in bracket)

n_{cv}	10	20	30	40	50
$ER(C_{\tilde{\boldsymbol{\beta}}_2})$s	0.9125(00:16:12)	0.9071(00:20:55)	0.9114(00:30:09)	0.9002(00:37:53)	0.9075(00:46:17)
$ER(C_{\tilde{\boldsymbol{\beta}}_{20}})$s	0.9666(00:29:33)	0.9710(00:42:11)	0.9753(01:00:27)	0.9981(01:19:22)	0.9781(01:36:20)

* The misclassification rate of the Bayes classifiers is 26.433%.

From the two simulated cases described in this section, it can be concluded that our incremental approach responds less sensitively in terms of to the fair quantities of data that are detached from the training data for validation in the tuning parameter search process. By increasing the size of the validation data, the performance of our approach is improved, but the improvement is not constant as the validation data size increases. An appropriate size of validation dataset needs to be considered, taking into account estimation accuracy, classifier effectiveness, and processing time. Except as otherwise noted, the validation dataset for our study is set to have a size of $n_{cv} = 20$.

4.4 Iteratively updated tuning parameter

In this section we first introduce an incremental algorithm that iteratively updates the penalty term during iterative estimation process. The algorithm in Section 4.2.2

approximates the penalty term using the estimates from previous data batches, so the penalty term remains constant throughout the iterations. The following simulation study is designed to compare the method introduced in Section 4.2.2 with the method described below in Section 4.4.1, and also with the penalised offline estimation method.

4.4.1 Incremental algorithm for penalised maximum likelihood estimation with iteratively updated penalty

In this section, we provide a detailed explanation of the algorithm with an iteratively updated penalty function. We start by investigating the simplest scenario, where two streaming data batches, $\mathcal{B}_1$ and $\mathcal{B}_2$, arrive in order.

As explained in Section 4.2.2, the MLE of $\boldsymbol{\beta}$, obtained from the offline penalised method and denoted as $\hat{\boldsymbol{\beta}}_2^*$, can satisfy Equation (4.6). Referring to (4.7) and removing the error terms, we can rewrite (4.6) as

$$\begin{aligned}&\mathbf{S}_1(\mathcal{B}_1;\hat{\boldsymbol{\beta}}_1)-n_1\mathbf{P}'_{\lambda_1}(\|\hat{\boldsymbol{\beta}}_1\|_1)+\mathbf{H}_1(\mathcal{B}_1;\hat{\boldsymbol{\beta}}_1)(\tilde{\boldsymbol{\beta}}_2-\hat{\boldsymbol{\beta}}_1)\\&+\mathbf{S}_2(\mathcal{B}_2;\tilde{\boldsymbol{\beta}}_2)-N_2\mathbf{P}'_{\lambda_2}(\|\tilde{\boldsymbol{\beta}}_2\|_1)+n_1\mathbf{P}'_{\lambda_1}(\|\hat{\boldsymbol{\beta}}_1\|_1)=\mathbf{0},\end{aligned}\tag{4.21}$$

where $\tilde{\boldsymbol{\beta}}_2$ can be interpreted as a second-order asymptotic approximation for $\hat{\boldsymbol{\beta}}_2^*$, and λ_1 represents the tuning parameter derived from $\mathcal{B}_1$, while λ_2, obtained from $\mathcal{B}_2$.

Since $\hat{\boldsymbol{\beta}}_1$ satisfies Equation (4.5), we can obtain the following equation:

$$\mathbf{H}_1(\mathcal{B}_1;\hat{\boldsymbol{\beta}}_1)(\tilde{\boldsymbol{\beta}}_2-\hat{\boldsymbol{\beta}}_1)+\mathbf{S}_2(\mathcal{B}_2;\tilde{\boldsymbol{\beta}}_2)-N_2\mathbf{P}'_{\lambda_2}(\|\tilde{\boldsymbol{\beta}}_2\|_1)+n_1\mathbf{P}'_{\lambda_1}(\|\hat{\boldsymbol{\beta}}_1\|_1)=\mathbf{0}.\tag{4.22}$$

Utilising the unit score function shown as Equation (4.22), we employ the Newton-Raphson algorithm to find its solution. This method essentially provides a second-order asymptotic approximation for the offline estimate $\hat{\boldsymbol{\beta}}_2^*$. Specifically, during the k-th iteration, where $k=1,2,\cdots$, the estimator takes the form:

$$\begin{aligned}\tilde{\boldsymbol{\beta}}_2^{(k)} &= \tilde{\boldsymbol{\beta}}_2^{(k-1)}-\left[\sum_{l=1}^{2}\mathbf{H}_l(\hat{\boldsymbol{\beta}}_1)-N_2\mathbf{P}''_{\lambda_2}(\|\tilde{\boldsymbol{\beta}}_2^{(k-1)}\|_1)\right]^{-1}\\&\quad\times\left\{\mathbf{H}_1(\hat{\boldsymbol{\beta}}_1)(\tilde{\boldsymbol{\beta}}_2^{(k-1)}-\hat{\boldsymbol{\beta}}_1)+\mathbf{S}_2(\tilde{\boldsymbol{\beta}}_2^{(k-1)})\right.\end{aligned}$$

$$-N_2 P'_{\lambda_2}(\|\tilde{\beta}_2^{(k-1)}\|_1) + n_1 P'_{\lambda_1}(\|\hat{\beta}_1\|_1)\Big\}, \tag{4.23}$$

Upon convergence of the iterative process detailed in Equation (4.23), we obtain the renewable MLE for β, denoted as $\tilde{\beta}_{2,\lambda^*}$.

Compared to the method elucidated in Section 4.2.2, the iterative estimation method introduced in this section requires the retention of additional quantities. Specifically, during the estimation process, we use historical statistical estimates such as $\hat{\beta}_1$ and $H_1(\hat{\beta}_1)$, in conjunction with the size of the streaming data in $\mathcal{B}_1$, denoted as n_1. Additionally, the first derivative of the penalty function with respect to $\hat{\beta}_1$, represented by $P'_{\lambda_1}(\|\hat{\beta}_1\|_1)$, is also used in the iterative estimation process.

The aforementioned historical metrics are then updated using $\tilde{\beta}_{2,\lambda^*}$, $\tilde{H}_2(\tilde{\beta}_{2,\lambda^*})$, and $P'_{\lambda_2}(\|\tilde{\beta}_{2,\lambda^*}\|_1)$. Both the volume of previously accumulated streaming data, denoted as n_1, and the size of the current training data, N_2, are retained. The estimate of dispersion parameter, denoted as $\tilde{\phi}_2$, and the estimated covariance matrix are derived in line with equations (4.13) and (4.14), respectively.

Now we discuss a more general case where b, $b \geqslant 2$, data batches are observed sequentially. At the k-th iteration, when the data in $\mathcal{B}_b$ is studied, we have the renewable estimate of β in the iteration as follows

$$\begin{aligned}\tilde{\beta}_b^{(k)} \;=\; & \tilde{\beta}_b^{(k-1)} - \Big[\tilde{H}_b(\tilde{\beta}_{b-1}) - N_b P''_{\lambda_b}(\|\tilde{\beta}_b^{(k-1)}\|_1)\Big]^{-1} \\ & \times\Big\{\tilde{H}_{b-1}(\tilde{\beta}_{b-1})(\tilde{\beta}_b^{(k-1)} - \tilde{\beta}_{b-1}) + S_b(\tilde{\beta}_b^{(k-1)}) \\ & -N_b P'_{\lambda_b}(\|\tilde{\beta}_b^{(k-1)}\|_1) + N_{b-1} P'_{\lambda_{b-1}}(\|\tilde{\beta}_{b-1}\|_1)\Big\}. \end{aligned} \tag{4.24}$$

When the above iteration converges, the renewable estimate denoted as $\tilde{\beta}_{b,\lambda^*}$ is obtained.

The previous statistics are then updated using $\tilde{\beta}_{b,\lambda^*}$, $\tilde{H}_b(\tilde{\beta}_{b,\lambda^*})$, and $P'_{\lambda_b}(\|\tilde{\beta}_{b,\lambda^*}\|_1)$. Additionally, the estimate for ϕ is refreshed according to equation (4.13). The estimated covariance matrix is revised in line with (4.14). Both the size of the previously collected streaming data, N_{b-1}, and the volume of the current training data, N_b, are preserved.

4.4.2 Simulation study

In this section, we present three simulation examples designed to compare the performance of the penalised offline method in Section 4.2.1, the incremental algorithm described in Section 4.2.2 with a constant penalty term during the estimation iterations, and the algorithm in Section 4.4.1 that has an iteratively updated penalty term. We denote the estimates obtained from the offline method as $\hat{\boldsymbol{\beta}}$s, the estimates from the renewable method in Section 4.2.2 as $\tilde{\boldsymbol{\beta}}_{\lambda}$s, and the estimates from the method introduced in Section 4.4.1 as $\tilde{\boldsymbol{\beta}}_{\lambda^*}$s.

In the first simulated example, different sizes of streaming data are studied in the same experiment; in the second simulation example, non-identical streaming data stored in separate data batches is discussed; and the third simulated example explores non-identical data that is gathered in the same data batch. As explained in Equation (4.16) in Section 4.2.3, we establish a threshold $\sigma_0 = 10^{-5}$ and assign values to the estimator when its absolute value is less than σ_0 during the iterations of the penalised estimation.

We have the parameter set as follows for the three simulated examples in this section:

$$\boldsymbol{\beta} = (0,\ \beta_2,\ \beta_3,\ \cdots,\ \beta_6,\ \beta_7, \cdots,\ \beta_{11},\ 0, \beta_{13}, \cdots, 0)^{\mathsf{T}}_{p\times 1},$$

$$\beta_2 = 0.85,\ \beta_3 = -0.9,\ \beta_6 = 1.15,\ \beta_7 = 0.85,\ \beta_{11} = -0.9,\ \beta_{13} = 1.15,$$

Example 1: In the first example, we study i.i.d. streaming data of different sizes. The size of the full training dataset is $N_b = 2400$ and we examine two models with different dimensions seperately, where $p = 15$ and $p = 100$. The correlation between $x_{j_1 i}$ and $x_{j_2 i}$, $i = 1, \cdots, N_b$, is $0.5^{|j_1 - j_2|}$ where $j_{1,2} = 2,\ \cdots,\ p$. The streaming data sizes are set as follows

$$n_l = \begin{cases} 200 & \text{if} \quad l+2 \pmod 3 \equiv 0, \\ 300 & \text{if} \quad l+2 \pmod 3 \equiv 1, \\ 400 & \text{otherwise}, \end{cases}$$

where $l = 1,\ \cdots, b-1$, and for the last observed data batch $\mathcal{B}_b$, it has a size of

$n_b = N_b - \sum_{l=1}^{b-1} n_l$. The simulation results are displayed in Table 4.28-Table 4.30.

As illustrated in Section 4.3, the settings of intervals and step size for searching the tuning parameter λ of SCAD can affect the estimation results. In this section, we set a consistent step size of 0.15 for all experiments to ensure fair comparisons. With regard to the search intervals, for both renewable estimation methods, when $p = 15$, the search interval is set to $[0.005, 0.15]$, and when $p = 100$, the search interval is set to $[0.015, 0.3]$. We test the offline method on different width intervals. For $p = 15$, we first set the interval for offline method to be the same as for renewable estimation methods and note the results as **C1**, and we also test the offline method on a different interval, $[0.035, 0.35]$ and denote the results as **C2**; when $p = 100$, we first use the same setting as renewable estimation methods, which is $[0.015, 0.3]$ and we also test the offline method on a larger interval, $[0.015, 0.5]$, which has a larger upper bound and is more likely to have larger values of λs. By testing two different widths of intervals, we aim to minimise the influence of the predetermined intervals that may not contain desirable values for the tuning parameter and to avoid coincidental outcomes as far as possible.

The AMRSEs and MMRSEs of the renewable estimates from two online methods and the estimates from the offline method are presented in Table 4.28. We notice that $\tilde{\boldsymbol{\beta}}_\lambda$s and $\tilde{\boldsymbol{\beta}}_{\lambda^*}$s have the similar performance when they are trained by the same dataset which has streaming data coming in different sizes. Moreover, when $p = 15$, the two renewable estimation methods outperform the offline method in different cases. Thus, when the dataset has a larger number of covariates ($p = 100$), making it sparser, the offline method exhibits better performance when the covariates are more correlated ($\rho = 0.5$) while the difference between renewable estimates and the offline estimates get smaller as ρ becomes smaller.

The ESEs and SDs of the estimates are presented in Appendix C.1, specifically in Table C.6. The performance of variable selections and CPs of non-zero and zero coefficients are shown in Table 4.29. Across all cases, it can be observed that the penalised renewable estimation methods and the penalised offline method, which incorporate SCAD into the model, can identify all important features, as evidenced by the 0 incorrect zero estimates in all experiments. It is important to note that $\tilde{\boldsymbol{\beta}}_{\lambda^*}$s consistently have the fewest incorrect non-zero estimates for zero coefficients of $\boldsymbol{\beta}$, which

is substantially lower than both $\tilde{\boldsymbol{\beta}}_\lambda$s and $\hat{\boldsymbol{\beta}}$s. Even with the larger search interval in **C2**, the offline method still generates the estimates with the highest number of incorrect non-zero estimates when $p = 100$. In this table, we can draw the same conclusion as in **Example 2** in Section 4.2.4, which is that under an equivalent computing setting where there is the same threshold for assigning a 0 value to insignificant features, the offline method is less effective than the renewable estimation methods in identifying insignificant features. Between the two renewable estimation methods, the one that uses an iteratively updated penalty term during the estimation iterations holds an advantage over the other method that uses a constant penalty term during iteration, in terms of generating more interpretable models.

Table 4.30 displays the efficiency of the classifiers and the computing time for each 100-replication experiment. As seen from the table, both renewable classifiers have very close efficiency with the Bayes classifier. When $p = 15$ and $\rho = 0.2$, $C(\tilde{\boldsymbol{\beta}}_{\lambda^*})$s perform slightly better than the Bayes classifier with the excess risks over 1. Both renewable classifiers also perform closely with the offline classifier across different cases. It is worth noting that the renewable estimation method with an iteratively updated penalty term in the model requires more computing time compared to the renewable method that does not update the penalty term during the estimation iterations, and in some cases, it requires a far longer time than the offline method.

Table 4.28: AMRSEs and MMRSEs (in brackets) of $\tilde{\boldsymbol{\beta}}_\lambda$s, $\tilde{\boldsymbol{\beta}}_{\lambda^*}$s and $\hat{\boldsymbol{\beta}}^*$s

$p = 15,\ \rho$	0	0.2	0.5
$\tilde{\boldsymbol{\beta}}_\lambda$	0.874(0.679)	0.806(0.666)	1.158(0.891)
$\tilde{\boldsymbol{\beta}}_{\lambda^*}$	0.734(0.648)	0.694(0.552)	0.982(0.756)
C1:$\hat{\boldsymbol{\beta}}^*$	1.338(0.533)	1.342(0.527)	2.723(0.660)
C2:$\hat{\boldsymbol{\beta}}^*$	2.724(0.263)	3.0617(0.257)	2.449(0.284)
$p = 100,\ \rho$	0	0.2	0.5
$\tilde{\boldsymbol{\beta}}_\lambda$	0.307(0.118)	0.423(0.160)	1.044(0.467)
$\tilde{\boldsymbol{\beta}}_{\lambda^*}$	0.361(0.121)	0.461(0.153)	1.303(0.465)
C1:$\hat{\boldsymbol{\beta}}^*$	0.202(0.093)	0.137(0.131)	0.318(0.149)
C2:$\hat{\boldsymbol{\beta}}^*$	0.304(0.102)	0.326(0.132)	0.326(0.149)

* All values recorded in this table multiplied by 10^{-3} are true values.

Table 4.29: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$

$p = 15, \rho$		0	0.2	0.5
non0 coeff.	$\tilde{\boldsymbol{\beta}}_\lambda$	0(0.838333)	0(0.853333)	0(0.830000)
	$\tilde{\boldsymbol{\beta}}_{\lambda^*}$	0(0.886667)	0(0.881667)	0(0.861667)
	C1:$\hat{\boldsymbol{\beta}}^*$	0(0.930000)	0(0.933333)	0(0.906667)
	C2:$\hat{\boldsymbol{\beta}}^*$	0(0.911667)	0(0.900000)	0(0.910000)
0 coeff.	$\tilde{\boldsymbol{\beta}}_\lambda$	6.79(0.947778)	6.57(0.951111)	6.44(0.94111)
	$\tilde{\boldsymbol{\beta}}_{\lambda^*}$	2.96(0.927778)	2.74(0.940000)	2.81(0.927778)
	C1:$\hat{\boldsymbol{\beta}}^*$	6.85(0.955556)	6.74(0.963333)	6.66(0.950000)
	C2:$\hat{\boldsymbol{\beta}}^*$	3.76(0.998889)	3.53(1.000000)	3.23(0.998889)
$p = 100, \rho$		0	0.2	0.5
non0 coeff.	$\tilde{\boldsymbol{\beta}}_\lambda$	0(0.678333)	0(0.581667)	0.02(0.455000)
	$\tilde{\boldsymbol{\beta}}_{\lambda^*}$	0(0.701667)	0(0.631667)	0.04(0.463333)
	C1:$\hat{\boldsymbol{\beta}}^*$	0(0.953333)	0(0.956667)	0(0.943333)
	C2:$\hat{\boldsymbol{\beta}}^*$	0(0.935000)	0(0.936667)	0(0.941667)
0 coeff.	$\tilde{\boldsymbol{\beta}}_\lambda$	40.44(0.997979)	39.84(0.997979)	40.74(0.997766)
	$\tilde{\boldsymbol{\beta}}_{\lambda^*}$	1.94(0.995319)	1.98(0.994787)	1.59(0.996064)
	C1:$\hat{\boldsymbol{\beta}}^*$	68.15(0.984468)	69.07(0.978191)	69.12(0.975745)
	C2:$\hat{\boldsymbol{\beta}}^*$	67.37(0.984362)	68.11(0.978617)	68.81(0.975745)

* $\boldsymbol{\beta}_{p\times 1}$ has 6 non-zero coefficients.

Table 4.30: Excess risks of various classifiers and computing time (in brackets)

$p = 15,\ \rho$	0	0.2	0.5
$\mathtt{ER}(C_{\tilde{\boldsymbol{\beta}}_\lambda})$	0.991703(00:05:15)	0.992933(00:06:24)	0.990700(00:05:47)
$\mathtt{ER}(C_{\tilde{\boldsymbol{\beta}}_{\lambda^*}})$	0.991505(00:08:05)	1.002210(01:29:09)	0.993462(00:54:33)
C1:$\mathtt{ER}(C_{\hat{\boldsymbol{\beta}}^*})$	0.980821(04:36:54)	0.990961(04:27:25)	0.975916(04:24:25)
C2:$\mathtt{ER}(C_{\hat{\boldsymbol{\beta}}^*})$	0.982472(05:03:30)	0.983051(05:03:51)	0.976377(04:55:50)
$p = 100,\ \rho$	0	0.2	0.5
$\mathtt{ER}(C_{\tilde{\boldsymbol{\beta}}_\lambda})$	0.993789(01:23:57)	0.990370(01:25:43)	0.962821(01:11:45)
$\mathtt{ER}(C_{\tilde{\boldsymbol{\beta}}_{\lambda^*}})$	0.980627(02:36:27)	0.987527(37:57:31)	0.953141(26:45:14)
C1:$\mathtt{ER}(C_{\hat{\boldsymbol{\beta}}^*})$	0.986281(17:41:49)	0.993823(20:19:27)	0.984376(20:32:05)
C2:$\mathtt{ER}(C_{\hat{\boldsymbol{\beta}}^*})$	0.983446(21:29:19)	0.989192(21:48:59)	0.982878(21:17:42)

* The misclassification rates of the Bayes classifiers for both $p = 15$ and $p = 100$ models are identical: 19.842%, 19.952%, and 20.666% for $\rho = 0$, $\rho = 0.2$, and $\rho = 0.5$, respectively.

* We implemented each experiment with 100 independent trials using C++ and ran them on 2 *Intel Xeon 6138 CPUs*. For each experiment of renewable estimations, we requested 35 cores. While for the experiments of the offline method, we requested 35 cores when $p = 15$ and 40 cores when $p = 100$.

Example 2: The second example in this section studies the streaming data which

have i.i.d. observations in the same stream data batch while can be not identical distributed among different data batches. The streaming data batches have the same size as $n_b = 300$ and two models which have number of covaraites as $p = 15$ and $p = 150$ are studied separately. The estimates trained by $b = 4$ and $b = 8$ stream data batches are recorded.

Specifically, for the data batch B_l, $l = 1, \cdots, b$, the correlation between $x_{j_1 i}$ and $x_{j_2 i}$, $i = 1, \cdots, N_b$, is $\rho_l^{|j_1 - j_2|}$, $j_{1,2} = 2, \cdots, p$, and we set

$$\rho_l = \begin{cases} 0 & \text{if} \quad l+3 \pmod 3 \equiv 0, \\ 0.2 & \text{if} \quad l+3 \pmod 3 \equiv 1, \\ 0.5 & \text{otherwise.} \end{cases}$$

Table 4.31-Table 4.33 record the numerical results for **Example 2**.

In this example, we set the step size for tuning parameter λ search in all tested methods as 0.15, and we use fair width intervals that can contain desirable values for λ. Specifically, for $p = 15$, we set the search interval as $[0.015, 0.25]$ for all methods, and for $p = 150$, the search interval is $[0.015, 0.35]$ for all methods.

Table 4.31 displays AMRSEs and MMRSEs of the estimates obtained from different methods. It is evident that the overall performance of the offline method is inferior than the renewable estimation method with iteratively updated λs, shown in the table that AMRSEs and MMRSEs of $\hat{\boldsymbol{\beta}}$s are much higher than those of $\tilde{\boldsymbol{\beta}}_{\lambda^*}$s in each case, except only for the case where $p = 150$ and $b = 8$, the AMRSE and MMRSE of $\hat{\boldsymbol{\beta}}$s are slightly lower than those of $\tilde{\boldsymbol{\beta}}_{\lambda}$s and $\tilde{\boldsymbol{\beta}}_{\lambda^*}$s. In this example, the estimation accuracy of $\tilde{\boldsymbol{\beta}}_{\lambda}$s and $\tilde{\boldsymbol{\beta}}_{\lambda^*}$s is closer when the model is sparser (i.e., $p = 150$), while $\tilde{\boldsymbol{\beta}}_{\lambda^*}$s demonstrate better overall performance, exhibiting lower AMRSEs than $\tilde{\boldsymbol{\beta}}_{\lambda}$s when $p = 15$.

Table C.7 records ESEs and SDs of the three estimates for the non-zero coefficients of $\boldsymbol{\beta}$ and can be seen in Appendix C.1. The performance of variable selection and the CPs for the coefficients of $\boldsymbol{\beta}$ are displayed in Table 4.32. Once again, both renewable penalised methods using SCAD perform well in identifying significant features, with only one important feature missed in all cases where 100 repetitions are run for each experiment. The two renewable estimation methods also have close values of CPs

for the non-zero and zero coefficients across different cases. Consistent with the findings in **Example 1** in this section, $\tilde{\boldsymbol{\beta}}_{\lambda^*}$s have the least incorrect non-zero estimates than both $\tilde{\boldsymbol{\beta}}_{\lambda}$s and $\hat{\boldsymbol{\beta}}^*$s. Furthermore, in all experiments, $\hat{\boldsymbol{\beta}}^*$s have more than half of their non-zero estimates incorrect. The comparison once again demonstrates that the incremental algorithms, which perform more variable selections during the estimation process, have an advantage over the offline method in generating a more interpretable model.

The Excess risks and computing time are recorded in Table 4.33. Both renewable classifiers have very close performance to the Bayes classifier and with more stream data batches are observed, the efficiency of both renewable classifiers improves. In this example, the two renewable classifiers are more efficient than the offline classifier while require less computing time when studying the same dataset. The method in 4.4.1 which updates the penalty term during the iterations needs much more time than the method in Section 4.2.2 that has a consistent penalty term during the iterative estimations. In addition, when the model has more covariates or more streaming data is collected, the gaps between the computing time of the two renewable methods is larger as expected. However, in this example, where non-identical stream data batches are studied, both incremental algorithms take much less time than the offline method.

Table 4.31: AMRSEs and MMRSEs (in brackets) of $\tilde{\boldsymbol{\beta}}_{\lambda}$s, $\tilde{\boldsymbol{\beta}}_{\lambda^*}$s and $\hat{\boldsymbol{\beta}}^*$s

		$\tilde{\boldsymbol{\beta}}_{\lambda}$	$\tilde{\boldsymbol{\beta}}_{\lambda^*}$	$\hat{\boldsymbol{\beta}}^*$
$p = 15$	$b = 4$	2.065(0.806)	1.067(0.809)	1.743(0.953)
	$b = 8$	1.069(0.336)	0.527(0.328)	2.841(0.370)
$p = 150$	$b = 4$	0.330(0.093)	0.459(0.110)	0.523(0.216)
	$b = 8$	0.175(0.042)	0.155(0.048)	0.142(0.026)

* All values recorded in this table multiplied by 10^{-3} are true values.

Table 4.32: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$

$p = 15$		$\tilde{\boldsymbol{\beta}}_{\lambda}$	$\tilde{\boldsymbol{\beta}}_{\lambda^*}$	$\hat{\boldsymbol{\beta}}^*$
$b = 4$	non0 coeff.	0.01(0.905000)	0.01(0.923333)	0(0.930000)
	0 coeff.	3.62(0.970000)	1.86(0.966667)	6.32(0.955556)
$b = 8$	non0 coeff.	0.01(0.895000)	0.01(0.920000)	0(0.913333)
	0 coeff.	3.16(0.981111)	0.58(0.983333)	5.69(0.956667)
$p = 150$		$\tilde{\boldsymbol{\beta}}_{\lambda}$	$\tilde{\boldsymbol{\beta}}_{\lambda^*}$	$\hat{\boldsymbol{\beta}}^*$
$b = 4$	non0 coeff.	0.01(0.800000)	0.01(0.803333)	0(0.885000)
	0 coeff.	16.51(0.999444)	14.20(0.991319)	104.48(0.969167)
$b = 8$	non0 coeff.	0.01(0.788333)	0.01(0.798333)	0(0.908333)
	0 coeff.	10.2(0.999444)	1.71(0.995347)	74.27(1.00000)

* $\boldsymbol{\beta}_{p\times 1}$ has 6 non-zero coefficients.

Table 4.33: Excess risks of $C_{\tilde{\boldsymbol{\beta}}_{\lambda}}$s, $C_{\tilde{\boldsymbol{\beta}}_{\lambda^*}}$s and $C_{\hat{\boldsymbol{\beta}}^*}$s, and computing time (in brackets)

		$ER(C_{\tilde{\boldsymbol{\beta}}_{\lambda}})$	$ER(C_{\tilde{\boldsymbol{\beta}}_{\lambda^*}})$	$ER(C_{\hat{\boldsymbol{\beta}}^*})$
$p = 15$	$b = 4$	0.983202(00:06:15)	0.984636(00:07:50)	0.982058(01:10:01)
	$b = 8$	0.992718(00:06:57)	0.995740(00:09:28)	0.974777(04:40:52)
$p = 150$	$b = 4$	0.977130(01:08:55)	0.960539(01:37:03)	0.952497(08:49:48)
	$b = 8$	0.996346(01:22:08)	0.993522(02:09:15)	0.996144(14:57:53)

* The misclassification rates of the Bayes classifiers are obtained by testing on different datasets separately for the cases where $b = 4$ and $b = 8$. Specifically, for both models with dimensions $p = 15$ and $p = 150$, the misclassification rate is 20.252% when $b = 4$, and 19.632% when $b = 8$.

* We implemented each experiment with 100 independent trials using C++ and ran them on 2 *Intel Xeon 6138 CPUs*. For each experiment of renewable estimations, we requested 35 cores. While for the experiments of the offline method, we requested 35 cores when $p = 15$ and 40 cores when $p = 150$.

Example 3: The last example in this section discusses streaming data that consists of independent but not identically distributed observations within the same data batch.

We explain how to generate the data for **Example 3**. For generating the covariates, we first generate the data in the set denoted as $\mathcal{B}^0_{l'}$, $l' = 1, \cdots, b'$. The data in the same set $\mathcal{B}^0_{l'}$ is i.i.d. and the covariates consisting of $x_{i1} = 1$ and the other $p - 1$ covariates have the correlation as $\rho_l^{|j_1 - j_2|}$, where $i \in \mathcal{B}^0_{l'}$ and $j_{1,2} = 2, \cdots, p$. The first $b' - 1$ data batches have the size of 73 data respectively, and the last data batch $\mathcal{B}^0_{b'}$ has a size of $n_{b'} = N_{b'} - 73 \times (b' - 1)$ data. The correlation parameter ρ_l can be different among different data batches $\mathcal{B}^0_{l'}$. In detail, the correlation between the covariates in the same set $\mathcal{B}^0_{l'}$ is set as

$$\rho_{l'} = \begin{cases} 0 & \text{if} \quad l' + 3 \pmod 3 \equiv 0, \\ 0.2 & \text{if} \quad l' + 3 \pmod 3 \equiv 1, \\ 0.5 & \text{otherwise.} \end{cases}$$

Subsequently, we take N_b, where $N_b < N_{b'}$, data from the generated $N_{b'}$ data and study the streaming data coming with a fixed size of $n_b = 300$, which is packed in stream data batch denoted as B_l, $l = 1, \cdots, b$. In this way, the simulation ensures that the observed data in the same data batch B_l are not identically distributed, with different correlated covariates. We take 500 independent data from the generated dataset for testing.

In this example, We set the dimension of $\boldsymbol{\beta}_{p \times 1}$ as $p = 150$ and record the performance of estimates trained by $b = 8$ and $b = 20$ stream data batches respectively. For the selection of tuning parameter λs in mLOPCV, we set the step size for the renewable estimation process as 0.15 and the search interval as $[0.015, 0.35]$; for the offline estimation process, we set the search step size as 0.25 and the search interval is $[0.015, 0.065]$. Table 4.34-Table 4.36 records the numerical results.

The AMRSEs and MMRSEs of the estimates are recorded in Table 4.34. In this example, the three estimation methods have similar performance. Specifically, when trained by the same training dataset, $\hat{\boldsymbol{\beta}}$s have the best overall performance compared with $\tilde{\boldsymbol{\beta}}_\lambda$s and $\tilde{\boldsymbol{\beta}}_{\lambda^*}$s, while the renewable estimates have very close or even smaller

MMESEs compared to the offline estimates for both cases. $\tilde{\boldsymbol{\beta}}_{\lambda^*}$s have slightly better performance than $\tilde{\boldsymbol{\beta}}_{\lambda}$s in each case.

ESEs and SDs of the three estimates for the non-zero coefficients of $\boldsymbol{\beta}$ can be seen in Appendix C.1, Table C.8. Table 4.35 displays the performance of variable selection of the three methods and the CPs for the coefficients of $\boldsymbol{\beta}$. The results show the same conclusions as the previous two examples. These are: Firstly that both two renewable penalised methods with SCAD have good performance in finding the significant features, with only 1 important feature missed in the 100 repetitions in all cases. Secondly that hugely outperform the offline method in diagnosing the insignificant features. It is also noted that the two renewable estimation methods have close values of CPs for the non-zero and zero coefficients across different cases. It should be taken into account that $\tilde{\boldsymbol{\beta}}_{\lambda^*}$s have far fewer incorrect non-zero estimates than $\tilde{\boldsymbol{\beta}}_{\lambda}$s, and it is noticeable when 20 stream data batches are collected, as there is only 1 incorrect non-zero estimates of $\tilde{\boldsymbol{\beta}}_{\lambda^*}$ in the 100 replication experiment.

The excess risks of the classifiers in Table 4.36 shows again that the renewable classifiers have almost the same efficiency as the Naive Bayes classifier. In this example, $C_{\tilde{\boldsymbol{\beta}}_{\lambda}}$s and $C_{\tilde{\boldsymbol{\beta}}_{\lambda^*}}$s have very close misclassification rates while the computing time of the incremental algorithm with the updated tuning parameter during the iteration is longer. Moreover, the two renewable classifiers have similar performance and are even slightly better than the classifiers of the offline method. Besides, the two incremental methods have obvious advantages over the offline method in requiring less computing time and storage space.

Table 4.34: AMRSEs and MMRSEs (in brackets) of $\tilde{\boldsymbol{\beta}}_{\lambda}$s, $\tilde{\boldsymbol{\beta}}_{\lambda^*}$s and $\hat{\boldsymbol{\beta}}^*$s

	$\tilde{\boldsymbol{\beta}}_{\lambda}$	$\tilde{\boldsymbol{\beta}}_{\lambda^*}$	$\hat{\boldsymbol{\beta}}^*$
$b = 8$	0.170(0.042)	0.167(0.046)	0.114(0.082)
$b = 20$	0.063(0.015)	0.052(0.013)	0.017(0.013)

* All values recorded in this table multiplied by 10^{-3} are true values.

Table 4.35: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for β

		$\tilde{\beta}_{\lambda}$	$\tilde{\beta}_{\lambda^*}$	$\hat{\beta}^*$
$b = 8$	non0 coeff.	0(0.773333)	0(0.776667)	0(0.940000)
	0 coeff.	10.73(0.998889)	1.27(0.997292)	104.34(0.981389)
$b = 20$	non0 coeff.	0(0.795000)	0(0.821667)	0(0.943333)
	0 coeff.	5.44(0.999444)	0.01(0.999583)	95(0.998472)

* $\beta_{150\times1}$ has 6 non-zero coefficients.

Table 4.36: Excess risks of classifiers $C_{\tilde{\beta}_{\lambda}}$s, $C_{\tilde{\beta}_{\lambda^*}}$s and $C_{\hat{\beta}^*}$s, and computing time (in brackets)

	$ER(C_{\tilde{\beta}_{\lambda}})$	$ER(C_{\tilde{\beta}_{\lambda^*}})$	$ER(C_{\hat{\beta}^*})$
$b = 8$	0.994320(01:21:40)	0.988705(02:04:39)	0.981822(12:32:46)
$b = 20$	0.995778(01:59:19)	0.995678(03:12:39)	0.990104(111:44:50)

* The misclassification rates of the Bayes classifiers are obtained by testing on different datasets separately for the cases where $b = 8$ and $b = 20$. Specifically, when $b = 8$ the misclassification rate is 20.308% when $b = 20$, it is 19.81%.

* We implemented each experiment with 100 independent trials using C++ and ran them on 2 *Intel Xeon 6138 CPUs*. For each experiment of renewable estimations, we requested 35 cores and for the experiments of the offline method, we requested 35 cores too.

To conclude, from the three simulated examples with different types of streaming data, we see the stability of our introduced incremental algorithms. The two renewable methods have stable performance with varying sizes of streaming data, and different correlated streaming data. Moreover, both of them outperform the offline method in estimation accuracy and generate more interpretable models, which can be explained by the algorithms with more rounds to select variables with the streaming data coming in different data batches. The renewable method with an iteratively updated penalty term is more capable of detecting insignificant features than the renewable method with a constant penalty term during the iterative estimation. That said, it requires much more time to update the penalty term in each iteration, especially when the sizes of data batches are large or the number of covariates is large. The choice of methods depends therefore on the specific case and the need to consider the computing efficiency or the interpretation of the generated models.

4.5 The incremental algorithm with Independence Screening (IS) and its variant approaches

In this section, we examine five penalised incremental algorithms designed for sparse streaming data, with a focus on high-dimensional streaming data. When observations originate from sparse models, particularly when covariate correlations are strong, obtaining precise estimation results becomes more challenging. Based on the simulation study outlined in earlier sections, the numerical findings indicate that SCAD is efficient in analysing low-dimensional sparse streaming data. However, our subsequent simulation study reveals that the incremental algorithm with SCAD does not converge during the estimation iterations for high-dimensional streaming data, where the number of covariates exceeds the size of the streaming data.

We initially introduce the IS algorithm for studying streaming data from the GLM and then present two algorithms implementing variants of IS. The following simulation study compares the performance of the renewable method using SCAD, IS-SCAD, and two variants of IS-SCAD in studying sparse streaming data. Subsequently, we present the IIS algorithm for analysing streaming data, along with an adaptation of IIS for online algorithms. One simulation study is designed to compare the performance of IS-SCAD with IIS-SCAD and their respective variants for high-dimensional streaming data. Another simulation study aims to compare the performance of online algorithms with IIS-SCAD, the variant of IIS-SCAD with the offline algorithm employing IIS-SCAD.

In this section, we represent the index set of non-zero coefficients of $\boldsymbol{\beta}$ as $\mathcal{M}_* = \{1 \leqslant j \leqslant p : \beta_j \neq 0\}$, which has a size of $s = |\mathcal{M}_*|$. We maintain the assumption that the intercept term of the model (4.1) is present.

4.5.1 The incremental algorithm with IS

Upon observing the data in the first batch $\mathcal{B}_1$, the estimation procedure is identical to the offline method with IS (Fan et al., 2009). When the stream data batch $\mathcal{B}_b$ with the size of n_b, $b = 2, \cdots$, comes, we first calculate MMLEs of $\boldsymbol{\beta}$ denoted as $\hat{\boldsymbol{\beta}}_j^{\mathcal{M}}$,

$j = 2, \cdots, p$, using the observations in $\mathcal{B}_b$,

$$\hat{\boldsymbol{\beta}}_j^M = (\hat{\beta}_{j,1}^M, \hat{\beta}_j^M) = \arg\max_{\beta_1, \beta_j} \sum_{i=1}^{n_b} \ell(\beta_1 + \beta_j X_{ji}, \ y_i). \tag{4.25}$$

We rank the absolute values of the MMLEs, excluding the intercept, based on their magnitudes. The indices of the selected estimates are kept in the index set denoted as

$$\hat{\mathcal{M}} = \{2 \leqslant j \leqslant p : |\hat{\beta}_j^M| \geqslant \sigma\}, \tag{4.26}$$

where $\sigma > 0$ is a given constant as a threshold and d, $1 \leqslant d < p$, and estimates that have the highest ranked absolute values are achieved. The variables not chosen, which have the indices in $\hat{\mathcal{M}}^c$, are seen as insignificant variables.

Referring to the algorithm introduced in Section 4.2.2, at the k-th, $k = 1, \cdots,$ iteration, we have the estimator of $\boldsymbol{\beta}_{\hat{\mathcal{M}}}$ as

$$\begin{aligned}(\tilde{\boldsymbol{\beta}}_{\hat{\mathcal{M}}})_b^{(k)} \ = \ & (\tilde{\boldsymbol{\beta}}_{\hat{\mathcal{M}}})_b^{(k-1)} - \left[\tilde{\mathbf{H}}_b((\tilde{\boldsymbol{\beta}}_{\hat{\mathcal{M}}})_{b-1}) - N_b \mathbf{P}''_{\lambda_b}(\|(\tilde{\boldsymbol{\beta}}_{\hat{\mathcal{M}}})_{b-1}\|_1)\right]^{-1} \\ & \times \Bigg\{ \left[\tilde{\mathbf{H}}_{b-1}((\tilde{\boldsymbol{\beta}}_{\hat{\mathcal{M}}})_{b-1}) - N_b \mathbf{P}''_{\lambda_b}(\|(\tilde{\boldsymbol{\beta}}_{\hat{\mathcal{M}}})_{b-1}\|_1)\right] \\ & \times ((\tilde{\boldsymbol{\beta}}_{\hat{\mathcal{M}}})_b^{(k-1)} - (\tilde{\boldsymbol{\beta}}_{\hat{\mathcal{M}}})_{b-1}) + \mathbf{S}_b((\tilde{\boldsymbol{\beta}}_{\hat{\mathcal{M}}})_b^{(k-1)}) - n_b \mathbf{P}'_{\lambda_b}(\|(\tilde{\boldsymbol{\beta}}_{\hat{\mathcal{M}}})_{b-1}\|_1) \Bigg\},\end{aligned}$$

where $(\tilde{\boldsymbol{\beta}}_{\hat{\mathcal{M}}})_{b-1} = \{(\tilde{\beta}_j)_{b-1}\}_{j \in \hat{\mathcal{M}}}$. Once the aforementioned iteration converges, the renewable MLE, denoted as $\tilde{\boldsymbol{\beta}}_b$, is obtained. Similarly to the previous renewable penalised estimation, we update the corresponding historical statistics from the previous $b-1$ data batches, and free up storage space for the data details.

In this study, we do not explore the threshold σ used for variable selection during the IS process. Rather, we assign threshold values manually in the ensuing simulation studies. This area presents an avenue for further exploration in future research. We posit that our renewable estimation method exhibits the sure screening property as delineated in (2.5) when applied to streaming data. Empirical evidence from subsequent simulation studies supports this conjecture, though a formal proof remains a topic for future exploration.

4.5.2 The incremental algorithm with IS_{V1}: Variant 1 of IS

In this section, we explain one variant of IS used for the incremental algorithm, which considers the historical information for the calculation of MMLEs when b, $b \geqslant 2$, data batches are collected. We denote the extension of IS in this section as IS_{V1}.

In detail, for calculating renewable MMLEs, we introduce the historical statistics from the previous data batches. At the k-th, $k = 1, \cdots$, iteration, for $(\boldsymbol{\beta}_b)_j^M = ((\beta_b)_1, (\beta_b)_j)^T$, the renewable estimator has the form of

$$\begin{aligned}(\tilde{\boldsymbol{\beta}}_j)_b^{M(k)} = (\tilde{\boldsymbol{\beta}}_j)_b^{M(k-1)} \quad &- \quad \left[\tilde{\mathbf{H}}_{b-1}((\tilde{\boldsymbol{\beta}}_j)_{b-1}) + \mathbf{H}_b((\tilde{\boldsymbol{\beta}}_j)_b^{M(k-1)})\right]^{-1} \\ &\times \quad \left\{\tilde{\mathbf{H}}_{b-1}((\tilde{\boldsymbol{\beta}}_j)_{b-1}) \times \left[(\tilde{\boldsymbol{\beta}}_j)_b^{M(k)} - (\tilde{\boldsymbol{\beta}}_j)_{b-1}\right]\right. \\ &\qquad \left. + \mathbf{S}_b((\tilde{\boldsymbol{\beta}}_j)_b^{M(k-1)})\right\}, \end{aligned} \tag{4.27}$$

where $(\tilde{\boldsymbol{\beta}}_j)_{b-1} = ((\tilde{\beta}_1)_{b-1}, (\tilde{\beta}_j)_{b-1})^T$. When the iteration converges, we get the estimates as $(\tilde{\boldsymbol{\beta}}_j)_b^M = ((\tilde{\beta}_{j,1})_b^M, (\tilde{\beta}_j)_b^M)^T$, where $j = 2, \cdots, p$. We name it renewable MMLEs. After ranking the absolute values of the $p-1$ renewable MMLEs, the index set containing the d, $1 \leqslant d < p$ largest absolute values estimates is got which is

$$\hat{\mathcal{M}}_b^{V1} = \{2 \leqslant j \leqslant p : |(\tilde{\beta}_j)_b^M| \geqslant \sigma\},$$

where $\sigma > 0$ is a given constant. Then, by following the incremental algorithm introduced in the previous section, the renewable estimate of $\boldsymbol{\beta}$ can be obtained, and the historical statistics are updated accordingly.

4.5.3 The incremental algorithm with IS_{V2}: Variant 2 of IS

In this section, we propose an additional extension of IS for the incremental algorithm and denote it as IS_{V2}. When the data in $\mathcal{B}_b$, $b \geqslant 2$ is observed, we follow the same procedure outlined in Section 4.5. First, through the screening process, we identify d predictors, where $1 \leqslant d < p$, with the highest ranked absolute values of the MMLEs. We obtain the index set denoted as $\hat{\mathcal{M}}_b$. Next, we verify whether the non-zero indices of $\tilde{\boldsymbol{\beta}}_{b-1}$ are contained in $\hat{\mathcal{M}}_b$, and a new index set is acquired with the following form:

$$\hat{\mathcal{M}}_b^{V2} = \hat{\mathcal{M}}_b \cup \{1 \leqslant j \leqslant p : (\tilde{\beta}_j)_{b-1} \neq 0\}.$$

Analysing the selected variables with indices in $\hat{\mathcal{M}}_b^{V2}$, the incremental algorithm with a penalty function introduced in the previous section is applied. Consequently, the renewable estimate of $\boldsymbol{\beta}$ can be obtained, and the historical statistics are updated accordingly.

4.5.4 Simulation study: Comparative analysis of online algorithms with SCAD, IS-SCAD, IS$_{V1}$-SCAD and IS$_{V2}$-SCAD for the case of $p < n_b$

In this subsection, we compare the performance of online algorithms with SCAD, IS-SCAD, IS$_{V1}$-SCAD, and IS$_{V2}$-SCAD using a simulated example. In this section, when we mention the names of the renewable estimation methods, we simplify them and use the names of variable selection methods such as SCAD, IS-SCAD, IS$_{V1}$-SCAD, and IS$_{V2}$-SCAD. For the screening processes of estimators using IS-SCAD, IS$_{V1}$-SCAD, and IS$_{V2}$-SCAD, we select $d = \lfloor n_b/(2 \times \log(n_b)) \rfloor$ variables in each screening process while analysing the data in $\mathcal{B}_b$, where $\mathcal{B}_b$ represents a batch of stream data with a size of n_b. Here, b, $b = 1, \cdots$, denotes the observed batch number. In all tuning parameter selection processes to find λs, the search interval is defined as $[0.025, 0.35]$ with a step size of 0.15.

The total size of the training dataset is $N_b = 1200$ and there are $n_b = 300$ observations in each stream data batch. The correlation between $x_{j_1 i}$ and $x_{j_2 i}$, where $i = 1, \cdots, N_b$, is given by $\rho^{|j_1 - j_2|}$, with $j_{1,2} = 2, \cdots, p$. We test $\rho = 0,\ 0.2,\ 0.5$, respectively. The parameter $\boldsymbol{\beta}$ is set as

$$\boldsymbol{\beta} = (0,\ \beta_2,\ \beta_3,\ \cdots,\ \beta_6,\ \beta_7,\ 0 \cdots,\ 0)^T_{p \times 1},$$

$$\beta_2 = -0.8,\ \beta_3 = 1,\ \beta_6 = -0.8,\ \beta_7 = 1,$$

where $p = 150$. Table 4.37-Table 4.40 record the numerical results of different variable selection methods.

The AMRSEs and MMRSEs of various renewable estimates are documented in Table 4.37. When the covariates exhibit lower correlation ($\rho = 0$ or 0.2), all tested methods produce renewable estimates with similar performance. As the correlation parameter increases (when $\rho = 0.5$), a consistent trend emerges where all estimates exhibit reduced accuracy, as indicated by the increasing values of both AMRSEs

and MMRSEs across all estimates. Given this observation, it is notable that $\tilde{\boldsymbol{\beta}}_{\mathrm{SCAD}}$s outperform other estimates obtained through IS and its variants in terms of estimation accuracy. The three methods utilising IS exhibit similar performance overall. For the case of $\rho = 0.5$, however, IS-SCAD and $\mathrm{IS}_{\mathrm{V2}}$-SCAD show similarity in their performance, while $\mathrm{IS}_{\mathrm{V1}}$-SCAD, which incorporates historical statistics during the screening process, demonstrates slightly superior performance compared to the other two methods.

Table 4.38 presents the ESEs and SDs of the four estimates across different cases. For all estimation methods, ESEs are close to SDs when $\rho = 0$ or $\rho = 0.2$. However, it is noteworthy that in this example, when $\rho = 0.5$, the estimation methods employing IS and its extensions fail to capture the non-zero coefficients in some of the 100-replication experiments. This leads to the ESE being denoted as 0, signifying that the non-zero coefficient is missed. The missed significant features are also evident in Table 4.39, which records the performance of variable selection for the methods.

Table 4.39 showcases the performance of variable selection, including the numbers of incorrect non-zero and zero estimates and CPs for non-zero and zero coefficients. As the correlation parameter ρ increases, CPs for non-zero coefficients decrease across all methods. However, the methods employing IS-SCAD and its extensions show a more pronounced decrease compared to the method using SCAD, particularly for IS-SCAD and $\mathrm{IS}_{\mathrm{V1}}$-SCAD. When analysing the same datasets, the method aligned with $\mathrm{IS}_{\mathrm{V1}}$-SCAD demonstrates higher CPs for non-zero coefficients compared to the other two methods that incorporate IS into their algorithms. The results demonstrate that the renewable estimation method using SCAD alone can identify all important features in all cases. When the covariates are uncorrelated ($\rho = 0$), the renewable methods utilising IS-SCAD and its two variants can identify all important features, with only a few being missed when the covariates are moderately correlated ($\rho = 0.2$). However, in the case of higher correlation ($\rho = 0.5$), two important features are nearly missed in the 100 replication simulation. Among the three IS methods, $\mathrm{IS}_{\mathrm{V1}}$-SCAD exhibits a lower number of missed non-zero coefficients compared to the other two methods in the same scenario across the 100 trials. Furthermore, it is noteworthy that, when using the same computing setting of σ_0 to screen out insignificant features in Equation (4.16), the renewable estimation method solely employing SCAD demonstrates a lower capability to detect insignificant features

compared to other renewable methods that combine SCAD with IS or its extensions.

Table 4.40 presents the excess risks of the classifiers and the computation time for different methods. When studying less correlated data, the four renewable classifiers have similar efficiency and are all close to the Bayes classifiers. When the correlation parameter is larger ($\rho = 0.5$), the incremental method employing IS-SCAD and its extensions is inferior to the classifiers trained by the method using SCAD alone. However, the classifiers of the method using IS_{V1}-SCAD perform slightly better than those of IS-SCAD and IS_{V2}-SCAD, displaying higher numerical values of excess risks across different cases. It should be noted that, under the same settings for choosing a tuning parameter, it is evident that the method using SCAD alone requires significantly more computation time than the methods adopting IS-SCAD and its extensions for variable selection. This highlights the advantage of IS in reducing the dimensions of a model and accelerating the computation efficiently.

Table 4.40 provides the excess risks of the classifiers and the computation time for different methods. When analysing less correlated data, the four renewable classifiers demonstrate similar efficiency, closely approaching the performance of the Bayes classifier. However, as the correlation parameter increases ($\rho = 0.5$), the incremental method utilising IS-SCAD and its extensions shows inferior performance compared to the classifiers trained by the method using SCAD alone. Nevertheless, the classifiers obtained through the method using IS_{V1}-SCAD perform slightly better than those obtained through IS-SCAD and IS_{V2}-SCAD, exhibiting higher numerical values of excess risks across different cases. It is worth noting that, when the same tuning parameter selection settings are applied, the method using SCAD alone requires significantly more computation time than the methods incorporating IS-SCAD and its extensions. This highlights the advantage of IS in reducing the dimensionality of a model and efficiently accelerating computation.

In conclusion, this example highlights the limitations of IS when dealing with highly correlated data. Specifically, when analysing low-dimensional sparse streaming data, the renewable estimation method incorporating SCAD demonstrates strong performance in terms of estimation accuracy and classifier efficiency across different scenarios with varying correlated covariates. However, this method requires significantly more computation time compared to the renewable estimation methods utilising IS-SCAD and its variants in this example. Nonetheless, the example also

showcases the inherent shortcomings of IS in effectively handling highly correlated covariates. While variants of IS-SCAD, such as IS_{V1}-SCAD, which considers historical statistics from previous stream data, offer some improvement to the performance of IS-SCAD, the enhancement remains limited. During the 100 repetition experiments, a notable number of important features are still missed, underscoring the significance of this limitation.

Table 4.37: AMRSEs and MMRSEs (in brackets) of $\tilde{\boldsymbol{\beta}}_{\mathrm{SCAD}}$s, $\tilde{\boldsymbol{\beta}}_{\mathrm{IS-SCAD}}$s, $\tilde{\boldsymbol{\beta}}_{\mathrm{IS_{V1}-SCAD}}$s and $\tilde{\boldsymbol{\beta}}_{\mathrm{IS_{V2}-SCAD}}$s

ρ	$\tilde{\boldsymbol{\beta}}_{\mathrm{SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\mathrm{IS-SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\mathrm{IS_{V1}-SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\mathrm{IS_{V2}-SCAD}}$
0.5	1.618(0.2359)	3.434(3.310)	2.327(1.831)	3.361(3.232)
0.2	0.569(0.087)	0.695(0.070)	0.708(0.081)	0.700(0.079)
0	0.383(0.084)	0.484(0.063)	0.530(0.073)	0.490(0.066)

* All values recorded in this table multiplied by 10^{-3} are true values.

Table 4.38: SDs and ESEs for non-zero $\boldsymbol{\beta}_{150\times 1}$ coefficient estimates

ρ		$\tilde{\boldsymbol{\beta}}_{\mathrm{SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\mathrm{IS-SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\mathrm{IS_{V1}-SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\mathrm{IS_{V2}-SCAD}}$
0.5	β_2	0.2161(0.0792,0.1175)	1.1861(0,0)	0.1286(0.0796,0.1180)	1.1861(0,0)
	β_3	0.2250(0.0846,0.1254)	0.6223(0.0690,0.1023)	0.2916(0.0817,0.1211)	0.5568(0.0697,0.1033)
	β_6	0.1600(0.0814,0.1207)	1.1861(0,0)	1.1861(0,0)	1.1861(0,0)
	β_7	0.1704(0.0844,0.1251)	0.6079(0.0693,0.1027)	0.5550(0.0704,0.1044)	0.5998(0.0693,0.1028)
0.2	β_2	0.0982(0.0780,0.1156)	0.1055(0.0802,0.1189)	0.0944(0.0806,0.1195)	0.1058(0.0802,0.1189)
	β_3	0.1178(0.0826,0.1225)	0.0981(0.0844,0.1252)	0.1050(0.0846,0.1255)	0.1005(0.0845,0.1252)
	β_6	0.1093(0.0783,0.1161)	0.1096(0.0806,0.1196)	0.1061(0.0810,0.1202)	0.1076(0.0807,0.1197)
	β_7	0.1186(0.0826,0.1225)	0.1056(0.0847,0.1256)	0.1080(0.0850,0.1259)	0.1053(0.0848,0.1257)
0	β_2	0.1065(0.0799,0.1185)	0.0872(0.0831,0.1232)	0.0922(0.0832,0.1233)	0.0841(0.0831,0.1232)
	β_3	0.1072(0.0838,0.1242)	0.0973(0.0862,0.1278)	0.1033(0.0863,0.1279)	0.0986(0.0859,0.1274)
	β_6	0.0868(0.0795,0.1179)	0.0926(0.0823,0.1221)	0.0892(0.0824,0.1221)	0.0960(0.0822,0.1219)
	β_7	0.1044(0.0834,0.1237)	0.1007(0.0865,0.1282)	0.1029(0.0866,0.1284)	0.1011(0.0864,0.1281)

* For each cell, the value outside the brackets represents SD, while the first and second numbers inside the brackets represent $\mathrm{ESE_m}$ and $\mathrm{ESE_o}$, respectively;

Table 4.39: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$

ρ		$\tilde{\boldsymbol{\beta}}_{\text{SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\text{IS}-\text{SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\text{IS}_{\text{V1}}-\text{SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\text{IS}_{\text{V2}}-\text{SCAD}}$
0.5	non0 coeff.	0(0.527500)	1.75(0.145000)	1.19(0.440000)	1.70(0.157500)
	0 coeff.	87.32(0.999247)	0.39(0.999795)	0.51(0.998630)	0.76(0.999452)
0.2	non0 coeff.	0(0.762500)	0.09(0.815000)	0.28(0.832500)	0.09(0.812500)
	0 coeff.	87.15(0.999247)	0.40(0.999795)	0.53(0.998767)	0.66(0.999589)
0	non0 coeff.	0(0.795000)	0(0.840000)	0.12(0.840000)	0(0.840000)
	0 coeff.	86.78(0.998904)	0.45(0.999795)	0.58(0.998630)	0.85(0.999315)

* $\boldsymbol{\beta}_{150\times 1}$ has 4 non-zero coefficients.

Table 4.40: Excess risks of various classifiers and computing time (in brackets)

ρ	$\text{ER}(C_{\tilde{\boldsymbol{\beta}}_{\text{SCAD}}})$	$\text{ER}(C_{\tilde{\boldsymbol{\beta}}_{\text{IS}-\text{SCAD}}})$	$\text{ER}(C_{\tilde{\boldsymbol{\beta}}_{\text{IS}_{\text{V1}}-\text{SCAD}}})$	$\text{ER}(C_{\tilde{\boldsymbol{\beta}}_{\text{IS}_{\text{V2}}-\text{SCAD}}})$
0.5	0.941705(02:03:38)	0.807212(00:10:52)	0.853721(00:11:32)	0.812438(00:11:01)
0.2	0.981736(01:16:43)	0.965509(00:11:31)	0.946228(00:11:24)	0.965291(00:11:44)
0	0.986102(01:47:00)	0.983837(00:11:48)	0.973295(00:11:35)	0.982788(00:11:56)

* The misclassification rates of the Bayes classifiers are 29.368%, 25.586% and 23.982% for $\rho = 0.5$, $\rho = 0.2$, and $\rho = 0$, respectively.

* We implemented each experiment with 100 independent trials using C++ and ran them on 2 *Intel Xeon 6138 CPUs*. 35 cores were requested for each experiment.

4.6 The incremental algorithm with Iterative Independence Screening (IIS) and its variant approach

In this section, we describe the algorithm of the online method incorporating IIS for variable selection.

In the previous section, we have found that compared to the renewable estimation algorithm with SCAD, the method introducing IS for variable selection is more computationally efficient. These shortcomings of IS have been discussed (Fan et al., 2009), as it does not take into account the joint effects between predictor variables. Therefore, IIS, using marginal regressions, has been developed to address the limitations of IS (Fan et al., 2009). In this section, we outline how we employ the IIS method in conjunction with the penalty function within the framework of an online algorithm for analysing streaming data.

4.6.1 The incremental algorithm with IIS

We aim to find d, $1 \leqslant d < p$, variables through the iterative screening process. When the streaming data in the data batch $\mathcal{B}_b$, $b \geqslant 2$, is collected, firstly, the MMLEs of $\boldsymbol{\beta}$, denoted as $\hat{\boldsymbol{\beta}}_j^{\mathcal{M}}$, $j = 2, \cdots, p$, are calculated following (4.25). We then rank the absolute values of the estimates, excluding the intercept, and obtain the index set of the selected variables as:

$$\hat{\mathcal{M}}_1 = \{2 \leqslant j \leqslant p : |\hat{\beta}_j^{\mathcal{M}}| \geqslant \sigma_1\},$$

where $\sigma_1 > 0$ is a given constant. We then obtain r_1 variables, where $1 \leqslant r_1 < p$, based on the highest ranked absolute values of the MMLEs. To ensure that the selection process takes at least two iterations, r_1 is typically smaller than p. For the variables whose indices are in $\hat{\mathcal{M}}_1^c$, the conditional marginal regressions proceed as follows:

$$\hat{\boldsymbol{\beta}}_j^{\mathcal{CM}} = (\hat{\beta}_{j,1}^{\mathcal{CM}}, \hat{\beta}_j^{\mathcal{CM}}) = \underset{\beta_{j,1}^{\mathcal{CM}}, \beta_j^{\mathcal{CM}}, \boldsymbol{\beta}_{\hat{\mathcal{M}}_1}}{\arg\max} \sum_{i \in \mathcal{B}_b} \ell(\beta_1 + \beta_j^{\mathcal{CM}} x_{ji} + \mathbf{X}_{i,\hat{\mathcal{M}}_1}^T \boldsymbol{\beta}_{\hat{\mathcal{M}}_1}, y_i),$$

where $j \in \hat{\mathcal{M}}_1^c$.

We then rank the absolute values of these conditional MMLEs, excluding the intercept, and preserve the indices of the $d - r_1$ variables having top-ranked MMLEs in a set denoted as $\mathcal{A}_1$, with $|\mathcal{A}_1| = d - r_1$. The incremental algorithm introduced in Section 4.2.2 can be applied to the selected dataset, and at the k-th iteration, $k = 1, \cdots$, the estimator takes the form as

$$\begin{aligned}
(\tilde{\boldsymbol{\beta}}_{\mathcal{A}_1 \cup \hat{\mathcal{M}}_1})_b^{(k)} = {} & (\tilde{\boldsymbol{\beta}}_{\mathcal{A}_1 \cup \hat{\mathcal{M}}_1})_b^{(k-1)} - \left[\tilde{\mathbf{H}}_b((\tilde{\boldsymbol{\beta}}_{\mathcal{A}_1 \cup \hat{\mathcal{M}}_1})_{b-1}) - n_b \mathbf{P}''_{\lambda_b}(\|(\tilde{\boldsymbol{\beta}}_{\mathcal{A}_1 \cup \hat{\mathcal{M}}_1})_{b-1}\|_1)\right]^{-1} \\
& \times \Bigg\{ \left[\tilde{\mathbf{H}}_{b-1}((\tilde{\boldsymbol{\beta}}_{\mathcal{A}_1 \cup \hat{\mathcal{M}}_1})_{b-1}) - n_b \mathbf{P}''_{\lambda_b}(\|(\tilde{\boldsymbol{\beta}}_{\mathcal{A}_1 \cup \hat{\mathcal{M}}_1})_{b-1}\|)\right] \\
& \quad \times \big((\tilde{\boldsymbol{\beta}}_b)_{\mathcal{A}_1 \cup \hat{\mathcal{M}}_1}^{(k-1)} - (\tilde{\boldsymbol{\beta}}_{b-1})_{\mathcal{A}_1 \cup \hat{\mathcal{M}}_1}\big) \\
& \quad + \mathbf{S}_b((\tilde{\boldsymbol{\beta}}_{\mathcal{A}_1 \cup \hat{\mathcal{M}}_1})_b^{(k-1)}) - n_b \mathbf{P}'_{\lambda_b}(\|(\tilde{\boldsymbol{\beta}}_{\mathcal{A}_1 \cup \hat{\mathcal{M}}_1})_{b-1}\|_1) \Bigg\},
\end{aligned} \tag{4.28}$$

where $(\tilde{\boldsymbol{\beta}}_{\mathcal{A}_1 \cup \hat{\mathcal{M}}_1})_{b-1} = ((\tilde{\beta}_j)_{b-1})_{j \in (\mathcal{A}_1 \cup \hat{\mathcal{M}}_1)}$. When the iteration converges, we obtain the estimate $(\tilde{\boldsymbol{\beta}}_{\mathcal{A}_1 \cup \hat{\mathcal{M}}_1})_b$. $\hat{\mathcal{M}}_1$ is then updated to $\hat{\mathcal{M}}_2$, where $\hat{\mathcal{M}}_2 = \{j \in \mathcal{A}_1 \cup \hat{\mathcal{M}}_1 : |(\tilde{\beta}_j)_b| > 0\}$. The screening process continues until the l-th iteration, $l = 1, \cdots$, when $\hat{\mathcal{M}}_{l-1} = \hat{\mathcal{M}}_l$ or $|\hat{\mathcal{M}}_l| = d$. With the selected dataset, the renewable estimates can be

obtained following the penalised incremental algorithm in Section 4.2.2.

4.6.2 The incremental algorithm with IIS_{V1}: Variant 1 of IIS

In this section, we present a variant of IIS, referred to as IIS_{V1}, for analysing streaming data, which takes historical inferences into account during the screening process. When the stream data batch $\mathcal{B}_b$ is observed, we aim to find d, $1 \leqslant d < p$, significant variables.

The renewable MMLEs of $\boldsymbol{\beta}$, denoted as $(\tilde{\boldsymbol{\beta}}_j^{CM})_b = ((\tilde{\beta}_{j,1}^{CM})_b, (\tilde{\beta}_j^{CM})_b)^T$, $j = 2, \cdots, p$, are calculated following (4.27). The absolute values of the renewable MMLEs are ranked, and the index set containing the r_1 selected variables, where $1 \leqslant r_1 < p$, is formed as

$$\hat{\mathcal{M}}_1 = \{2 \leqslant j \leqslant p : |(\tilde{\beta}_j)_b^M| \geqslant \sigma_1\},$$

where $\sigma_1 > 0$ is a given constant as a threshold to select r_1 variables. The conditional renewable MMLEs are calculated for the variables whose indices are in $\hat{\mathcal{M}}_1^c$ and follows

$$\begin{aligned}
&(\tilde{\boldsymbol{\beta}}_j)_b^{CM(k)} \\
&= ((\tilde{\boldsymbol{\beta}}_j)_b^{CM(k-1)}, (\tilde{\boldsymbol{\beta}}_{\hat{\mathcal{M}}_1})_b) - \left[\tilde{\mathbf{H}}_{b-1}((\tilde{\boldsymbol{\beta}}_j)_{b-1}, (\tilde{\boldsymbol{\beta}}_{\hat{\mathcal{M}}_1})_{b-1}) + \mathbf{H}_b((\tilde{\boldsymbol{\beta}}_j)_b^{CM(k-1)}, (\tilde{\boldsymbol{\beta}}_{\hat{\mathcal{M}}_1})_b)\right]^{-1} \\
&\quad \times \Big\{\tilde{\mathbf{H}}_{b-1}((\tilde{\boldsymbol{\beta}}_j)_{b-1}, (\tilde{\boldsymbol{\beta}}_{\hat{\mathcal{M}}_1})_{b-1}) \times \left[((\tilde{\boldsymbol{\beta}}_j)_b^{CM(k-1)}, (\tilde{\boldsymbol{\beta}}_{\hat{\mathcal{M}}_1})_b) - ((\tilde{\boldsymbol{\beta}}_j)_{b-1}, (\tilde{\boldsymbol{\beta}}_{\hat{\mathcal{M}}_1})_{b-1})\right] \\
&\qquad + \mathbf{S}_b((\tilde{\boldsymbol{\beta}}_j)_b^{CM(k-1)}, (\tilde{\boldsymbol{\beta}}_{\hat{\mathcal{M}}_1})_b)\Big\},
\end{aligned}$$

where $(\tilde{\boldsymbol{\beta}}_{\hat{\mathcal{M}}_1})_{b-1} = ((\tilde{\boldsymbol{\beta}}_j)_{b-1})_{j \in \hat{\mathcal{M}}_1}$ and $(\tilde{\boldsymbol{\beta}}_{\hat{\mathcal{M}}})_b = ((\tilde{\boldsymbol{\beta}}_j)_b)_{j \in \hat{\mathcal{M}}}$.

After ranking the absolute values of the conditional MMLEs and identifying the top-ranked variables' indices, a set denoted as $\mathcal{A}_1$ is obtained, with $|\mathcal{A}_1| = d - r_1$. The penalised incremental method introduced in Section 4.2.2 is then applied to the dataset containing the variables whose indices are in $\mathcal{A}_1 \cup \hat{\mathcal{M}}_1$, and the computation process follows (4.28). The non-zero estimates of $\boldsymbol{\beta}$ are found, and $\hat{\mathcal{M}}_1$ is updated to $\hat{\mathcal{M}}_2$, with $\hat{\mathcal{M}}_2 = \{j \in \mathcal{A}_1 \cup \hat{\mathcal{M}}_1 : |(\tilde{\beta}_j)_b| > 0\}$. At the l-th iteration, $l = 1, \cdots$, when $\hat{\mathcal{M}}_{l-1} = \hat{\mathcal{M}}_l$ or $|\hat{\mathcal{M}}_l| = d$, we stop the screening process. When $\hat{\mathcal{M}}_l$ is obtained, the

penalised incremental algorithm introduced in Section 4.2.2 can be employed for the chosen dataset to obtain the renewable MLEs.

4.6.3 Simulation study: Comparative analysis of online algorithms with IS-SCAD, IS_{V1}-SCAD, IIS-SCAD, and IIS_{V1}-SCAD for the cases of $n_b < p < N_b$ and $p > N_b$

In this section, we conduct two designed examples to examine different sizes of high-dimensional streaming data and compare the performance of online algorithms utilising IS-SCAD, IS_{V1}-SCAD, IIS-SCAD, and IIS_{V1}-SCAD. In this section, as we only focus on the online algorithms, we simplify the names of the methods as IS-SCAD, IS_{V1}-SCAD, IIS-SCAD, and IIS_{V1}-SCAD. Additionally, it is worth mentioning that the method employing SCAD fails to converge when analysing the high-dimensional streaming data in this example. As a result, its performance is not recorded in this section. Similarly, based on the findings from the previous simulation study in Section 4.5.4, it has been observed that the performance of online algorithms utilising IS-SCAD and IS_{V2}-SCAD is very similar. Therefore, we have decided to omit conducting experiments on the method with IS_{V2}-SCAD.

For the two simulated examples in this section, the parameter is set as

$$\boldsymbol{\beta} = (0,\ \beta_2,\ \beta_3,\ \cdots,\ \beta_6,\ \beta_7,\ 0\cdots,\ 0)^T_{p\times 1},$$

$$\beta_2 = -0.8,\ \beta_3 = 1,\ \beta_6 = -0.8,\ \beta_7 = 1.$$

The correlation between the two covariates $x_{j_1 i}$ and $x_{j_2 i}$, where $i = 1, \cdots, N_b$, and $j_{1,2} = 2, \cdots, p$, is $\rho^{|j_1 - j_2|}$, and we consider $\rho = 0, 0.2, 0.5$.

Example 1: High-dimensional streaming data with $n_b < p < N_b$. In the first example, we set the full-size training dataset with $N_b = 1200$ data and the streaming data arriving in a size of $n_b = 300$. The dimension of the vector $\boldsymbol{\beta}_{p\times 1}$ is $p = 350$. We select $d = \lfloor n_b/(4 \times \log(n_b)) \rfloor$ in each screening process. Table 4.41-Table 4.44 presents the performance of the three methods.

Table 4.41 presents the AMRSEs and MMRSEs of estimates obtained from the three methods. The table illustrates that the AMRSEs of all estimates increase as the covariates become more correlated. Specifically, when analysing the same datasets,

IS_{V1}-SCAD exhibits superior performance compared to IS-SCAD, which is consistent with the trend observed in the simulation study discussed in Section 4.5.4. Across different cases, IIS-SCAD can generate more accurate renewable estimates than the methods using IS-SCAD and IS_{V1}-SCAD. As the data becomes more correlated, the gaps between the estimated coefficients $\tilde{\boldsymbol{\beta}}_{\text{IIS-SCAD}}$s, $\tilde{\boldsymbol{\beta}}_{\text{IS-SCAD}}$s, and $\tilde{\boldsymbol{\beta}}_{\text{IS}_{\text{V1}}\text{-SCAD}}$s widen. The numerical results indicate that $\tilde{\boldsymbol{\beta}}_{\text{IIS-SCAD}}$s consistently exhibit stable performance across various correlated covariates, as evidenced by the close MMRSEs of $\tilde{\boldsymbol{\beta}}_{\text{IIS-SCAD}}$s observed across different cases.

Table 4.42 presents the ESEs calculated using (4.14) and the SDs of the estimates obtained from the three methods. The table reveals a similar phenomenon to the simulation study discussed in Section 4.5.4, wherein the methods utilising IS and its variant denoted as IS_{V1}, can miss significant features when the covariates are highly correlated. However, in the case of estimates obtained through the renewable method with IIS-SCAD, the ESEs calculated using Equation (4.14) perform well and closely match the SDs in all cases.

Table 4.43 records the performance of the online algorithms with IS-SCAD, IS_{V1}-SCAD, and IIS-SCAD, considering the incorrect non-zero and zero estimates and CPs for non-zero and zero coefficients of $\boldsymbol{\beta}$ when studying datasets with different correlated covariates. The table clearly demonstrates that the methods using IS cannot find the important features when the data is more correlated, as shown in the table by the increasing incorrect zero estimates with the increase in ρ. As the correlation parameter ρ increases, the values of CPs for non-zero coefficients from IS-SCAD and IS_{V1}-SCAD approach zero, indicating less accurate estimation results. However, in the 100 repeated simulations, IIS-SCAD demonstrates noticeably fewer missed important features. This highlights the advantage of IIS over IS in variable selection, particularly when dealing with data that has highly correlated covariates.

The excess risks of classifiers presented in Table 4.44 demonstrate a consistent trend, indicating that as the correlation of the covariates increases, the efficiency of all classifiers decreases. However, the classifier $C_{\tilde{\boldsymbol{\beta}}_{\text{IIS-SCAD}}}$ exhibits more stable performance compared to the other two classifiers trained using the methods employing IS-SCAD and its variant across different cases. Furthermore, the excess risks of $C_{\tilde{\boldsymbol{\beta}}_{\text{IIS-SCAD}}}$s remain stable and close to 1. In addition, $C_{\tilde{\boldsymbol{\beta}}_{\text{IIS-SCAD}}}$s also exhibit very close efficiency to the Bayes classifiers across all cases. On the other hand, $C_{\tilde{\boldsymbol{\beta}}_{\text{IS-SCAD}}}$s

and $C_{\tilde{\boldsymbol{\beta}}_{\mathrm{IS_{V1}-SCAD}}}$s show a noticeable decrease in efficiency as ρ increases. However, when trained on the same datasets, $C_{\tilde{\boldsymbol{\beta}}_{\mathrm{IS_{V1}-SCAD}}}$s consistently outperform $C_{\tilde{\boldsymbol{\beta}}_{\mathrm{IS-SCAD}}}$s. It is worth noting that the online algorithm with IIS-SCAD, which involves additional screening processes for the conditional MMLEs, requires more time to study the same dataset compared to the algorithms utilising IS-SCAD and its variant.

Table 4.41: AMRSEs and MMRSEs (in brackets) of $\tilde{\boldsymbol{\beta}}_{\mathrm{IS-SCAD}}$s, $\tilde{\boldsymbol{\beta}}_{\mathrm{IS_{V1}-SCAD}}$s and $\tilde{\boldsymbol{\beta}}_{\mathrm{IIS-SCAD}}$s

ρ	$\tilde{\boldsymbol{\beta}}_{\mathrm{IS-SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\mathrm{IS_{V1}-SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\mathrm{IIS-SCAD}}$s
0.5	1.786(1.636)	1.264(1.471)	0.201(0.025)
0.2	0.402(0.568)	0.205(0.031)	0.097(0.030)
0	0.100(0.024)	0.089(0.022)	0.066(0.026)

* All values recorded in this table multiplied by 10^{-3} are true values.

Table 4.42: SDs and ESEs for non-zero $\boldsymbol{\beta}_{350\times1}$ coefficient estimates

ρ		$\tilde{\boldsymbol{\beta}}_{\mathrm{IS-SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\mathrm{IS_{V1}-SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\mathrm{IIS-SCAD}}$
0.5	β_2	1.1861(0,0)	1.1861(0,0)	0.0851(0.0859,0.1273)
	β_3	0.8326(0.0652,0.0966)	0.7594(0.0671,0.0995)	0.1041(0.0899,0.1333)
	β_6	1.1861(0,0)	1.1861(0,0)	0.0864(0.0869,0.1288)
	β_7	0.7291(0.0663,0.0983)	0.6896(0.0679,0.1007)	0.0939(0.0902,0.1337)
0.2	β_2	0.1358(0.0789,0.1170)	0.1002(0.0803,0.1191)	0.0888(0.0825,0.1223)
	β_3	0.1385(0.0825,0.1223)	0.1090(0.0845,0.1252)	0.1016(0.0875,0.1297)
	β_6	0.1425(0.0786,0.1165)	0.0958(0.0809,0.1200)	0.1059(0.0830,0.1230)
	β_7	0.1459(0.0816,0.1210)	0.1150(0.0843,0.1250)	0.1028(0.0874,0.1295)
0	β_2	0.0756(0.0823,0.1220)	0.0790(0.0823,0.1220)	0.0831(0.0847,0.1255)
	β_3	0.0848(0.0861,0.1276)	0.0833(0.0864,0.1280)	0.1011(0.0891,0.1321)
	β_6	0.0843(0.0821,0.1217)	0.0857(0.0823,0.1220)	0.0946(0.0846,0.1254)
	β_7	0.0975(0.0859,0.1273)	0.0871(0.0859,0.1274)	0.1039(0.0884,0.1310)

* For each cell, the value outside the brackets represents SD, while the first and second numbers inside the brackets represent $\mathrm{ESE_m}$ and $\mathrm{ESE_o}$, respectively.

Table 4.43: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$

ρ		$\tilde{\boldsymbol{\beta}}_{\mathrm{IS-SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\mathrm{IS}_{\mathrm{V1}}-\mathrm{SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\mathrm{IIS-SCAD}}$
0.5	non0 coeff.	2.27(0.015000)	1.52(0.230000)	0.11(0.875000)
	0 coeff.	0.70(0.999595)	1.01(0.999422)	0.56(0.999740)
0.2	non0 coeff.	0.56(0.690000)	0.25(0.835000)	0.01(0.902500)
	0 coeff.	0.58(0.999740)	0.64(0.999624)	0.55(0.999682)
0	non0 coeff.	0.08(0.910000)	0.09(0.930000)	0(0.922500)
	0 coeff.	0.51(0.999827)	0.51(0.999855)	0.61(0.999855)

* $\boldsymbol{\beta}_{350\times1}$ has 4 non-zero coefficients.

Table 4.44: Excess risks of various classifiers and computing time

ρ	$\mathrm{ER}(C_{\tilde{\boldsymbol{\beta}}_{\mathrm{IS-SCAD}}})$	$\mathrm{ER}(C_{\tilde{\boldsymbol{\beta}}_{\mathrm{IS}_{\mathrm{V1}}-\mathrm{SCAD}}})$	$\mathrm{ER}(C_{\tilde{\boldsymbol{\beta}}_{\mathrm{IIS-SCAD}}})$
0.5	0.771421(00:29:04)	0.832238(00:28:44)	0.978803(00:40:00)
0.2	0.923349(00:29:03)	0.960508(00:28:49)	0.986962(00:37:32)
0	0.979017(00:29:08)	0.979897(00:28:30)	0.988867(00:37:02)

* The misclassification rates of the Bayes classifiers are 29.368%, 25.586% and 23.982% for $\rho = 0.5$, $\rho = 0.2$, and $\rho = 0$, respectively.

* We implemented each experiment with 100 independent trials using C++ and ran them on 2 *Intel Xeon 6138 CPUs*. 35 cores were requested for each experiment.

Example 2: High-dimensional streaming data with $p > N_b$. In **Example 2**, we evaluate the efficiency of incremental algorithms that employ varying variable selection techniques including IS-SCAD, $\mathrm{IS}_{\mathrm{V1}}$-SCAD, IIS-SCAD, and $\mathrm{IIS}_{\mathrm{V1}}$-SCAD.

Each dataset comprises $p = 1900$ predictors, with a training dataset size of $N_b = 1800$. The streaming data is recorded in data batches of size n_b. This example examines two scenarios, one with $n_b = 300$ and the other with $n_b = 600$. In the screening process, for the case of $n_b = 300$, $\lfloor n_b/(3 \times \log(n_b)) \rfloor$ variables are chosen, and for $n_b = 600$, $\lfloor n_b/(4 \times \log(n_b)) \rfloor$ variables are chosen. The numerical results for this example are shown in Tables 4.45 through 4.47.

The AMRSEs and MMRSEs of the estimates obtained from the four different variable selection methods are presented in Table 4.45. The results confirm similar conclusions as in **Example 1**, indicating that as the correlation parameter of the covariates increases, the estimation accuracy of all tested methods decreases. When

analysing the same streaming data, it is important to note that in each case, $\tilde{\boldsymbol{\beta}}_{\text{IIS}-\text{SCAD}}$s and $\tilde{\boldsymbol{\beta}}_{\text{IIS}_{\text{V1}}-\text{SCAD}}$s consistently outperform $\tilde{\boldsymbol{\beta}}_{\text{IS}-\text{SCAD}}$s and $\tilde{\boldsymbol{\beta}}_{\text{IS}_{\text{V1}}-\text{SCAD}}$s, with only a slight exception observed when $n_b = 300$ and $\rho = 0$. Furthermore, as the correlation of the covariates increases, the differences between the estimates of the renewable methods using IS and those using IIS become more pronounced. In this example, $\tilde{\boldsymbol{\beta}}_{\text{IIS}_{\text{V1}}-\text{SCAD}}$s demonstrate either comparable or better performance than $\tilde{\boldsymbol{\beta}}_{\text{IIS}-\text{SCAD}}$s across different cases. Additionally, it is worth noting that for the same full-size training datasets, increasing the streaming data size improves the performance of all tested methods, although the improvement in the algorithm using IIS_{V1}-SCAD is less significant compared to the others. Overall, the performance of IIS_{V1}-SCAD exhibits greater stability across different training datasets.

The ESEs obtained using (4.14) and the corresponding SDs of the renewable estimates are presented in Table C.9 in Appendix C.1. The performance of variable selection and the CPs for the coefficients of $\boldsymbol{\beta}$ are shown in Table 4.46. Similar to **Example 1**, the results indicate that datasets with more correlated covariates lead to poorer performance for IS-SCAD and IS_{V1}-SCAD. This is evidenced by a remarkable increase in incorrect zero estimates and a notable decrease in CPs for non-zero coefficients, approaching zero, especially when the size of the streaming data is small. In contrast, although the performance of IIS-SCAD and IISV1-SCAD deteriorates as the covariates become more correlated, they still outperform IS-SCAD and its variant, with fewer missed important features and less variation in the CPs for non-zero coefficients. The results in the table consistently support the conclusion of Table 4.45: when studying the same training dataset, increasing the size of the streaming data improves the performance of variable selection methods, albeit with less pronounced effects on IIS-SCAD and IISV1-SCAD compared to IS-SCAD and ISV1-SCAD.

Table 4.47 shows the excess risks of the classifiers and the computation time of the estimation processes. It is observed that all classifiers trained on more correlated data exhibit lower efficiency compared to those trained on less correlated data. Consistent with **Example 1**, the methods utilising IIS-SCAD generate more efficient classifiers compared to those using IS-SCAD and its extensions. Notably, the classifiers obtained through IIS_{V1}-SCAD outperform those obtained through IIS-SCAD, particularly when the size of the streaming data is smaller ($n_b = 300$). In terms of computation time, there is no noticeable deviation among the estimation methods in this section.

However, the methods employing IIS-SCAD and its variant generally require slightly more time compared to IS-SCAD and IS_{V1}-SCAD. Considering classifier efficiency, we recommend utilising IIS or its extension for studying high-dimensional data with an incremental algorithm.

Table 4.45: AMRSEs and MMRSEs (in brackets) of $\tilde{\boldsymbol{\beta}}_{\mathrm{IS-SCAD}}$s, $\tilde{\boldsymbol{\beta}}_{\mathrm{IS_{V1}-SCAD}}$s, $\tilde{\boldsymbol{\beta}}_{\mathrm{IIS-SCAD}}$s and $\tilde{\boldsymbol{\beta}}_{\mathrm{IIS_{V1}-SCAD}}$s.

	ρ	$\tilde{\boldsymbol{\beta}}_{\mathrm{IS-SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\mathrm{IS_{V1}-SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\mathrm{IIS-SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\mathrm{IIS_{V1}-SCAD}}$
$n_b = 300$	0.5	0.389(0.399)	0.291(0.293)	0.097(0.004)	0.053(0.003)
	0.2	0.194(0.210)	0.065(0.007)	0.039(0.003)	0.037(0.003)
	0	0.078(0.005)	0.019(0.003)	0.042(0.004)	0.029(0.004)
$n_b = 600$	0.5	0.299(0.264)	0.200(0.146)	0.070(0.004)	0.070(0.004)
	0.2	0.054(0.004)	0.055(0.003)	0.043(0.004)	0.043(0.004)
	0	0.030(0.003)	0.031(0.003)	0.018(0.005)	0.018(0.005)

* All values recorded in this table multiplied by 10^{-3} are true values.

Table 4.46: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$

ρ	$n_b = 300$	$\tilde{\boldsymbol{\beta}}_{\mathrm{IS-SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\mathrm{IS_{V1}-SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\mathrm{IIS-SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\mathrm{IIS_{V1}-SCAD}}$
0.5	non0 coeff.	2.65(0.010000)	2.05(0.070000)	0.55(0.770000)	0.29(0.835000)
	0 coeff.	0.74(0.999794)	0.32(0.999984)	0.64(0.999842)	0.45(0.999921)
0.2	non0 coeff.	1.53(0.275000)	0.50(0.667500)	0.12(0.815000)	0.17(0.830000)
	0 coeff.	0.30(0.999989)	0.44(0.999953)	0.30(0.999963)	0.30(0.999963)
0	non0 coeff.	0.39(0.757500)	0.10(0.900000)	0.19(0.825000)	0.12(0.855000)
	0 coeff.	0.24(0.999989)	0.44(0.999931)	0.32(0.999958)	0.33(0.999958)
ρ	$n_b = 600$	$\tilde{\boldsymbol{\beta}}_{\mathrm{IS-SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\mathrm{IS_{V1}-SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\mathrm{IIS-SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\mathrm{IIS_{V1}-SCAD}}$
0.5	non0 coeff.	1.73(0.082500)	1.65(0.352500)	0.15(0.787500)	0.24(0.795000)
	0 coeff.	0.18(1.000000)	0.55(0.999863)	0.30(0.999963)	0.28(0.999958)
0.2	non0 coeff.	0.09(0.820000)	0.30(0.822500)	0.05(0.807500)	0.11(0.815000)
	0 coeff.	0.22(0.999984)	0.23(0.999979)	0.25(0.999963)	0.26(0.999958)
0	non0 coeff.	0.01(0.880000)	0.12(0.880000)	0(0.817500)	0.02(0.825000)
	0 coeff.	0.33(0.999979)	0.33(0.999979)	0.39(0.999979)	0.39(0.999979)

* $\boldsymbol{\beta}_{1900\times 1}$ has 4 non-zero coefficients.

Table 4.47: Excess risks of various classifiers and computing time

	ρ	$ER(C_{\tilde{\beta}_{IS-SCAD}})$	$ER(C_{\tilde{\beta}_{IIS_{V1}-SCAD}})$	$ER(C_{\tilde{\beta}_{IIS-SCAD}})$	$ER(C_{\tilde{\beta}_{IIS_{V1}-SCAD}})$
	0.5	0.727892(12:30:29)	0.783857(12:39:28)	0.924820(14:13:02)	0.958311(13:18:13)
$n_b = 300$	0.2	0.807983(12:46:00)	0.924423(13:00:58)	0.973545(13:20:51)	0.971095(13:08:24)
	0	0.926730(11:56:05)	0.976220(12:14:38)	0.962050(13:16:24)	0.972061(12:43:13)
	ρ	$ER(C_{\tilde{\beta}_{IS-SCAD}})$	$ER(C_{\tilde{\beta}_{IIS_{V1}-SCAD}})$	$ER(C_{\tilde{\beta}_{IIS-SCAD}})$	$ER(C_{\tilde{\beta}_{IIS_{V1}-SCAD}})$
	0.5	0.796841(12:51:50)	0.791976(12:44:38)	0.961801(15:31:57)	0.960212(14:18:27)
$n_b = 600$	0.2	0.997623(12:54:15)	0.947678(12:43:03)	0.979989(16:10:55)	0.975784(14:24:43)
	0	0.983911(12:37:26)	0.968806(12:19:14)	0.991145(15:31:23)	0.988255(14:02:25)

* When $\rho = 0.5$, the misclassification rate of Bayes classifier is 29.056%; when $\rho = 0.2$, the misclassification rate of Bayes classifier is 25.466%; when $\rho = 0$ the misclassification rate of Bayes classifier is 23.728%.

* We implemented each experiment with 100 independent trials using C++ and ran them on 2 *Intel Xeon 6138 CPUs*. 35 cores were requested for each experiment.

In conclusion, the simulated examples in this section highlight the improved performance of IIS-SCAD and its variants, IIS_{V1}-SCAD, which incorporates conditional marginal regressions, when applied to high-dimensional streaming data. IIS has been proven that it can improve the limitations of IS (Fan and Lv, 2008, Fan and Song, 2010). The simulation study of our study also demonstrates that the online algorithm utilising IIS-SCAD or IIS_{V1}-SCAD consistently exhibits stable estimation accuracy across varying correlated covariates and streaming data sizes, achieving comparable efficiency to the Bayes classifier. In addition, the computing time of IIS-SCAD or IIS_{V1}-SCAD does not exhibit a substantial increase compared to the online methods employing IS-SCAD or IS_{V1}-SCAD. The renewable estimation method employing IIS_{V1}-SCAD demonstrates slightly superior performance compared to the method utilising IIS-SCAD in terms of variable selection. It exhibits fewer incorrect non-zero and zero estimates and achieves higher CPs for coefficients. However, it should be noted that the difference between the two methods is not substantial.

4.6.4 Simulation study: Comparative analysis of online algorithms with IIS-SCAD and IIS_{V1}-SCAD, along with the offline algorithm with IIS-SCAD for the case of $p > N_b$

In this section, we employ several simulated examples to compare the effectiveness of online algorithms that integrate IIS-SCAD and IIS_{V1}-SCAD with an offline algorithm

that utilises IIS-SCAD. In this section, we denote the estimates of the two online methods as $\tilde{\boldsymbol{\beta}}_{\text{IIS}-\text{SCAD}}$s and $\tilde{\boldsymbol{\beta}}_{\text{IIS}_{\text{V1}}-\text{SCAD}}$s and the estimate of the offline method as $\hat{\boldsymbol{\beta}}^{*}_{\text{IIS}-\text{SCAD}}$s. These comparisons are performed on various types of high-dimensional streaming data.

To be more specific, the first example focuses on covariates exhibiting different levels of correlation. The second example investigates the impact of varying sizes of streaming data. Lastly, the third example explores the performance across different full-size training datasets. All examples share the same parameter set, which is set as

$$\boldsymbol{\beta} = (0,\ \beta_2,\ \beta_3,\ \cdots,\ \beta_6,\ \beta_7,\ 0\cdots,\ 0)^{\mathsf{T}}_{p\times 1},$$

$$\beta_2 = -0.8,\ \beta_3 = 1,\ \beta_6 = -0.8,\ \beta_7 = 1,$$

and we set $p = 3000$. The correlation between the two covariates $x_{j_1 i}$ and $x_{j_2 i}$, where $i = 1, \cdots, N_b$, and $j_{1,2} = 2, \cdots, p$, is $\rho^{|j_1 - j_2|}$.

Example 1: varying correlated covariates. In **Example 1**, we study different datasets where the correlation parameter of covariates varies, testing $\rho = 0, 0.2, 0.5$ respectively.

We have each full-size training dataset with a size of $N_b = 2500$ and the streaming data comes in a fixed size of $n_b = 500$. For the iterative screening process, both the incremental algorithms select $\lfloor n_b/(5 \times \log(n_b)) \rfloor$ variables, and the offline method selects $\lfloor N_b/(20 \times \log(N_b)) \rfloor$ variables. Table 4.48-4.49 record the numerical results for **Example 1**.

In Table 4.48, the results show that, when trained on the same datasets, the renewable estimates, $\tilde{\boldsymbol{\beta}}_{\text{IIS}-\text{SCAD}}$s and $\tilde{\boldsymbol{\beta}}_{\text{IIS}_{\text{V1}}-\text{SCAD}}$s, exhibit better overall performance than the $\hat{\boldsymbol{\beta}}^{*}_{\text{IIS}-\text{SCAD}}$s of the offline method. Specifically, the AMRSEs of $\tilde{\boldsymbol{\beta}}_{\text{IIS}-\text{SCAD}}$s and $\tilde{\boldsymbol{\beta}}_{\text{IIS}_{\text{V1}}-\text{SCAD}}$s are much smaller than those of $\hat{\boldsymbol{\beta}}^{*}_{\text{IIS}-\text{SCAD}}$s. While the MMRSEs of all the estimates are close. In this example, the performance of $\tilde{\boldsymbol{\beta}}_{\text{IIS}-\text{SCAD}}$s and $\tilde{\boldsymbol{\beta}}_{\text{IIS}_{\text{V1}}-\text{SCAD}}$s show little difference. The numerical results indicate that the varying correlation of covariates has minimal influence on the performance of the incremental algorithms incorporating IIS-SCAD and its extension. These algorithms consistently yield similar outcomes regardless of the correlation present in the data.

Table C.10 displays the ESEs and SDs of the estimates obtained from the renewable

and offline estimation methods, respectively, as given in (4.14) and (4.4). These results can be found in Appendix C.1. Table 4.49 displays the performance of variable selection for the three tested methods. It is evident that when compared with the two renewable estimation methods using IIS-SCAD or its variant, the offline method using IIS-SCAD has the highest number of incorrect estimates for zero coefficients across different cases, and CPs from the offline method for non-zero coefficients of $\boldsymbol{\beta}$ are the lowest in this example. On the contrary, as the data becomes more correlated, the online algorithms only miss a few significant features, and the CPs for the nonzero coefficients of $\boldsymbol{\beta}$ from both online algorithms remain consistent, whereas the CPs from the offline method exhibit a noticeable decrease. This phenomenon can be attributed to the advantage that online methods have in analysing streaming data. These methods have more time for the screening process, which enables them to effectively select variables that are better suited for the data. Based on these findings, we conclude that our proposed methods outperform the offline method in terms of variable selection for highly correlated high-dimensional streaming data.

Table 4.50 presents the computing time and memory usage for each experiment comprising 100 trials, as well as the efficiency of the classifiers trained by the three methods. In this example, the efficiency of both renewable classifiers is higher than that of the offline method in all cases. Additionally, $\mathrm{C}_{\tilde{\boldsymbol{\beta}}_{\mathrm{IIS-SCAD}}}$s and $\mathrm{C}_{\tilde{\boldsymbol{\beta}}_{\mathrm{IIS_{V1}-SCAD}}}$s perform very close to the Naive Bayes classifier. Consistent with the previous simulation study, the offline method requires more time than both incremental algorithms under the same computation settings. Another advantage that has been once again demonstrated in this example is that online methods require noticeably less time and computing resources (as indicated by memory utilisation) compared to offline methods when studying the same datasets. Both incremental algorithms exhibit similar computing times for different correlated data. However, the online algorithm incorporating $\mathrm{IIS_{V1}}$-SCAD requires slightly less time and utilises fewer computing resources compared to the online method using IIS-SCAD.

Table 4.48: AMRSEs and MMRSEs (in brackets) of $\tilde{\boldsymbol{\beta}}_{\mathrm{IIS-SCAD}}$s, $\tilde{\boldsymbol{\beta}}_{\mathrm{IIS_{V1}-SCAD}}$s and $\hat{\boldsymbol{\beta}}^{*}_{\mathrm{IIS-SCAD}}$s

ρ	$\tilde{\boldsymbol{\beta}}_{\mathrm{IIS-SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\mathrm{IIS_{V1}-SCAD}}$	$\hat{\boldsymbol{\beta}}^{*}_{\mathrm{IIS-SCAD}}$
0.5	0.011(0.001)	0.011(0.001)	0.046(0.001)
0.2	0.003(0.001)	0.004(0.001)	0.043(0.001)
0	0.003(0.001)	0.003(0.002)	0.044(0.001)

* All values recorded in this table multiplied by 10^{-3} are true values.

* Each training dataset consists of $N_b = 2500$ data. In the renewable estimation experiments, each data stream batch has a size of $n_b = 500$.

Table 4.49: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$

ρ		$\tilde{\boldsymbol{\beta}}_{\mathrm{IIS-SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\mathrm{IIS_{V1}-SCAD}}$	$\hat{\boldsymbol{\beta}}^{*}_{\mathrm{IIS-SCAD}}$
0.5	non0 coeff.	0.04(0.890000)	0.07(0.905000)	0(0.820000)
	0 coeff.	0.53(0.999993)	0.52(0.999997)	11.33(0.999997)
0.2	non0 coeff.	0(0.932500)	0.01(0.937500)	0(0.840000)
	0 coeff.	0.61(0.999987)	0.61(0.999987)	10.70(0.996429)
0	non0 coeff.	0(0.920000)	0.01(0.922500)	0(0.920000)
	0 coeff.	0.46(0.999977)	0.46(0.999977)	11.63(0.997553)

* $\boldsymbol{\beta}_{3000\times 1}$ has 4 non-zero coefficients.

Table 4.50: Excess risks of $C_{\tilde{\boldsymbol{\beta}}_{\mathrm{IIS-SCAD}}}$s, $C_{\tilde{\boldsymbol{\beta}}_{\mathrm{IIS_{V1}-SCAD}}}$s and $C_{\hat{\boldsymbol{\beta}}_{\mathrm{IIS-SCAD}}}$s, and computing time and memory utilisations (in brackets)

ρ	$ER(C_{\tilde{\boldsymbol{\beta}}_{\mathrm{IIS-SCAD}}})$	$ER(C_{\tilde{\boldsymbol{\beta}}_{\mathrm{IIS_{V1}-SCAD}}})$	$ER(C_{\hat{\boldsymbol{\beta}}_{\mathrm{IIS-SCAD}}})$
0.5	0.992128(33:13:48,5.11 GB)	0.991111(31:42:22,3.72 GB)	0.965880(51:29:53,22.19 GB)
0.2	0.995260(33:11:43,5.49 GB)	0.994004(31:23:55,3.57 GB)	0.980925(50:02:57,22.88 GB)
0	0.999661(32:41:44,5.80 GB)	0.998224(30:50:31,3.57 GB)	0.963828(44:29:02,17.63 GB)

* When $\rho = 0.5$, the misclassification rate of Bayes classifier is 28.988%; when $\rho = 0.2$, the misclassification rate of Bayes classifier is 25.198%; when $\rho = 0$, the misclassification rate of Bayes classifier is 23.608%.

* We implemented each experiment with 100 independent trials using C++ and ran them on 2 *Intel Xeon 6138 CPUs*. 40 cores were requested for each experiment.

* For each cell in the table, the value outside the brackets represents the excess risk of the classifier. The first and second numbers inside the brackets denote the computing time of each experiment consisting of 100 trials and the memory utilisation, respectively.

Example 2: varying streaming data sizes In the second simulated example, we investigate how different streaming data sizes affect the performance of the renewable methods. Since the training datasets are the same with a fixed size of $N_b = 2000$, the performance of the offline method remains constant in this example. Specifically, for the renewable estimation methods, we test the sizes of the streaming data, $n_b = 400,\ 500,\ 1000$. We examine covariates with strong correlation and set the correlation parameter $\rho = 0.5$. In the screening process, for the renewable estimation methods, $\lfloor n_b/(4 \times \log(n_b)) \rfloor$, $\lfloor n_b/(5 \times \log(n_b)) \rfloor$ and $\lfloor n_b/(9 \times \log(n_b)) \rfloor$ variables are selected for the case where the streaming data comes in sizes of $n_b = 400, 500, 1000$ respectively, while for the offline method, $\lfloor n_b/(10 \times \log(n_b)) \rfloor$ variables are selected. Table 4.51-Table 4.53 record the outcomes for this case.

In Table 4.51, the results show the performance of $\tilde{\boldsymbol{\beta}}_{\text{IIS}-\text{SCAD}}$s, $\tilde{\boldsymbol{\beta}}_{\text{IIS}_{\text{V1}}-\text{SCAD}}$s, and $\hat{\boldsymbol{\beta}}^*_{\text{IIS}-\text{SCAD}}$s. Presetting fair settings for searching the tuning parameters λs of SCAD guarantees that desirable values of the λs can be achieved and have a less negative influence on the results. Clearly, $\tilde{\boldsymbol{\beta}}_{\text{IIS}-\text{SCAD}}$s and $\tilde{\boldsymbol{\beta}}_{\text{IIS}_{\text{V1}}-\text{SCAD}}$s outperform $\hat{\boldsymbol{\beta}}^*_{\text{IIS}-\text{SCAD}}$s. Trained by the same datasets, the performance of $\tilde{\boldsymbol{\beta}}_{\text{IIS}-\text{SCAD}}$s and $\tilde{\boldsymbol{\beta}}_{\text{IIS}_{\text{V1}}-\text{SCAD}}$s are almost the same in this case. The accuracy of the renewable estimates is less affected by the size of the streaming data batches, as evidenced by the similar MMRSEs and close AMRSEs in all cases.

Table C.11 records the ESEs and SDs and can be found in Appendix C.1. The performance of variable selection for each method is presented in Table 4.52. The offline method using IIS-SCAD finds all the important features while it performs the worst with the largest numbers of incorrect non-zero estimates for zero coefficients of $\boldsymbol{\beta}$ and with the lowest CPs for non-zero coefficients when compared with other renewable estimates. Both renewable estimation methods have few numbers of incorrect zero estimates and non-zero estimates across different cases. Specifically, when $n_b = 400$, $\tilde{\boldsymbol{\beta}}_{\text{IIS}_{\text{V1}}-\text{SCAD}}$ has the highest number of missing significant features, while with the increase in the size of the streaming data, the improvement is evident. $\tilde{\boldsymbol{\beta}}_{\text{IIS}_{\text{V1}}-\text{SCAD}}$s have the least incorrect non-zero estimates, which shows the renewable estimation method using IIS_{V1}-SCAD generates better interpretable models compared with the other two methods.

The excess risks of the three classifiers, as well as the corresponding computing time and memory utilisation, are recorded in Table 4.53. As for the two incremental

algorithms, when studying the same datasets, their computing time is very close to each other. In this example, the renewable classifiers perform closely to the Naive Bayes classifier and are more efficient than the traditional offline classifiers. Both renewable classifiers have stable performance with the change in sizes of the streaming data.

The most noticeable difference among the three methods is that, compared with both renewable estimation methods, the offline method needs more than twice the time to study the same datasets even if more computing resources are allocated.

Table 4.51: AMRSEs and MMRSEs (in brackets) of estimates

n_b	$\tilde{\beta}_{\mathrm{IIS-SCAD}}$	$\tilde{\beta}_{\mathrm{IIS_{V1}-SCAD}}$	$\hat{\beta}^{*}_{\mathrm{IIS-SCAD}}$
400	0.014(0.002)	0.015(0.002)	0.065(0.056)
500	0.012(0.002)	0.011(0.002)	0.065(0.056)
1000	0.012(0.002)	0.012(0.002)	0.065(0.056)

* All values recorded in this table multiplied by 10^{-3} are true values.

* The size of full training data is $N_b = 2000$.

Table 4.52: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for β

n_b		$\tilde{\beta}_{\mathrm{IIS-SCAD}}$	$\tilde{\beta}_{\mathrm{IIS_{V1}-SCAD}}$	$\hat{\beta}^{*}_{\mathrm{IIS-SCAD}}$
400	non0 coeff.	0.08(0.910000)	0.11(0.912500)	0(0.790000)
	0 coeff.	0.67(0.999977)	0.67(0.999977)	22.76(0.995441)
500	non0 coeff.	0.04(0.917500)	0.07(0.922500)	0(0.790000)
	0 coeff.	0.58(0.999983)	0.55(0.999983)	22.76(0.995441)
1000	non0 coeff.	0.01(0.897500)	0.01(0.892500)	0(0.790000)
	0 coeff.	1.03(0.999930)	0.78(0.999930)	22.76(0.995441)

* $\beta_{3000\times 1}$ has 4 non-zero coefficients.

Table 4.53: Excess risks of $C_{\tilde{\beta}_{IIS-SCAD}}$s, $C_{\tilde{\beta}_{IIS_{V1}-SCAD}}$s and $C_{\hat{\beta}^*_{IIS-SCAD}}$s, and computing time and memory utilisations (in brackets)

n_b	$ER(C_{\tilde{\beta}_{IIS-SCAD}})$	$ER(C_{\tilde{\beta}_{IIS_{V1}-SCAD}})$	$ER(C_{\hat{\beta}_{IIS-SCAD}})$
400	0.985517(32:16:05,4.53 GB)	0.982977(31:18:03,3.21 GB)	0.923126(52:19:39,22.36 GB)
500	0.988947(33:03:17,5.06 GB)	0.987330(31:35:39,3.66 GB)	0.923126(52:19:39,22.36 GB)
1000	0.991857(35:35:47,8.66 GB)	0.992944(34:24:24,6.76 GB)	0.923126(52:19:39,22.36 GB)

* The misclassification rate of Bayes classifier is 28.988%

* We implemented each experiment with 100 independent trials using C++ and ran them on 2 *Intel Xeon 6138 CPUs*. 40 cores were requested for each experiment.

* For each cell in the table, the value outside the brackets represents the excess risk of the classifier. The first and second numbers inside the brackets denote the computing time of each experiment consisting of 100 trials and the memory utilisation, respectively.

Example 3: varying training data sizes. In the last example of this section, we compare our renewable estimation methods with the offline method by using different sizes of training datasets. Specifically, the full-size training datasets with sizes of $N_b = 1000,\ 1500,\ 2000,\ 2500,$ are studied separately. The streaming data comes in a fixed size of $n_b = 500$ and the correlation parameter is set as $\rho = 0.5$. In the screening process, for the renewable estimation methods, $\lfloor n_b/(5 \times \log(n_b))\rfloor$ variables are selected, and for the offline estimation methods, $\lfloor n_b/(9 \times \log(n_b))\rfloor$, $\lfloor n_b/(13\times\log(n_b))\rfloor$, $\lfloor n_b/(15\times\log(n_b))\rfloor$ and $\lfloor n_b/(20\times\log(n_b))\rfloor$ variables are selected for the training datasets with a size of $N_b = 1000,\ 1500,\ 2000,\ 2500,$ respectively. Under this setting of the offline method, the numbers of selected variables are close across different cases. Table 4.54-Table 4.56 record the numerical results for this case.

Table 4.54 presents the AMRSEs and MMRSEs of the renewable estimates and the estimates from the offline method. The results show that the accuracy of all estimates improves when the size of the training data increases, which can be seen by the decrease of MMRSEs of the estimates. As observed in the previous examples, **Example 1** and **Example 2**, both renewable estimates demonstrate substantially better overall performance than the offline estimates, with much lower AMRSEs, when trained by the same datasets. When the training data size increases, the difference between the offline method and the renewable methods becomes smaller, but the renewable methods still exhibit noticeably better performance than the offline method.

Table C.12 records the ESEs and SDs from the three estimation methods and is included in Appendix C.1. As seen in Table 4.55, all methods exhibit good performance in variable selection. In detail, the incorrect zero estimates of $\hat{\boldsymbol{\beta}}$s are lower than those of $\tilde{\boldsymbol{\beta}}_{\text{IIS}-\text{SCAD}}$s and $\tilde{\boldsymbol{\beta}}_{\text{IIS}_{\text{V1}}-\text{SCAD}}$s. However, the difference is not substantial, and for both renewable estimation methods, only a few significant features are missed in 100 repetitions. Moreover, CPs of $\hat{\boldsymbol{\beta}}$s for the non-zero coefficients are notably lower than those of $\tilde{\boldsymbol{\beta}}_{\text{IIS}-\text{SCAD}}$s and $\tilde{\boldsymbol{\beta}}_{\text{IIS}_{\text{V1}}-\text{SCAD}}$s. The outcomes are consistent with the previous two simulated examples in this section, $\hat{\boldsymbol{\beta}}$s in that they have visibly more incorrect non-zero estimates than $\tilde{\boldsymbol{\beta}}_{\text{IIS}-\text{SCAD}}$s and $\tilde{\boldsymbol{\beta}}_{\text{IIS}_{\text{V1}}-\text{SCAD}}$s. The two renewable estimation methods have similar performance in variable selections.

Excess risks of the classifiers and the computing time are shown in Table 4.56. The table demonstrates that, with different size training datasets, both renewable classifiers behave almost the same as the Naive Bayes classifiers, as indicated by the excess risks being close to 1 in various situations. Both renewable classifiers make fewer mislabels in this example than the classifier of the offline method. In addition, the renewable estimation methods once again display obvious advantages over the offline method with significantly reduced computing time and resources.

Table 4.54: AMRSEs and MMRSEs (in brackets) of estimates

N_b	1000	1500	2000	2500
$\tilde{\boldsymbol{\beta}}_{\text{IIS}-\text{SCAD}}$	0.019(0.005)	0.014(0.003)	0.012(0.002)	0.011(0.001)
$\tilde{\boldsymbol{\beta}}_{\text{IIS}_{\text{V1}}-\text{SCAD}}$	0.020(0.006)	0.013(0.003)	0.011(0.002)	0.011(0.001)
$\hat{\boldsymbol{\beta}}^{*}_{\text{IIS}-\text{SCAD}}$	0.053(0.008)	0.071(0.003)	0.065(0.056)	0.046(0.001)

* All results are the numbers in the table $\times 10^{-3}$.

Table 4.55: Average numbers of incorrect zero estimates for non-zero coefficients and incorrect non-zero estimates for zero coefficients, along with the coverage probabilities (in parentheses) for $\boldsymbol{\beta}$

N_b		1000	1500	2000	2500
$\tilde{\boldsymbol{\beta}}_{\text{IIS}-\text{SCAD}}$	non0 coeff.	0.04(0.887500)	0.04(0.892500)	0.04(0.917500)	0.04(0.890000)
	0 coeff.	0.89(0.999930)	0.65(0.999963)	0.58(0.999983)	0.53(0.999993)
$\tilde{\boldsymbol{\beta}}_{\text{IIS}_{\text{V1}}-\text{SCAD}}$	non0 coeff.	0.07(0.887500)	0.07(0.902500)	0.07(0.922500)	0.07(0.905000)
	0 coeff.	0.88(0.999883)	0.59(0.999970)	0.55(0.999983)	0.52(0.999997)
$\hat{\boldsymbol{\beta}}^{*}_{\text{IIS}-\text{SCAD}}$	non0 coeff.	0.01(0.832500)	0(0.752500)	0(0.790000)	0(0.820000)
	0 coeff.	12.50(0.999663)	11.50(0.999893)	22.76(0.995441)	11.33(0.999997)

* $\boldsymbol{\beta}_{3000\times 1}$ has 4 non-zero coefficients.

Table 4.56: Excess risks of $C_{\tilde{\beta}_{IIS-SCAD}}$s, $C_{\tilde{\beta}_{IIS_{V1}-SCAD}}$s and $C_{\hat{\beta}^*_{IIS-SCAD}}$s, and computing time and memory utilisations (in brackets)

N_b	1000	1500	2000	2500
$ER(C_{\tilde{\beta}_{IIS-SCAD}})$	0.979060 (32:16:35,4.25 GB)	0.983978 (32:43:56,4.24 GB)	0.988947 (33:03:17,5.06 GB)	0.992128 (33:13:48,5.11 GB)
$ER(C_{\tilde{\beta}_{IIS_{V1}-SCAD}})$	0.976158 (31:37:43,3.59 GB)	0.981646 (31:46:50,3.62 GB)	0.987330 (31:35:39,3.66 GB)	0.991111 (31:42:22,3.72 GB)
$ER(C_{\hat{\beta}^*_{IIS-SCAD}})$	0.957648 (36:05:31,7.16 GB)	0.959423 (40:20:13,11.69 GB)	0.987330 (31:35:39,22.36 GB)	0.923126 (52:19:39,22.19 GB)

* We have the same 500 out-of-sample testing data for each case. The misclassification rate of Bayes classifier is 28.988%.

* We implemented each experiment with 100 independent trials using C++ and ran them on 2 *Intel Xeon 6138 CPUs*. 40 cores were requested for each experiment.

* For each cell in the table, the value in the first row outside the brackets represents the excess risk of the classifier. The first and second numbers inside the brackets denote the computing time of each experiment consisting of 100 trials and the memory utilisation, respectively.

In summary, we conduct three simulated examples with datasets consisting of high-dimensional data with more covariates than the total sample sizes ($p > N_b$). The simulation study compares the renewable estimation methods using IIS-SCAD and IIS_{V1}-SCAD with the offline method using IIS-SCAD. We have found that our renewable estimation methods using IIS-SCAD and its variant can generate accurate estimates, identify the important features, and also have efficient classifiers with performance comparable to Bayes classifiers. The renewable estimates are stable with changes in the correlation of the covariates, the sizes of the streaming data batches, and the sizes of the total training data. Furthermore, the advantages of the two renewable estimation methods over the offline method are that they generate more interpretable models and require less time in computing. We demonstrate that our proposed renewable methods can work well for high-dimensional datasets and are less computationally expensive than the traditional offline method.

4.7 Real data analysis

In this section, we explore the practical application of our proposed penalised renewable estimation methods. We conduct the analysis on data obtained from the National Automotive Sampling System Crashworthiness Data System. Monthly accident data have been compiled to generate streaming data for a seven-year period, from January

2009 to December 2015. We treat the streaming data collected during the same month as a "data batch" and denote it as $\mathbf{B}_b$, $b = 1, \cdots, 84$, with a total of 84 data batches as training datasets available.

The "Fatality" resulting from a crash serves as the dependent variable and is denoted as $y_i = \{0, 1\}$, $i = 1, \cdots, N_b$, where N_b represents the size of the entire training dataset for these seven years. Following Luo and Song (2020)'s work, we encode the three age groups in our analysis using dummy variables as well, with the middle-aged group serving as the reference. Table 4.57 provides detailed explanations for the variables studied in this section of the datasets. We have the observations denoted as $\mathbf{X}_i$, $i = 1, \cdots, N_b$, and we represent the predictor variables as $\mathbf{X}_i = (x_{1i}, \cdots, x_{pi})^T$, which is a p-vector, while p denotes the number of predictor variables in the dataset. There is an intercept term in our model, so $x_{1i} \equiv 1$, $i = 1, \cdots, N_b$.

We assume that

$$\pi_y(X_i) = P(y_i = 1|X_i), \quad \text{logit}\{\pi_y(X_i)\} = X_i^T \boldsymbol{\beta},$$

where $\boldsymbol{\beta}$ is a p-dimensional vector that represents the coefficients associated with the predictor variables.

We examine two datasets with various predictor factors (predictor variables) using logistic regression analysis to investigate the relationship between fatality in an accident and these predictor variables. Assuming sparsity, there are unimportant factors involved.

We aim to find the MLEs of the unknown parameter $\boldsymbol{\beta}$ using our proposed iterative estimation methods introduced in Section 4.2.2 and Section 4.4.1. In addition, we apply the offline estimation method described in Section 4.2.1 to study the entire training dataset with a sample size of N_b. The offline method assumes access to all data at once and serves as a benchmark for comparing the performance of the iterative estimation methods. We integrate the SCAD penalty function into the models that have a penalty term, which can accomplish estimation and variable selection simultaneously. We assign the value of parameter a as 3.7 in the penalisation function (2.4). For the process of selecting the tuning parameter λ of the SCAD penalty function, we employ mLPOCV process described in Section 4.3.1, with a fixed cross-validation sample size of $n_{cv} = 20$. In this section, we use preset search intervals and steps with

appropriate sizes for the analysis. To avoid the zero absorbing state as far as possible, we set the threshold σ_0 in Equation (4.16) to 10^{-10} when analysing the monthly data for the years 2009-2011, and to 10^{-2} for the later years. To implement the penalised offline method, we define a threshold of $\sigma_0 = 10^{-2}$ for the estimate of β_j, $j = 1, \cdots, p$, to reach 0 during the estimation process iterations.

We perform the Wald test for the final renewable estimate $\tilde{\boldsymbol{\beta}}_{84}$ and also the offline MLE $\hat{\boldsymbol{\beta}}^*$. Our null hypothesis is $H_0 : \beta_j = 0$, against $H_1 : \beta_j \neq 0$, $j = 1, \cdots, 9$, with a significance level of $\alpha = 0.05$. The p values for each estimate are recorded in the following tables.

Referring to (3.6), we can test the efficiency of the classifiers. Specifically, we use the next month's data batch $\mathcal{B}_{b+1}$ as the out-of-sample testing data for the estimates $\tilde{\boldsymbol{\beta}}_b$, $b = 1, \cdots, 83$. Consequently, there are 83 testing data batches. The reason for choosing the following monthly data batch is that data collected in different batches may not be identically distributed, and significant features can change over time. Therefore, we believe that predictions on more recent data can more effectively measure the performance of renewable estimates. We also use the same 83 data batches $\mathcal{B}_{b+1}$, $b = 1, \cdots, 83$, to evaluate the efficiency of the offline classifier. It should be noted that this constitutes an in-sample test for the offline classifier. The misclassification rates of the offline classifiers serve as a benchmark for comparing the performance of the renewable classifiers in this section.

Table 4.57: Variable Descriptions

Variables	Data Type	Description
Dependent ($\mathbf{y}$)		
fatality	Nominal	1 for there is death case and 0 for otherwise (in the data "FATAL" AND "FATAL-RULED DISEASE").
Independent ($\mathbf{X}$)		
AGE	Categorical	Driver's age categorized as YOUNG(< 21), OLD($\geqslant 65$), and Reference group($21 \leqslant$ age < 65).
SEX	Nominal	Gender of the driver (1 for Male, 0 for Female).
PARUSE	Nominal	Police-reported seat belt use (1 for Used, 0 for Not Used).
LGTCOND	Nominal	Daylight condition at the time of the crash (0 for Daylight, 1 for Not Daylight).
DRINKING	Nominal	Police-reported alcohol involvement (1 for Yes, 0 for No).
FOURWHDR	Nominal	Four-wheel drive status of the vehicle (1 for Four-wheel Drive, 0 for Not Four-wheel Drive).
TRCTLFCT	Nominal	Traffic control device functioning properly at the time of the crash (1 for Functioning Properly, 0 for Not Functioning Properly).
DRGINV	Nominal	Drug involvement reported for any involved driver (1 for Yes, 0 for No).
ALIGNMNT	Nominal	If a roadway which has a curvature of a roadway, left or right, it is 1 or there is no perceptually determined curvature is 0.
CLIMATE	Nominal	The clear atmospheric condition is 0 and other conditions are denoted as 1.
SURCOND	Nominal	The precrash environment data presented in the location. The variable is 0 if it is dry or the variable is 1 with other cases.
TRAFFLOW	Nominal	The variable is 0 if it is one-way traffic or no divided in the road, and it is 1 for other cases.
VEHNO	Continuous	The number of cars get involved in the crash.
OCUPANTS	Continuous	The number of occupants structured into the case for this vehicle.
LANES	Continuous	Roadway-number of travel lanes.
SPLIMIT	Continuous	Identified speed limits for crash scene locations.
VEHAGE	Continuous	The age of the vehicle in the crash.
WEIGHT	Continuous	The driver's weights.
HEIGHT	Continuous	The driver's heights.

Case 1: For the first analysis in this section, we study the same dataset as Luo and Song (2020). Specifically, seven features—"Age", "Sex", "PARUSE", "LGTCOND", "DRINKING", "SPLIMIT", and "TRCTLFCT"—are selected for the dataset as predictor variables, and the observations are denoted as x_{ji}, where $i = 1, \cdots, N_b$, $j = 2, \cdots, p$, and $p = 9$. After removing missing values, the full-size training data we employ consists of $N_b = 23184$ observations which are kept in $b = 84$ data batches. We compare Luo and Song (2020)'s renewable estimation approach not considering variable selections and denote the estimates as $\tilde{\boldsymbol{\beta}}_{b,\lambda=0}$, with our proposed penalised incremental algorithms in Section 4.2.2 and Section 4.4.1, denoted as $\tilde{\boldsymbol{\beta}}_{b,\lambda}$ and $\tilde{\boldsymbol{\beta}}_{b,\lambda^*}$

respectively. In addition, we apply the unpenalised offline method and the penalised offline method for keeping consistency with the online unpenalised method and the online penalised method, and the estimates of $\boldsymbol{\beta}$ are denoted as $\hat{\boldsymbol{\beta}}^*_{b,\lambda=0}$ and $\hat{\boldsymbol{\beta}}^*_b$ respectively.

Table 4.58 presents the estimated coefficients and p-values from the aforementioned renewable estimation methods and also the offline estimation methods. All unpenalised and penalised renewable estimation methods perform very close to the corresponding offline methods, which is shown by the small difference between $\hat{\boldsymbol{\beta}}^*_{84,\lambda=0}$ and $\tilde{\boldsymbol{\beta}}_{84,\lambda=0}$, and between $\hat{\boldsymbol{\beta}}^*_{84}$ and $\tilde{\boldsymbol{\beta}}_{84,\lambda}$ and between $\hat{\boldsymbol{\beta}}^*_{84}$ and $\tilde{\boldsymbol{\beta}}_{84,\lambda^*}$. For the results of unpenalised offline and online methods, we have the same conclusion as Luo and Song (2020). In detail, from the recorded p-values of the Wald test, the variable YOUNG is not significantly associated with the dependent variable at a significance level of $\alpha = 0.05$, while the other predictors exhibit significant associations. Moreover, both the small numerical results of $\tilde{\boldsymbol{\beta}}_{84,\lambda=0}$ and $\hat{\boldsymbol{\beta}}^*_{84,\lambda=0}$ reveal that the predictor variables for SEX, LGTCOND, and TRCTLFCT have a weaker impact, which also corroborates the claim made by Luo and Song (2020) that these variables are the weakest predictors. In comparison, the penalised offline method and both our proposed approaches, which can perform estimation and variable selection simultaneously, explicitly identify that the predictor variable YOUNG associated with the driver's age which is under 21 is insignificant. Besides, the offline penalised method and the renewable estimation with constant penalty term during the iterations show that the predictor variables of gender, the variable SEX, has minimal influence on accident fatalities, while $\tilde{\boldsymbol{\beta}}_{84,\lambda}$ gives non-zero value to the predictor variable related to gender. For the weak predictor variable LGTCOND, both the penalised offline method and the renewable method with iterative penalty term view it as an insignificant feature however, the value of $\tilde{\boldsymbol{\beta}}_{84,\lambda}$ shows the opposite conclusion. We note that the penalised offline method evaluates the variable TRCTLFCT as an insignificant feature while both two renewable estimation methods have different conclusions, although the values of the estimates are small, the p values are much less than $\alpha = 0.05$ and the null hypothesis is rejected under the given significance level.

The 95% pointwise confidence intervals for $\beta_j,\ j = 1,\ \cdots,\ 9,$ are illustrated in Figure 4.1-Figure 4.3. The two offline estimates $(\hat{\boldsymbol{\beta}}^*_{\lambda=0})_j$ and $\hat{\beta}^*_j$, $j = 1, \cdots, 9$, are incorporated in the plots corresponding to the unpenalised renewable estimates and

the penalised renewable estimates separately. The plots demonstrate that the performance of the incremental algorithms is comparable to that of the offline methods in terms of efficacy. Figure 4.1 provides a clear visualization that, in the SCAD-based methods, the predictor variable YOUNG has no significant effect on the dependent variable. In particular, in the estimation process with either a constant penalty term or the iterative penalty term through iterations, the renewable estimates associated with this predictor gradually approach 0 and become exactly zero around the monthly data collected in December 2011. The estimate of the predictor from the penalised offline method is also 0. However, the methods that do not consider variable selection still provide estimates for the predictor variable YOUNG, but the confidence intervals obtained from both methods show that the interval includes the value 0.

Figure 4.2 and Figure 4.3 demonstrate that the offline penalised method clearly shows that the predictors associated with gender, light condition, and traffic control function have no impact on accident fatalities. Likewise, the trace plots of the renewable method with constant penalty term during iteration show the exact same trace as the offline penalised estimation method. However, the trace plot of the renewable estimation with the iterative penalty term indicates a different trend with the increasing trend values of the variable SEX. For the predictor variable LGTCOND, the estimation process with iterative penalty term is consistent with the penalised offline method, as shown by the estimated value being 0 at the end of the year 2010; In contrast, the estimation process with constant penalty term differs from the offline estimation process, displaying an increasing trend of the varaible LGTCOND. Although the two penalised renewable methods indicate different conclusions compared to the penalised offline method, which reject the null hypothesis and give non-zero values to the estimates for the predictor variables SEX, LGTCOND, and TRCTLFCT, the given values are small. It is worth noting that the estimates for the three predictors that are assigned non-zero values by both the penalised renewable methods are similar to those of the unpenalised offline and online methods.

In this analysis, we also observe that the misclassification rates of all classifiers are identical. Figure 4.4 depicts the misclassification rates of the classifiers, which exhibit precisely the same trend when tested on 83 monthly data batches. Table C.13 in Appendix C.1 records the details of the size of each testing data batch and the misclassification rates of classifiers of unpenalised methods, which are $C_{\hat{\boldsymbol{\beta}}^*_{\lambda=0}}$ and

$C_{\tilde{\beta}_{b,\lambda=0}}$, and the classifiers of the penalised methods, which are $C_{\hat{\beta}^*_b}$, $C_{\tilde{\beta}_{b,\lambda}}$, and $C_{\tilde{\beta}_{b,\lambda^*}}$, where $b = 1, \cdots, 83$. For all the tested classifiers, the averaged misclassification rate for the 83 testing data batches is identical, which is 3.3707%.

In summary, both unpenalised and penalised online algorithms are adept at producing estimates that closely resemble the values obtained from their respective unpenalised and penalised offline counterparts. Moreover, the penalised renewable estimation methods generate more comprehensible models, which can aid in devising policies aimed at effectively mitigating fatal crash outcomes. For example, this could involve confidently implementing checks for drivers over the age of 65, while avoiding the allocation of extra effort to check young drivers under 21, thus conserving resources.

Table 4.58: Comparisons of $\hat{\beta}^*_{84,\lambda=0}$ and $\tilde{\beta}_{84,\lambda=0}$ with $\hat{\beta}^*$, $\tilde{\beta}_{84,\lambda}$ and $\tilde{\beta}_{84,\lambda^*}$

Unpenalised methods		Intercept	YOUNG	OLD	SEX	PARUSE	LGTCOND	DRINKING	SPLIMIT	TRCTLFCT
$\hat{\beta}^*_{84,\lambda=0}$	Estimates	**-3.078550**	-0.052275	**0.888406**	**0.318312**	**-1.084010**	**0.393369**	**0.861523**	**0.381774**	**-0.285487**
	p values	$<$ **10^{-6}**	0.679512	$<$ **10^{-6}**	**0.000061**	$<$ **10^{-6}**	**0.000002**	$<$ **10^{-6}**	$<$ **10^{-6}**	**0.001196**
$\tilde{\beta}_{84,\lambda=0}$	Estimates	**-3.037070**	-0.0536644	**0.887480**	**0.313039**	**-1.101370**	**0.392372**	**0.856984**	**0.381366**	**-0.295664**
	p values	$<$ **10^{-6}**	0.687054	$<$ **10^{-6}**	**0.000056**	$<$ **10^{-6}**	**0.000002**	$<$ **10^{-6}**	$<$ **10^{-6}**	**0.000662**
$\lvert(\hat{\beta}^*_j)_{84,\lambda=0} - (\tilde{\beta}_j)_{84,\lambda=0}\rvert$		0.041478	0.001389	0.000926	0.005273	0.017362	0.000997	0.004539	0.000408	0.010177
penalised methods		Intercept	YOUNG	OLD	SEX	PARUSE	LGTCOND	DRINKING	SPLIMIT	TRCTLFCT
$\hat{\beta}^*_{84}$	Estimates	**-2.822521**	0	**0.819037**	0	**-1.141305**	0	**1.106345**	**0.444160**	0
	p values	$<$ **10^{-6}**	1.000000	$<$ **10^{-6}**	1.000000	$<$ **10^{-6}**	1.000000	$<$ **10^{-6}**	$<$ **10^{-6}**	1.000000
$\tilde{\beta}_{84,\lambda}$	Estimates	**-2.854330**	0	**0.900608**	0	**-1.124860**	**0.433900**	**0.854183**	**0.374177**	**-0.283943**
	p values	$<$ **10^{-6}**	1.000000	$<$ **10^{-6}**	1.000000	$<$ **10^{-6}**	$<$ **10^{-6}**	$<$ **10^{-6}**	$<$ **10^{-6}**	**0.000900**
$\lvert\hat{\beta}^*_{84,j} - (\tilde{\beta}_j)_{84,\lambda}\rvert$		0.03181	0	0.081571	0	0.01645	0.4339	0.252157	0.069983	0.283943
$\tilde{\beta}_{84,\lambda^*}$	Estimates	**-2.904430**	0	**0.777164**	**0.341383**	**-1.048210**	0	**1.021490**	**0.358074**	**-0.311303**
	p values	$<$ **10^{-6}**	1.000000	$<$ **10^{-6}**	**0.000007**	$<$ **10^{-6}**	1.000000	$<$ **10^{-6}**	$<$ **10^{-6}**	**0.000188**
$\lvert\hat{\beta}^*_{84,j} - (\tilde{\beta}_j)_{84,\lambda^*}\rvert$		0.081910	0	0.041873	0.341383	0.0931	0	0.084850	0.086086	0.311303

* $\hat{\beta}^*_{84,\lambda=0}$ denotes the results obtained from the offline method without variable selection, while $\tilde{\beta}_{84,\lambda=0}$ represents the results obtained from the renewable estimation method without variable selection. On the other hand, $\hat{\beta}^*_{84,\lambda}$ denotes the results obtained from the offline method incorporating SCAD, while $\tilde{\beta}_{84,\lambda}$ and $\tilde{\beta}_{84,\lambda^*}$ represent the results obtained from the penalised renewable estimation methods introduced in Section 4.2.2 and Section 4.4.1, respectively.

* The mean absolute difference between $\hat{\beta}^*_{84,\lambda=0}$ and $\tilde{\beta}_{84,\lambda=0}$ is 0.009172, the mean absolute difference between $\hat{\beta}^*_{84}$ and $\tilde{\beta}_{84,\lambda}$ is 0.129979, and the mean absolute difference between $\hat{\beta}^*_{84}$ and $\tilde{\beta}_{84,\lambda^*}$ is 0.115612.

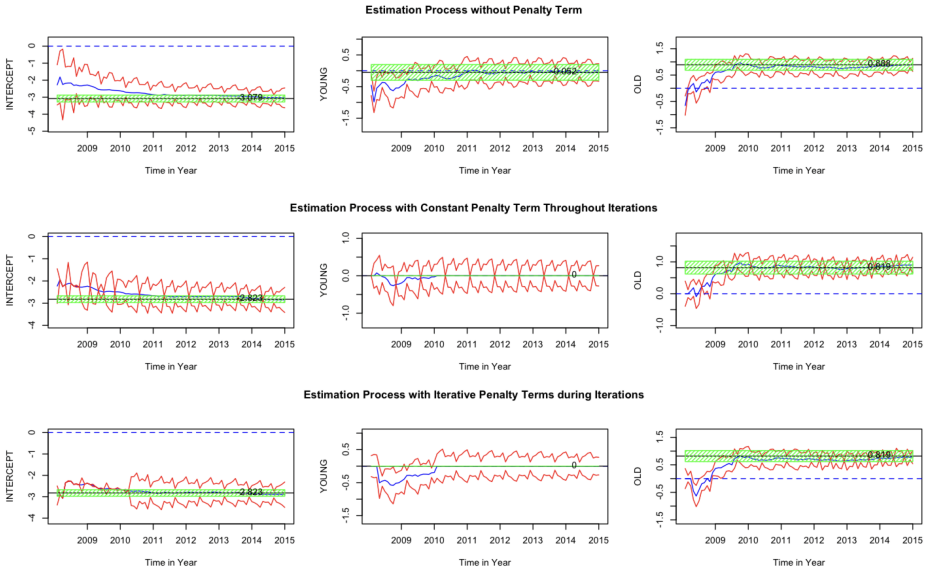

Figure 4.1: Trace plots for the coefficient estimates of "INTERCEPT", "YOUNG", and "OLD". The blue lines represent the 84 monthly renewable estimates, and the areas between the two red lines denote their 95% pointwise confidence zones. The black lines correspond to the estimates from the offline methods, with their values marked on the plots, while the green shaded areas indicate the 95% confidence intervals. The three plots in the first row display estimates for the online and offline methods without the penalty function, and the three plots in the second row depict estimates for the online and offline methods using SCAD. The blue dotted lines serve as reference lines for 0 values.

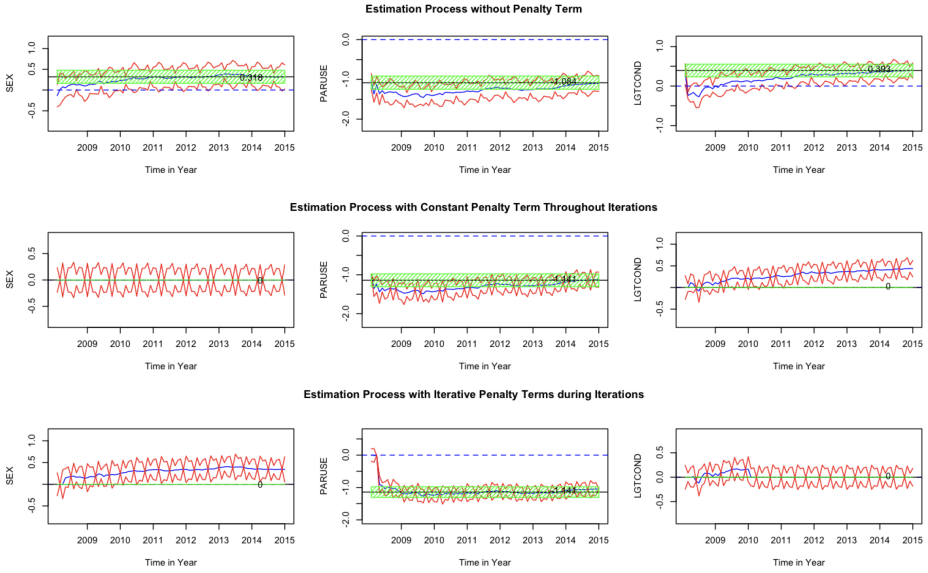

Figure 4.2: Trace plots for the coefficient estimates of "SEX", "PARUSE" and "LGTCOND". The blue lines represent the 84 monthly renewable estimates, and the areas between the two red lines denote their 95% pointwise confidence zones. The black lines correspond to the estimates from the offline methods, with their values marked on the plots, while the green shaded areas indicate the 95% confidence intervals. The three plots in the first-row display estimates for the online and offline methods without the penalty function, and the three plots in the second row depict estimates for the online and offline methods using SCAD. The blue dotted lines serve as reference lines for 0 values.

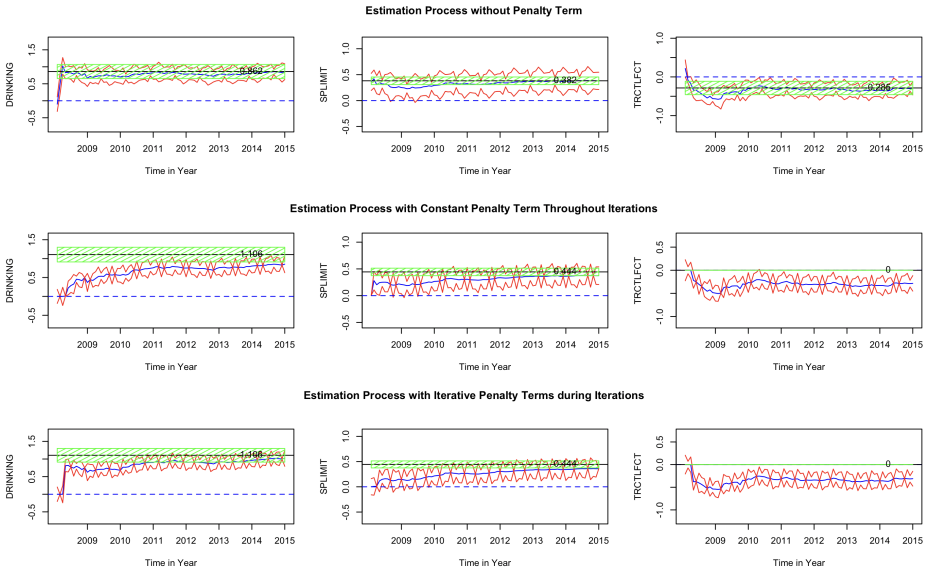

Figure 4.3: Trace plots for the coefficient estimates of "DRINKING", "SPLIMIT" and "TRCTLFCT". The blue lines represent the 84 monthly renewable estimates, and the areas between the two red lines denote their 95% pointwise confidence zones. The black lines correspond to the estimates from the offline methods, with their values marked on the plots, while the green shaded areas indicate the 95% confidence intervals. The three plots in the first-row display estimates for the online and offline methods without the penalty function, and the three plots in the second row depict estimates for the online and offline methods using SCAD. The blue dotted lines serve as reference lines for 0 values.

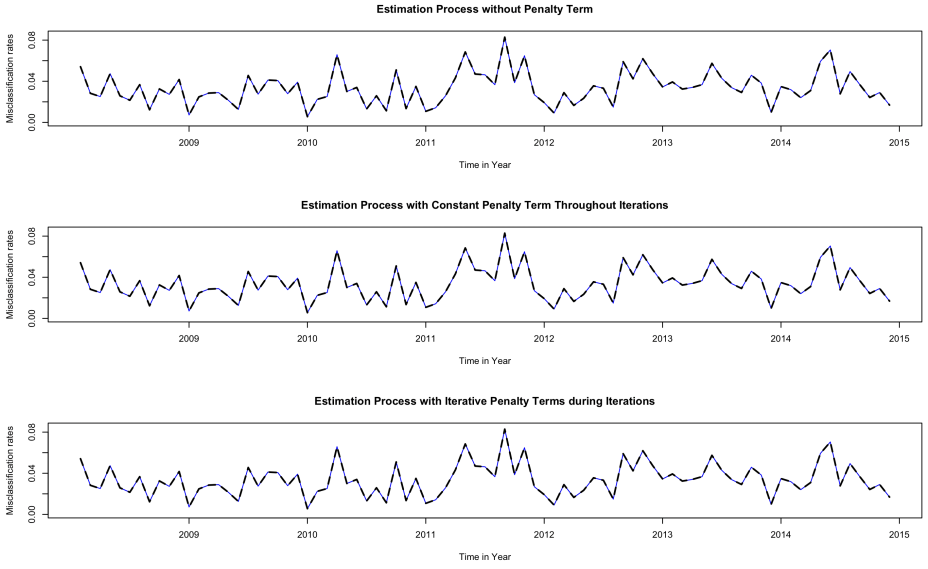

Figure 4.4: Trace plots for the misclassification rates of renewable classifiers and offline classifiers. The testing data batches are $\mathcal{B}_{b+1}$, where $b = 1, \cdots, 83$. The blue lines represent the misclassification rates of renewable classifiers, while the black dotted lines correspond to the classifiers from offline methods. The top plot displays the renewable classifier $C_{\tilde{\boldsymbol{\beta}}_{b,\lambda=0}}$ and $C_{\hat{\boldsymbol{\beta}}^*_{\lambda=0}}$; the middle plot showcases the renewable classifier $C_{\tilde{\boldsymbol{\beta}}_{b,\lambda}}$ and $C_{\hat{\boldsymbol{\beta}}^*}$; the bottom plot features the renewable classifier $C_{\tilde{\boldsymbol{\beta}}_{b,\lambda*}}$ and $C_{\hat{\boldsymbol{\beta}}^*}$, where $b = 1, \cdots, 83$.

Case 2: In the second real data analysis in this section, we examine more predictor variables in the dataset compared to **Case 1**. It is more practical to include additional

predictor variables when conducting an analysis, as researchers typically collect data on numerous factors. By considering a broader range of factors, we avoid overlooking any potentially important ones. However, many factors might be insignificant, and it can be difficult to manually determine their correlation with the response variable. To address this issue, we employ the SCAD penalty function for variable selection to identify the essential features.

In detail, the dataset comprises incidents that occurred between January 2009 and December 2015, with 84 monthly data batches, each considered as a single streaming data batch $\mathcal{B}_b$, $b = 1, \cdots, 84$. We include the categorical variable "AGE", and similar to **Case 1**, we use dummy variables and separate drivers with different ages into three groups. In addition to the predictor variables considered in **Case 1** ("Age", "Sex", "PARUSE", "LGTCOND", "DRINKING", "SPLIMIT", and "TRCTLFCT"), we incorporate 12 extra factors into the dataset and $p = 21$, denoted as "OURWHDR", "DRGINV", "ALIGNMNT", "CLIMATE", "SURCOND", "TRAFFLOW", "VEHNO", "OCUPANTS", "LANES", "WEIGHT", and "HEIGHT". After removing missing values, the full-size training dataset consists of $N_b = 11884$ observations.

For this dataset, Luo and Song (2020)'s method without variable selection fails to converge; therefore, we only present the numerical results of the methods incorporating the SCAD penalty function. These methods include the offline penalised estimation method and the renewable penalised estimation methods introduced in Section 4.2.2 and 4.4.1. For simplicity, we omit the term "penalised" in the following context when referring to both the offline and online estimation methods using SCAD.

Table 4.59 displays the estimated values of the coefficients for the three methods, the p-values from the hypothesis tests, and the absolute values of the differences between the estimates from the offline method and the renewable estimation methods. Similar to **Case 1**, both renewable estimation methods exhibit performance closely aligned with the offline estimation method, as evidenced by the small values of the absolute differences between the estimates of $(\hat{\beta}_j)_b^*$ and $(\tilde{\beta}_j)_{b,\lambda}$, and $(\tilde{\beta}_j)_{b,\lambda^*}$, for $j = 1, \cdots, 21$. Furthermore, all results indicate that factors coded as OLD, PARUSE, DRINKING, and SPLIMIT are significant features, as demonstrated by the non-zero values of the estimates and p-values less than 0.05, consistent with **Case 1**. Additionally, among the added features, the offline method selects factors related

to drug use (DRGINV), roadway curvature (ALIGNMNT), and identified speed limits at crash scene locations (SPLIMIT) as significant features as well. For other features, the offline method considers them insignificant and assigns 0 values to them. It is worth mentioning that the renewable estimation method, which incorporates iterative penalty terms during iterations, considers factors SEX, LGTCOND, and TRCTLFCT as insignificant features. These factors are assigned 0 values and yield different conclusions compared to **Case 1**. In contrast, for $\tilde{\boldsymbol{\beta}}_{b,\lambda}$, a non-zero value is assigned to SEX, but the hypothesis test does not reject the null hypothesis, and LGTCOND is still regarded as a significant feature.

It is apparent that the renewable estimation method with the iterative penalty term demonstrates a performance closer to the offline method in variable selection. For the significant features, the renewable estimation method with the iterative penalty term identifies almost the same important features as the offline estimation method. However, the renewable estimation method employing a constant penalty term during iteration selects a larger number of features. For the insignificant features, $\tilde{\boldsymbol{\beta}}_{84,\lambda^*}$ also displays a performance more aligned with $\hat{\boldsymbol{\beta}}^*_{84}$, meaning that for features considered as insignificant by the methods, the assigned values are 0s, whereas $\tilde{\boldsymbol{\beta}}_{84,\lambda}$ allocates non-zero values to the features, even when the p-values are considerably higher than $\alpha = 0.05$, suggesting no rejection of the null hypothesis.

Figure 4.5 illustrates the misclassification rates of $C_{\hat{\boldsymbol{\beta}}^*_b}$, $C_{\tilde{\boldsymbol{\beta}}_{b,\lambda}}$, and $C_{\tilde{\boldsymbol{\beta}}_{b,\lambda^*}}$, where $b = 1, \cdots, 83$, and the testing data is in batch $\mathcal{B}_{b+1}$. Table C.14 in Appendix C.1 documents the details of the size of each testing data batch and the misclassification rates of each classifier. The overall performance and trend of the three classifiers are similar. Specifically, the average misclassification rate for the 83 classifiers of $C_{\hat{\boldsymbol{\beta}}^*_b}$ is 3.9660%, for $C_{\tilde{\boldsymbol{\beta}}_{b,\lambda}}$s is 3.8926%, and for $C_{\tilde{\boldsymbol{\beta}}_{b,\lambda^*}}$s is 3.9365%, where $b = 1, \cdots, 83$. The overall performance of the classifiers from the renewable estimation method with a constant penalty term is marginally better than both the offline method and the renewable estimation method using an iterative penalty term during iterations.

In summary, when more features are involved in the collected dataset, the renewable estimation method that does not consider variable selection fails to perform effectively. As such, the method not considering variable selection is not recommended due to its limited analytical capabilities. Both proposed penalised renewable estimation methods incorporating SCAD work well in achieving low misclassification

rates for out-of-sample validations and demonstrate a performance close to the offline method. However, the offline method has well-known disadvantages, including the ideal assumption of having access to the entire training dataset and the extensive resources required for storage and computation. Our proposed incremental algorithms offer significant advantages over the offline method in terms of computational efficiency and resource savings. The renewable estimation with the iterative penalty term can generate more interpretable models than the renewable estimation method using a constant penalty term, which can be explained by their algorithms introduced in Section 4.2.2 and Section 4.4.1. More specifically, the renewable estimation with the iterative penalty term includes an additional penalty term related to the estimator of the iteration, while the method using the constant penalty term is based on the historical estimate.

Table 4.59: Comparisons of $\hat{\boldsymbol{\beta}}^*_{84,\lambda}$, $\tilde{\boldsymbol{\beta}}_{84,\lambda}$ and $\tilde{\boldsymbol{\beta}}_{84,\lambda^*}$

	$\hat{\boldsymbol{\beta}}^*_{84,\lambda}$		$\tilde{\boldsymbol{\beta}}_{84,\lambda}$			$\tilde{\boldsymbol{\beta}}_{84,\lambda^*}$		
	Estimates	p values	Estimates	p values	$\lvert\hat{\beta}^*_j - (\tilde{\beta}_j)_{84,\lambda}\rvert$	Estimates	p values	$\lvert\hat{\beta}^*_j - (\tilde{\beta}_j)_{84,\lambda^*}\rvert$
Intercept	**-2.952385**	$<$ **10^{-6}**	**-3.126050**	$<$ **10^{-6}**	0.173660	**-2.952450**	$<$ **10^{-6}**	0.000060
YOUNG	0	1.000000	0	1.000000	0	0	1.000000	0
OLD	**1.095510**	$<$ **10^{-6}**	**1.146640**	$<$ **10^{-6}**	0.051130	**1.127080**	$<$ **10^{-6}**	0.031570
SEX	0	1.000000	0.0871726	0.452506	0.087173	0	1.000000	0
PARUSE	**-1.416917**	$<$ **10^{-6}**	**-1.333950**	$<$ **10^{-6}**	0.082970	**-1.388820**	$<$ **10^{-6}**	0.028100
LGTCOND	0	1.000000	**0.386738**	**0.000935**	0.386738	0	1.000000	0
DRINKING	**0.490140**	**0.001469**	**0.356169**	**0.038865**	0.133971	**0.491575**	**0.002413**	0.001435
FOURWHDR	0	1.000000	-0.127282	0.323737	0.127282	0	1.000000	0
TRCTLFCT	0	1.000000	0	1.000000	0	0	1.000000	0
DRGINV	**1.634611**	$<$ **10^{-6}**	**1.602850**	$<$ **10^{-6}**	0.031760	**1.638770**	$<$ **10^{-6}**	0.004160
ALIGNMNT	**0.423420**	**0.000171**	**0.332916**	**0.005479**	0.090504	**0.394359**	**0.000688**	0.029061
CLIMATE	0	1.000000	0	1.000000	0	0	1.000000	0
SURCOND	0	1.000000	-0.113413	0.404823	0.113413	0	1.000000	0
TRAFFLOW	0	1.000000	0	1.000000	0	0	1.000000	0
VEHNO	0	1.000000	0.079532	0.154801	0.079532	0	1.000000	0
OCUPANTS	0	1.000000	0.081675	0.109400	0.081675	0.120591	0.011061	0.120591
LANES	0	1.000000	**0.402179**	**0.021799**	0.128336	0	1.000000	0
SPLIMIT	**0.433783**	$<$ **10^{-6}**	**0.383206**	$<$ **10^{-6}**	0.050577	**0.402179**	$<$ **10^{-6}**	0.031604
VEHAGE	0	1.000000	**0.230883**	**0.000004**	0.230883	**0.225579**	**0.000007**	0.225579
WEIGHT	0	1.000000	**0.191136**	**0.000344**	0.191136	0	1.000000	0
HEIGHT	0	1.000000	0	1.000000	0	0	1.000000	0
Computing time	02:58:31		00:07:04			00:11:57		
Memory Utilised	1.08 GB		13.34 MB			13.31 MB		

* $\hat{\boldsymbol{\beta}}^*_{84,\lambda}$ denotes the results obtained from the offline method incorporating SCAD, while $\tilde{\boldsymbol{\beta}}_{84,\lambda}$ and $\tilde{\boldsymbol{\beta}}_{84,\lambda^*}$ represent the results obtained from the penalised renewable estimation methods introduced in Section 4.2.2 and Section 4.4.1, respectively.

* The mean absolute difference between $\hat{\boldsymbol{\beta}}^*_{84}$ and $\tilde{\boldsymbol{\beta}}_{84,\lambda}$ is 0.0971773 and the mean absolute difference between $\hat{\boldsymbol{\beta}}^*$ and $\tilde{\boldsymbol{\beta}}_{84,\lambda^*}$ is 0.02248381.

* We implemented each experiment using C++ and ran them on 1 *Intel Xeon 6138 CPUs*. For each experiment of renewable estimations, we requested one core.

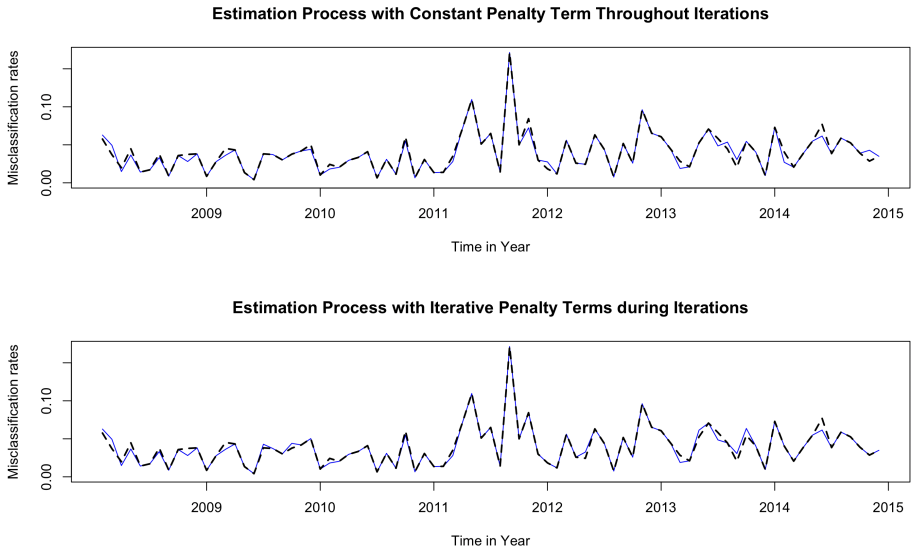

Figure 4.5: Trace plots for the misclassification rates of two renewable classifiers and the offline classifier. The testing data batches are $\mathcal{B}_{b+1}$, where $b = 1, \cdots, 83$. The blue lines represent the misclassification rates of renewable classifiers, while the black dotted lines correspond to the classifier from the offline method. The top plot displays the renewable classifier $C_{\tilde{\boldsymbol{\beta}}_{b,\lambda}}$ and $C_{\hat{\boldsymbol{\beta}}^*}$; the bottom plot shows the renewable classifier $C_{\tilde{\boldsymbol{\beta}}_{b,\lambda^*}}$ and $C(\hat{\boldsymbol{\beta}}^*)$, where $b = 1, \cdots, 83$.

5 Discussion

In this chapter, we encapsulate our main conclusions relating to classification with mislabelling and online algorithms for streaming data. Both topics are approached within a wide range of applications, particularly in the context of sparse datasets, including those that are high-dimensional. These scenarios present notable challenges and necessitate the use of specialised techniques and methodologies. We also discuss potential areas for further exploration in these two research domains.

5.1 Research objective 1: Classification with mislabelled data

We propose novel methods for addressing binary classification problems in the presence of mislabelling, which is a prevalent issue in real-world problems. In our study, the dataset under investigation contains incorrect labels, and the correct labels for each observation are unknown. Our study explores a versatile scenario where label noise exhibits dependencies on both the class and the features.

To tackle this challenge, we adopt the resampling approach, which provides practicality, cost-effectiveness, time efficiency, and ease of implementation compared to traditional data cleansing methods. Consequently, our argument relies on the condition that the resampled dataset is provided. In detail, our proposed method consists of a two-step estimation process that incorporates regularisation methods, as described in Section 3.1 of Chapter 3, which is designed to handle mislabelling in low-dimensional sparse datasets. Specifically, three unknown parameters, corresponding to the LR classifier and the flipping probabilities, necessitate estimation in our iterative algorithm. Additionally, we introduce methods using IS and IIS, which are presented in Section 3.9 and Section 3.10 respectively. By incorporating IS and IIS into the method described in Section 3.1, we broaden the applicability of the method to sparser data, specifically in the context of high-dimensional mislabelled datasets.

We execute a comprehensive simulation study encompassing both low-dimensional and high-dimensional datasets with mislabelling under various scenarios. Our method incorporates the SCAD regularisation technique, as proposed by Fan and Li (2001). Drawing inspiration from the insights of Cannings et al. (2020), who have pioneered a potent measure for contrasting classifiers against the Bayes benchmark, our comparisons consistently reflect a performance strikingly analogous to that of

the Bayes classifier. For our study's purposes, the Bayes classifier is constructed using a classifier with pre-established known values of the unknown parameters, mirroring the Bayes classifier's quintessential trait of minimising risk. However, it is noteworthy to highlight that while the Bayes classifier is universally lauded for its optimal classification capabilities, it is not devoid of challenges. It typically necessitates rigorous assumptions concerning class distributions, can be vulnerable to mislabelling in the training dataset, and at times, might pose computational challenges. In contrast, the LR classifier, derived from our proposed two-step estimation method, champions computational feasibility and demonstrates resilience against mislabelling in the training dataset.

In Section 3.7, we explore alternative methods that handle mislabelling in different ways. These include utilising the raw corrupted dataset directly, following the approach used by Cannings et al. (2020), correcting all labels in the dataset, and correcting only a subset of the data while disregarding label noise during estimation. We have performed simulation studies using diverse corrupted datasets to compare our method with the aforementioned alternative methods, as well as with a method that exclusively utilised the small clean data from the resampled dataset.

Notably, the results demonstrate the significant superiority of our proposed method over the two methods that do not consider the estimation of flipping probabilities. Importantly, it validates the fact that the LR classifier performs no better than a random guess when ignoring the imperfectly labelled data. Correcting only a subset of the dataset without considering flipping probabilities during estimation results in only minimal improvement in performance. Moreover, our method outperforms the approach that solely analyses the small clean dataset, demonstrating its effectiveness in leveraging imperfectly labelled data and enhancing the classifier's performance trained on the small resampled data. Furthermore, the comparable performance between our method and the approach that utilises all corrected data further confirms the effectiveness of our proposed approach. Considering real-world applications, our method offers additional advantages such as practicality, time efficiency, cost savings, and more.

Based on the discussions surrounding the aforementioned classification problem with mislabelling and low-dimensional data, we have expanded the application of our method by integrating IS and IIS to address more complex scenarios, including

high-dimensional datasets. IS and IIS are widely recognised as effective techniques for dimensionality reduction, significantly improving the performance of traditional regularisation methods such as SCAD, particularly in high-dimensional data settings (Fan and Lv, 2008, Fan and Song, 2010).

In our simulation study involving various high-dimensional datasets with mislabelling, our proposed method utilising IS-SCAD and IIS-SCAD and studying mislabelled dataset consistently perform competitively with the methods that incorporate IS or IIS while studying perfectly labelled datasets. Moreover, our classifier achieved performance levels comparable to the Bayes classifier. Based on these results, we can confidently conclude that IS and IIS remain effective even in high-dimensional datasets with mislabelling. However, a formal proof of this efficacy remains unexplored and is earmarked for future investigation.

Indeed, it is crucial to mention that previous studies, including the works by Fan and Lv (2008), Fan and Song (2010), Saldana and Feng (2018), have highlighted the limitations of IS when dealing with data that includes irrelevant features that are marginally associated with significant features. These limitations are also observed in our simulation study, corroborating the findings from prior research. Therefore, we strongly recommend adopting the two-step estimation method with IIS. By doing so, we can effectively address the challenges arising from high-dimensional data and mislabelling in various scenarios. The inclusion of IIS helps overcome the limitations associated with IS, ensuring robust performance even in the presence of highly marginally related insignificant features.

The real data analysis conducted on the FHS study in Section 3.11 demonstrates the competitive performance of our proposed method compared to the other methods mentioned, in terms of both model interpretation and classification accuracy.

After summarising the main findings of our study, we would like to discuss potential extensions for future research. Firstly, our research concentrated on two-class supervised classification issues for simplicity. However, our proposed method possesses the potential to address multi-class classification problems and even extend to unsupervised classification settings. Future studies might consider these as compelling areas for exploration. Secondly, a significant assumption in most of our simulated scenarios is that the observations are independent and identically distributed (i.i.d.). Nevertheless, real-world data often exhibit dependencies, evident

in time series data, image data for classification, and text classification. Therefore, an investigation into the performance of our method on various types of mislabelled datasets employing novel approaches could provide a fascinating direction for future research. Lastly, the extension of our proposed method to more advanced and adaptable models, such as semi-parametric models, is worth consideration. For instance, generalised additive models (GAMs) can effectively capture both the known relationships between variables and the potential complexity inherent within the data. Additionally, non-parametric models, such as Decision Trees, Support Vector Machines (SVM), K-Nearest Neighbors (KNN), and Neural Networks (NN), can be explored. These extended applications provide a wider perspective for understanding and interpreting complex data structures.

5.2 Research objective 2: Online algorithms for streaming data

The challenges posed by streaming data have captured our interest due to its pervasive presence in real-world problems and its growing significance in academic research. Indeed, traditional offline methods are frequently unsuitable for analysing streaming data due to their high computational demands and resource requirements. Additionally, offline methods are less preferable for real-time analysis.

Luo and Song (2020) offers a compelling approach to the analysis of streaming data. Compared to previous methods, their work attains computational efficiency and loses little information compared to offline methods, enabling the construction of online references such as confidence intervals. However, considering Luo and Song (2020)'s work does not extend to the more general case where streaming data is sparse and potentially high-dimensional, our study focuses on sparse streaming data.

We detail a method in Section 4.2.2 that incorporates a penalty function for the analysis of low-dimensional streaming data from sparse models. Given the characteristics of renewable estimation, we present another method in Section 4.4. Unlike the one in Section 4.2.2, which approximates the penalty term around the historical statistic, this method iteratively updates the penalty term during estimation. Simulation results indicate comparable performance between the two methods in terms of estimation accuracy. However, the latter excels in eliminating irrelevant features.

Extensive simulation studies have been conducted in our study. Notably, in

comparison to the offline method, our method exhibits competitive performance. Our approach closely resembles the offline method in estimating the accuracy of unknown parameters and calculating the standard errors of the parameters in various streaming data scenarios. As our study primarily focuses on logistic regression (LR), the efficiency of our classifier competes favourably with the traditional offline method and performs closely to the Bayes classifier. We also compare our proposed method using SCAD with the method proposed by Luo and Song (2020), which does not consider variable selection for low-dimensional sparse streaming data, in a simulation study detailed in Section 4.3.4. Our proposed method noticeably outperforms the latter in different criteria, including estimation accuracy, variable selection, and classification efficiency. It is worth noting that Luo and Song (2020)'s method fails to analyse sparser models, evidenced by inconclusive results.

To apply our method in broader scenarios, such as for sparser and high-dimensional data, we incorporate IS and IIS into our model, described in Section 4.5 and Section 4.6. Given the nature of streaming data, and considering the historical statistics, the extensions of IS and IIS in incremental algorithms are discussed. Due to IS's limitation in analysing data with insignificant features correlated with significant ones, we recommend using IIS, which is supported by our simulation results. Compared with the offline method using IIS-SCAD in the simulation study, our method achieves similar performance in estimation accuracy, variable selection, and classification efficiency. More importantly, it uses considerably fewer computing resources and less time. A rigorous proof affirming the sure screening property (2.5) in the context of our proposed method integrated with IS and IIS has yet to be delineated. We have designated this topic for in-depth investigation in future research.

Our proposed methods also yield promising results in the real data analysis using LR on the National Automotive Sampling System Crashworthiness Data System. When comparing our method to Luo and Song (2020)'s method using the same dataset, our approach demonstrates superior efficiency, effectiveness, and, most importantly, interpretable model. We have observed that the method proposed by Luo and Song (2020) faces challenges when analysing datasets that involved more predictors. The results once again demonstrate that our methods maintain comparable performance to the offline method, which serves as the benchmark. The misclassification rates closely resemble those of the offline method, and the

variable selection performance is on par, ensuring the interpretability of the resulting model. Importantly, our method is more practical, providing timely analysis without requiring extensive storage for data details, thereby saving resources and preserving privacy.

Our study has successfully charted several promising avenues for further investigation. Firstly, our focus primarily resides on logistic regression (LR) applied to sparse and high-dimensional streaming data. However, our model's application is not strictly limited to LR. It holds the potential for generalisation to include other generalised linear models (GLMs), thereby broadening its utility and relevance across a more diverse range of data structures. Secondly, in Section 4.4.2, we delve into the analysis of heterogeneous streaming data, which consists of independent but non-identically distributed observations. However, there are still numerous other intriguing data types that warrant further exploration. Examples include, but are not limited to, dependent streaming data, data streams with incorrect labels, and streams containing missing values. In future research, it would be valuable to extend the applicability of our method to a wider range of common applications and investigate its effectiveness in handling diverse data types. Engaging in such studies would undoubtedly make a substantial contribution to advancing our understanding and utilisation of streaming data analysis methods. These efforts would provide us with more nuanced insights into the robustness and adaptability of the model, while also identifying areas that may require further enhancement. Lastly, it is worth discussing the potential application of semi-parametric and non-parametric models in the context of streaming data analysis. By exploring these models, we could gain additional insights into the strengths and potential improvements of our method in handling complex and diverse data streams.

References

Abbott, R. D. (1985). Logistic regression in survival analysis. *American journal of epidemiology 121*(3), 465–471.

Abu Alfeilat, H. A., A. B. Hassanat, O. Lasassmeh, A. S. Tarawneh, M. B. Alhasanat, H. S. Eyal Salman, and V. S. Prasath (2019). Effects of distance measure choice on k-nearest neighbor classifier performance: a review. *Big data 7*(4), 221–248.

Allison, P. D. (2008). Convergence failures in logistic regression. In *SAS Global Forum*, Volume 360, pp. 11.

Amari, S.-i., H. Park, and K. Fukumizu (2000). Adaptive method of realizing natural gradient learning for multilayer perceptrons. *Neural computation 12*(6), 1399–1409.

Ambrish, G., B. Ganesh, A. Ganesh, C. Srinivas, K. Mensinkal, et al. (2022). Logistic regression technique for prediction of cardiovascular disease. *Global Transitions Proceedings 3*(1), 127–130.

Angluin, D. and P. Laird (1988). Learning from noisy examples. *Machine Learning 2*(4), 343–370.

Balakrishnama, S. and A. Ganapathiraju (1998). Linear discriminant analysis-a brief tutorial. *Institute for Signal and information Processing 18*(1998), 1–8.

Bhattacharyya, S. K. and K. Rahul (2013). Face recognition by linear discriminant analysis. *International Journal of Communication Network Security 2*(2), 31–35.

Bickel, P. J. and E. Levina (2004). Some theory for fisher's linear discriminant function,naive bayes', and some alternatives when there are many more variables than observations. *Bernoulli 10*(6), 989–1010.

Bifet, A., S. Maniu, J. Qian, G. Tian, C. He, and W. Fan (2015). Streamdm: Advanced data mining in spark streaming. In *2015 IEEE International Conference on Data Mining Workshop (ICDMW)*, pp. 1608–1611. IEEE.

Bootkrajang, J. and J. Chaijaruwanich (2020). Towards instance-dependent label noise-tolerant classification: a probabilistic approach. *Pattern Analysis and Applications 23*(1), 95–111.

Bootkrajang, J. and A. Kabán (2012). Label-noise robust logistic regression and its applications. In *Joint European conference on machine learning and knowledge discovery in databases*, pp. 143–158. Springer.

Bootkrajang, J. and A. Kabán (2013). Classification of mislabelled microarrays using robust sparse logistic regression. *Bioinformatics 29*(7), 870–877.

Bootkrajang, J. and A. Kabán (2014). Learning kernel logistic regression in the presence of class label noise. *Pattern Recognition 47*(11), 3641–3655.

Bordes, A., L. Bottou, and P. Gallinari (2009). Sgd-qn: Careful quasi-newton stochastic gradient descent. *Journal of Machine Learning Research 10*, 1737–1754.

Breiman, L. (1995). Better subset regression using the nonnegative garrote. *Technometrics 37*(4), 373–384.

Cai, T. T. and J. Lv (2007). Discussion: The dantzig selector: statistical estimation when p is much larger than n. *The Annals of Statistics 35*(6), 2365–2369.

Candes, E. and T. Tao (2007). The dantzig selector: Statistical estimation when p is much larger than n. *The annals of Statistics 35*(6), 2313–2351.

Cannings, T. I., Y. Fan, and R. J. Samworth (2020). Classification with imperfect training labels. *Biometrika 107*(2), 311–330.

Casella, F. and B. Bachmann (2021). On the choice of initial guesses for the newton-raphson algorithm. *Applied Mathematics and Computation 398*, 125991.

Cervantes, J., F. Garcia-Lamont, L. Rodríguez-Mazahua, and A. Lopez (2020). A comprehensive survey on support vector machine classification: Applications, challenges and trends. *Neurocomputing 408*, 189–215.

Cheng, J., T. Liu, K. Ramamohanarao, and D. Tao (2020). Learning with bounded instance and label-dependent label noise. In *International Conference on Machine Learning*, pp. 1789–1799. PMLR.

Cortes, C. and V. Vapnik (1995). Support-vector networks. *Machine learning 20*(3), 273–297.

Dobriban, E. and S. Wager (2018). High-dimensional asymptotics of prediction: Ridge regression and classification. *The Annals of Statistics 46*(1), 247–279.

Donoho, D. L. et al. (2000). High-dimensional data analysis: The curses and blessings of dimensionality. *AMS math challenges lecture 1*(2000), 32.

Dormann, C. F., J. Elith, S. Bacher, C. Buchmann, G. Carl, G. Carré, J. R. G. Marquéz, B. Gruber, B. Lafourcade, P. J. Leitão, et al. (2013). Collinearity: a review of methods to deal with it and a simulation study evaluating their performance. *Ecography 36*(1), 27–46.

Efron, B., T. Hastie, I. Johnstone, and R. Tibshirani (2004). Least angle regression. *The Annals of statistics 32*(2), 407–499.

Efron, B., T. Hastie, and R. Tibshirani (2007). Discussion: The dantzig selector: Statistical estimation when p is much larger than n. *The Annals of Statistics 35*(6), 2358–2364.

Fan, J. and Y. Fan (2008). High dimensional classification using features annealed independence rules. *Annals of statistics 36*(6), 2605.

Fan, J., Y. Fan, and Y. Wu (2011). High-dimensional classification. In *High-dimensional data analysis*, pp. 3–37. World Scientific.

Fan, J., F. Han, and H. Liu (2014). Challenges of big data analysis. *National science review 1*(2), 293–314.

Fan, J. and R. Li (1999). Variable selection via penalized likelihood. *Department of Statistics UCLA*.

Fan, J. and R. Li (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American statistical Association 96*(456), 1348–1360.

Fan, J. and J. Lv (2008). Sure independence screening for ultrahigh dimensional feature space. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 70*(5), 849–911.

Fan, J. and J. Lv (2010). A selective overview of variable selection in high dimensional feature space. *Statistica Sinica*, 101–148.

Fan, J. and J. Lv (2018). Sure independence screening. *Wiley StatsRef: Statistics Reference Online*.

Fan, J., R. Samworth, and Y. Wu (2009). Ultrahigh dimensional feature selection: beyond the linear model. *The Journal of Machine Learning Research 10*, 2013–2038.

Fan, J. and R. Song (2010). Sure independence screening in generalized linear models with np-dimensionality. *The Annals of Statistics 38*(6), 3567–3604.

Fang, Y. (2019). Scalable statistical inference for averaged implicit stochastic gradient descent. *Scandinavian Journal of Statistics 46*(4), 987–1002.

Frank, L. E. and J. H. Friedman (1993). A statistical view of some chemometrics regression tools. *Technometrics 35*(2), 109–135.

Frénay, B., A. Kabán, et al. (2014). A comprehensive introduction to label noise. In *ESANN*. Citeseer.

Fuangkhon, P. and T. Tanprasert (2009). An incremental learning algorithm for supervised neural network with contour preserving classification. In *2009 6th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*, Volume 2, pp. 740–743. IEEE.

Goldberger, J. and E. Ben-Reuven (2016). Training deep neural-networks using a noise adaptation layer.

Hassanat, A. B., M. A. Abbadi, G. A. Altarawneh, and A. A. Alhasanat (2014). Solving the problem of the k parameter in the knn classifier using an ensemble learning approach. *arXiv preprint arXiv:1409.0919*.

Hastie, T., R. Tibshirani, and R. J. Tibshirani (2017). Extended comparisons of best subset selection, forward stepwise selection, and the lasso. *arXiv preprint arXiv:1707.08692*.

Hazan, E., A. Agarwal, and S. Kale (2007). Logarithmic regret algorithms for online convex optimization. *Machine Learning 69*(2), 169–192.

He, X., L. Wang, and H. G. Hong (2013). Quantile-adaptive model-free variable screening for high-dimensional heterogeneous data.

Hinton, G., O. Vinyals, J. Dean, et al. (2015). Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531 2*(7).

Hoerl, A. E. and R. W. Kennard (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics 12*(1), 55–67.

Hunter, D. R. and R. Li (2005). Variable selection using mm algorithms. *Annals of statistics 33*(4), 1617.

Jolliffe, I. T. and J. Cadima (2016). Principal component analysis: a review and recent developments. *Philosophical transactions of the royal society A: Mathematical, Physical and Engineering Sciences 374*(2065), 20150202.

Jorgensen, B. (1997). *The theory of dispersion models*. CRC Press.

Kalantar, B., B. Pradhan, S. A. Naghibi, A. Motevalli, and S. Mansor (2018). Assessment of the effects of training data selection on the landslide susceptibility mapping: a comparison between support vector machine (svm), logistic regression (lr) and artificial neural networks (ann). *Geomatics, Natural Hazards and Risk 9*(1), 49–69.

Konečnỳ, J. and P. Richtárik (2013). Semi-stochastic gradient descent methods. *arXiv preprint arXiv:1312.1666*.

Kong, X.-B., Z. Liu, Y. Yao, and W. Zhou (2017). Sure screening by ranking the canonical correlations. *Test 26*(1), 46–70.

Kotsiantis, S. B., I. D. Zaharakis, and P. E. Pintelas (2006). Machine learning: a review of classification and combining techniques. *Artificial Intelligence Review 26*(3), 159–190.

Krause, J., B. Sapp, A. Howard, H. Zhou, A. Toshev, T. Duerig, J. Philbin, and L. Fei-Fei (2016). The unreasonable effectiveness of noisy data for fine-grained recognition. In *European Conference on Computer Vision*, pp. 301–320. Springer.

Lever, J., M. Krzywinski, and N. Altman (2017). Points of significance: Principal component analysis. *Nature methods 14*(7), 641–643.

Li, Y., J. Yang, Y. Song, L. Cao, J. Luo, and L.-J. Li (2017). Learning from noisy labels with distillation. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1910–1918.

Liu, D. C. and J. Nocedal (1989). On the limited memory bfgs method for large scale optimization. *Mathematical programming 45*(1), 503–528.

Liu, L. (2018). Research on logistic regression algorithm of breast cancer diagnose data by machine learning. In *2018 International Conference on Robots & Intelligent System (ICRIS)*, pp. 157–160. IEEE.

Losing, V., B. Hammer, and H. Wersing (2018). Incremental on-line learning: A review and comparison of state of the art algorithms. *Neurocomputing 275*, 1261–1274.

Luo, L. and P. X.-K. Song (2020). Renewable estimation and incremental inference in generalized linear models with streaming data sets. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 82*(1), 69–97.

Mahmood, S. S., D. Levy, R. S. Vasan, and T. J. Wang (2014). The framingham heart study and the epidemiology of cardiovascular disease: a historical perspective. *The lancet 383*(9921), 999–1008.

McCullagh, P. (1983). *Generalized linear models*. Routledge.

Meinshausen, N. (2007). Relaxed lasso. *Computational Statistics & Data Analysis 52*(1), 374–393.

Mishra, S. K., B. Ram, S. K. Mishra, and B. Ram (2019). Quasi-newton methods. *Introduction to Unconstrained Optimization with R*, 245–289.

Morgan, S. P. and J. D. Teachman (1988). Logistic regression: Description, examples, and comparisons. *Journal of Marriage and Family 50*(4), 929–936.

Naifei, Z., X. Qingsong, T. Man-Lai, J. Binyan, C. Ziqi, and W. Hong (2020). High dimensional variable screening under multicollinearity [j]. *Stat 9*(1).

Ng, A. and M. Jordan (2001). On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. *Advances in neural information processing systems 14*.

Nhu, V.-H., A. Shirzadi, H. Shahabi, S. K. Singh, N. Al-Ansari, J. J. Clague, A. Jaafari, W. Chen, S. Miraki, J. Dou, et al. (2020). Shallow landslide susceptibility mapping: A comparison between logistic model tree, logistic regression, naïve bayes tree, artificial neural network, and support vector machine algorithms. *International journal of environmental research and public health 17*(8), 2749.

Parvin, H., H. Alizadeh, and B. Minati (2010). A modification on k-nearest neighbor classifier. *Global Journal of Computer Science and Technology*.

Pavlidis, P., I. Wapinski, and W. S. Noble (2004). Support vector machine classification on the web. *Bioinformatics 20*(4), 586–587.

Pohar, M., M. Blas, and S. Turk (2004). Comparison of logistic regression and linear discriminant analysis: a simulation study. *Metodoloski zvezki 1*(1), 143.

Polyak, B. T. and A. B. Juditsky (1992). Acceleration of stochastic approximation by averaging. *SIAM journal on control and optimization 30*(4), 838–855.

Ridzuan, F. and W. M. N. W. Zainon (2019). A review on data cleansing methods for big data. *Procedia Computer Science 161*, 731–738.

Rish, I. et al. (2001). An empirical study of the naive bayes classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence*, Volume 3, pp. 41–46.

Robbins, H. and S. Monro (1951). A stochastic approximation method. *The annals of mathematical statistics*, 400–407.

Rodríguez, G. (2007). Chapter 3: Logit models for binary data. *Lecture Notes on Generalized Linear Models*.

Rosasco, L. and S. Villa (2015). Learning with incremental iterative regularization. *Advances in Neural Information Processing Systems 28*.

Sakrison, D. J. (1965). Efficient recursive estimation; application to estimating the parameters of a covariance function. *International Journal of Engineering Science 3*(4), 461–483.

Salazar, D. A., J. I. Vélez, and J. C. Salazar (2012). Comparison between svm and logistic regression: Which one is better to discriminate? *Revista Colombiana de Estadística 35*(SPE2), 223–237.

Saldana, D. F. and Y. Feng (2018). Sis: An r package for sure independence screening in ultrahigh-dimensional statistical models. *Journal of Statistical Software 83*, 1–25.

Schraudolph, N. N., J. Yu, and S. Günter (2007). A stochastic quasi-newton method for online convex optimization. In *Artificial intelligence and statistics*, pp. 436–443. PMLR.

Shao, X. and J. Zhang (2014). Martingale difference correlation and its use in high-dimensional variable screening. *Journal of the American Statistical Association 109*(507), 1302–1318.

Sirimongkolkasem, T. and R. Drikvandi (2019). On regularisation methods for analysis of high dimensional data. *Annals of Data Science 6*(4), 737–763.

Stone, C. J. (1977). Consistent nonparametric regression. *The annals of statistics*, 595–620.

Sukhbaatar, S., J. Bruna, M. Paluri, L. Bourdev, and R. Fergus (2014). Training convolutional networks with noisy labels. *arXiv preprint arXiv:1406.2080*.

Székely, G. J., M. L. Rizzo, and N. K. Bakirov (2007). Measuring and testing dependence by correlation of distances.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological) 58*(1), 267–288.

Toulis, P., E. Airoldi, and J. Rennie (2014). Statistical analysis of stochastic gradient methods for generalized linear models. In *International Conference on Machine Learning*, pp. 667–675. PMLR.

Tran, D., P. Toulis, and E. M. Airoldi (2015). Stochastic gradient descent methods for estimation with large data sets. *arXiv preprint arXiv:1509.06459*.

Tsangaratos, P. and I. Ilia (2016). Comparison of a logistic regression and naïve bayes classifier in landslide susceptibility assessments: The influence of models complexity and training dataset size. *Catena 145*, 164–179.

Tu, J. V. (1996). Advantages and disadvantages of using artificial neural networks versus logistic regression for predicting medical outcomes. *Journal of clinical epidemiology 49*(11), 1225–1231.

Vijayakumar, S. and S. Schaal (2000). Locally weighted projection regression: An o (n) algorithm for incremental real time learning in high dimensional space. In *Proceedings of the seventeenth international conference on machine learning (ICML 2000)*, Volume 1, pp. 288–293. Morgan Kaufmann.

Wang, F., S. Mukherjee, S. Richardson, and S. M. Hill (2020). High-dimensional regression in practice: an empirical study of finite-sample prediction, variable selection and ranking. *Statistics and computing 30*(3), 697–719.

Wang, H. (2009). Forward regression for ultra-high dimensional variable screening. *Journal of the American Statistical Association 104*(488), 1512–1524.

Wang, H. (2012). Factor profiled sure independence screening. *Biometrika 99*(1), 15–28.

Wang, S., B. Nan, S. Rosset, and J. Zhu (2011). Random lasso. *The annals of applied statistics 5*(1), 468.

Wilson, P. W., W. P. Castelli, and W. B. Kannel (1987). Coronary risk prediction in adults (the framingham heart study). *The American journal of cardiology 59*(14), G91–G94.

Wold, S., K. Esbensen, and P. Geladi (1987). Principal component analysis. *Chemometrics and intelligent laboratory systems 2*(1-3), 37–52.

Xie, H., J. Li, and H. Xue (2017). A survey of dimensionality reduction techniques based on random projection. *arXiv preprint arXiv:1706.04371*.

Ypma, T. J. (1995). Historical development of the newton–raphson method. *SIAM review 37*(4), 531–551.

Zhang, C.-H. (2010). Nearly unbiased variable selection under minimax concave penalty. *The Annals of statistics 38*(2), 894–942.

Zhang, N., W. Jiang, and Y. Lan (2019). On the sure screening properties of iteratively sure independence screening algorithms.

Zhang, Y., S. Zheng, P. Wu, M. Goswami, and C. Chen (2021). Learning with feature-dependent label noise: a progressive approach. *arXiv preprint arXiv:2103.07756*.

Zhao, P. and B. Yu (2006). On model selection consistency of lasso. *The Journal of Machine Learning Research 7*, 2541–2563.

Zou, H. (2006). The adaptive lasso and its oracle properties. *Journal of the American statistical association 101*(476), 1418–1429.

Zou, H. and T. Hastie (2005). Regularization and variable selection via the elastic net. *Journal of the royal statistical society: series B (statistical methodology) 67*(2), 301–320.

A Appendix: Derivation of estimators for *Classification with mislabelled data*

A.1 Two-step estimation method using LQA and Newton-Raphson algorithm

In this section, we provide a thorough derivation for the two-step estimation process outlined in Section 3.1.2. Given the similarities in the estimation procedures for $\boldsymbol{\beta}$ and those for $\boldsymbol{\eta_0}$ and $\boldsymbol{\eta_1}$, our exposition focuses on the estimation process for the p-dimensional unknown parameter $\boldsymbol{\beta}$, omitting the latter two for conciseness.

For the *first-step estimation*, analysing the resampled data $\{(X_i,\ z_i) :\ i \in \mathcal{D}\}$ with $|\mathcal{D}| = m$, we aim to find the MLE of $\boldsymbol{\beta}$ in (3.4), represented as

$$L_{\mathcal{D}}(\boldsymbol{\beta}) = \sum_{i\in\mathcal{D}} \Big[z_i \text{logit}(\pi_z(X_i)) + \log(1-\pi_z(X_i)) \Big] - \mathbf{P}_{\lambda_\beta}(\|\boldsymbol{\beta}\|_1),$$

which, for simplicity, we express as a combination of the unpenalised log-likelihood function and the penalty term:

$$L_{\mathcal{D}}(\boldsymbol{\beta}) = \ell_{\mathcal{D}}(\boldsymbol{\beta}) - \mathbf{P}_{\lambda_\beta}(\|\boldsymbol{\beta}\|_1). \tag{A.1}$$

Utilising the Newton-Raphson algorithm, the standard update formula for optimising equation (A.1) during the k-th iteration, for $k = 1, \cdots$, is given by:

$$\hat{\boldsymbol{\beta}}^k = \hat{\boldsymbol{\beta}}^{(k-1)} - \Big[\nabla^2 L_{\mathcal{D}}(\hat{\boldsymbol{\beta}}^{(k-1)})\Big]^{-1} \nabla L_{\mathcal{D}}(\hat{\boldsymbol{\beta}}^{(k-1)}), \tag{A.2}$$

where $\nabla L_{\mathcal{D}}(\hat{\boldsymbol{\beta}}^{(k-1)}) = \frac{\partial L_{\mathcal{D}}(\hat{\boldsymbol{\beta}}^{(k-1)})}{\partial \boldsymbol{\beta}}$ and $\nabla^2 L_{\mathcal{D}}(\hat{\boldsymbol{\beta}}^{(k-1)}) = \frac{\partial L^2_{\mathcal{D}}(\hat{\boldsymbol{\beta}}^{(k-1)})}{\partial \boldsymbol{\beta}\partial \boldsymbol{\beta}^\intercal}$.

Assuming the smoothness of the log-likelihood function $\ell(\boldsymbol{\beta})$, its two partial derivatives with respect to $\boldsymbol{\beta}$ remain continuous. Given an estimate of $\boldsymbol{\beta}$ denoted as $\boldsymbol{\beta_0}$ which is close to $\boldsymbol{\beta}$, the LQA application yields

$$\ell_{\mathcal{D}}(\boldsymbol{\beta}) \approx \ell_{\mathcal{D}}(\boldsymbol{\beta_0}) + \nabla\ell_{\mathcal{D}}(\boldsymbol{\beta_0})(\boldsymbol{\beta}-\boldsymbol{\beta_0})^\intercal + \frac{1}{2}(\boldsymbol{\beta}-\boldsymbol{\beta_0})\nabla^2\ell_{\mathcal{D}}(\boldsymbol{\beta_0})(\boldsymbol{\beta}-\boldsymbol{\beta_0})^\intercal, \tag{A.3}$$

where $\nabla\ell_{\mathcal{D}}(\boldsymbol{\beta_0}) = \frac{\partial \ell_{\mathcal{D}}(\boldsymbol{\beta_0})}{\partial \boldsymbol{\beta}}$ and $\nabla^2\ell_{\mathcal{D}}(\boldsymbol{\beta}_0) = \frac{\partial \ell^2_{\mathcal{D}}(\boldsymbol{\beta_0})}{\partial \boldsymbol{\beta}\partial \boldsymbol{\beta}^\intercal}$.

Referring to Fan and Li (2001), we employ the LQA method to approximate the

penalty function $\mathbf{P}_{\lambda_\beta}(\|\boldsymbol{\beta}\|_1)$. For coefficients β_{j0}, $j = 1, \cdots, p$, close to 0, the estimate of β_j is set to 0. Otherwise we have the approximation as

$$[p_\lambda(|\beta_j|)]' = p'_\lambda(|\beta_j|)\mathrm{sgn}(\beta_j) \approx \{p'_\lambda(|\beta_{j0}|)/|\beta_{j0}|\}\beta_j,$$

where $p_\lambda(\cdot)$ is the penalty function with λ as a tuning parameter. For $\beta_j \neq 0$, we then have

$$p_\lambda(|\beta_j|) \approx p_\lambda(|\beta_{j0}|) + \frac{1}{2}\{p'_\lambda(|\beta_{j0}|)/|\beta_{j0}|\}(\beta_j^2 - \beta_{j0}^2). \tag{A.4}$$

We denote the d, $1 \leqslant d \leqslant p$, non-zero coefficients of $\boldsymbol{\beta_0}$ as

$$\Sigma_\lambda(\boldsymbol{\beta_0}) = \mathrm{diag}\{p'_\lambda(|\beta_{10}|)/|\beta_{10}|, \cdots, p'_\lambda(|\beta_{d0}|)/|\beta_{d0}|\}.$$

Considering equations (A.3) and (A.4), and after omitting constant terms, the optimisation of (A.1) simplifies to maximising the following equation:

$$\ell_{\mathcal{D}}(\boldsymbol{\beta_0}) + \nabla\ell_{\mathcal{D}}(\boldsymbol{\beta_0})(\boldsymbol{\beta} - \boldsymbol{\beta_0})^\mathsf{T} + \frac{1}{2}(\boldsymbol{\beta} - \boldsymbol{\beta_0})\nabla^2\ell_{\mathcal{D}}(\boldsymbol{\beta_0})(\boldsymbol{\beta} - \boldsymbol{\beta_0})^\mathsf{T} - \frac{1}{2}m\boldsymbol{\beta}\Sigma_\lambda(\boldsymbol{\beta_0})\boldsymbol{\beta}^\mathsf{T}.$$

Subsequently, referring to the update formula by Newton-Raphson algorithm (A.2), in the k-th iteration, where $k = 1, \cdots$, the estimator takes the form:

$$\hat{\boldsymbol{\beta}}^k = \hat{\boldsymbol{\beta}}^{(k-1)} - \left\{\nabla^2\ell_{\mathcal{D}}(\hat{\boldsymbol{\beta}}^{(k-1)}) - m\Sigma_\lambda(\hat{\boldsymbol{\beta}}^{(k-1)})\right\}^{-1}\left\{\nabla\ell_{\mathcal{D}}(\hat{\boldsymbol{\beta}}^{(k-1)}) - m\Sigma_\lambda(\hat{\boldsymbol{\beta}}^{(k-1)})\hat{\boldsymbol{\beta}}^{(k-1)}\right\}, \tag{A.5}$$

and the iteration halts upon convergence. The MLE of $\boldsymbol{\beta}$, denoted as $\boldsymbol{\beta}^{(0)}$. Among the components $\beta_j^{(0)}$, where $j = 1, \cdots, p$, those with non-zero coefficients satisfy the condition

$$\frac{\partial\ell_{\mathcal{D}}(\boldsymbol{\beta}^{(0)})}{\partial\beta_j} - mp'_\lambda(|\beta_j^{(0)}|)\mathrm{sgn}(\beta_j^{(0)}) = 0.$$

In the *second-step estimation*, we concurrently analyse both resampled and unresampled data, represented as $\{(X_i, y_i, z_i)\}$, $i = 1, \cdots, n$. It is pertinent to note that the true label of observation y_i, denoted as z_i, is only identifiable within the resampled dataset, that is when $i \in \mathcal{D}$. We aim to find the maximisers of (3.1).

As illustrated in Section 3.1.2, during each k-th iteration, where $k = 1, \cdots$, once steps (2) and (3) have been executed, we obtain the estimates $\boldsymbol{\eta}_0^{(k)}$ and $\boldsymbol{\eta}_1^{(k)}$, along with their associated tuning parameters $\lambda_{\boldsymbol{\eta}_0^{(k)}}$ and $\lambda_{\boldsymbol{\eta}_a^{(k)}}$. The focus then shifts to step

(3) of the second-step estimation, aimed at identifying the MLE of $\boldsymbol{\beta}$ in $L(\boldsymbol{\beta},\ \boldsymbol{\eta_0},\ \boldsymbol{\eta_1})$.

$$
\begin{aligned}
& L(\boldsymbol{\beta},\ \boldsymbol{\eta}_{\mathbf{0}}^{(k)},\ \boldsymbol{\eta}_{\mathbf{1}}^{(k)}) \\
= & \sum_{i\in\mathcal{D}^c} \big(y_i \text{logit}\left[\pi_{y|0}(X_i) + \left\{\pi_{y|1}(X_i) - \pi_{y|0}(X_i)\right\}\pi_z(X_i)\right] + \\
& \qquad \log\left[1 - \pi_{y|0}(X_i) - \left\{\pi_{y|1}(X_i) - \pi_{y|0}(X_i)\right\}\pi_z(X_i)\right]\big) \\
& + \sum_{i\in\mathcal{D}} (1-z_i)\left\{y_i(X_i^T\boldsymbol{\eta}_{\mathbf{0}}^{(k)}) + \log(1-\pi_{y|0}(X_i)) + (1-z_i)\log(1-\pi_z(X_i))\right\} \\
& + \sum_{i\in\mathcal{D}} z_i\left\{y_i(X_i^T\boldsymbol{\eta}_{\mathbf{1}}^{(k)}) + \log(1-\pi_{y|1}(X_i)) + z_i\log(\pi_z(X_i))\right\} \\
& - \mathbf{P}_{\lambda_\beta}(\|\boldsymbol{\beta}\|_1) - \mathbf{P}_{\lambda_{\boldsymbol{\eta}_{\mathbf{0}}^{(k)}}}(\|\boldsymbol{\eta_0}\|_1) - \mathbf{P}_{\lambda_{\boldsymbol{\eta}_{\mathbf{1}}^{(k)}}}(\|\boldsymbol{\eta_1}\|_1). \qquad \text{(A.6)}
\end{aligned}
$$

Given the known values of $\boldsymbol{\eta}_{\mathbf{0}}^{(k)}, \boldsymbol{\eta}_{\mathbf{1}}^{(k)}$ and $\pi_{y|0}(\cdot), \pi_{y|1}(\cdot)$, $\mathbf{P}_{\lambda_{\boldsymbol{\eta}_{\mathbf{0}}^{(k)}}}(\|\boldsymbol{\eta_0}\|_1)$ and $\mathbf{P}_{\lambda_{\boldsymbol{\eta}_{\mathbf{1}}^{(k)}}}(\|\boldsymbol{\eta_1}\|_1)$, the equation for $L(\boldsymbol{\beta},\ \boldsymbol{\eta}_{\mathbf{0}}^{(k)},\ \boldsymbol{\eta}_{\mathbf{1}}^{(k)})$ simplifies to:

$$L(\boldsymbol{\beta}) = \ell(\boldsymbol{\beta}) - \mathbf{P}_{\lambda_\beta}(\|\boldsymbol{\beta}\|_1) - C,$$

where C is a constant and $C = \mathbf{P}_{\lambda_{\boldsymbol{\eta}_{\mathbf{0}}^{(k)}}}(\|\boldsymbol{\eta_0}\|_1) + \mathbf{P}_{\lambda_{\boldsymbol{\eta}_{\mathbf{1}}^{(k)}}}(\|\boldsymbol{\eta_1}\|_1)$.

Here

$$\ell(\boldsymbol{\beta}) = \ell_{\mathcal{D}^c}(\mathbf{y}, \boldsymbol{\beta}) + \ell_{\mathcal{D}}(\boldsymbol{z}, \boldsymbol{\beta}),$$

and

$$
\begin{aligned}
\ell_{\mathcal{D}^c}(\mathbf{y}, \boldsymbol{\beta}) &= \sum_{i\in\mathcal{D}^c} \ell(y_i, \boldsymbol{\beta}) \\
&= \big(y_i \text{logit}\left[\pi_{y|0}(X_i) + \left\{\pi_{y|1}(X_i) - \pi_{y|0}(X_i)\right\}\pi_z(X_i)\right] + \\
&\qquad \log\left[1 - \pi_{y|0}(X_i) - \left\{\pi_{y|1}(X_i) - \pi_{y|0}(X_i)\right\}\pi_z(X_i)\right]\big) \\
\ell_{\mathcal{D}}(\boldsymbol{z}, \boldsymbol{\beta}) &= \sum_{i\in\mathcal{D}} \ell(z_i, \boldsymbol{\beta}) \\
&= (1-z_i)\left\{y_i(X_i^T\boldsymbol{\eta}_{\mathbf{0}}^{(k)}) + \log(1-\pi_{y|0}(X_i)) + (1-z_i)\log(1-\pi_z(X_i))\right\} \\
&\qquad + z_i\left\{y_i(X_i^T\boldsymbol{\eta}_{\mathbf{1}}^{(k)}) + \log(1-\pi_{y|1}(X_i)) + z_i\log(\pi_z(X_i))\right\}.
\end{aligned}
$$

Analogous to the first-step estimation, we use LQA to approximate both $\ell(\boldsymbol{\beta})$ and $\mathbf{P}_{\lambda_\beta}(\|\boldsymbol{\beta}\|_1)$. Applying the Newton-Raphson algorithm, in the k-th iteration, $k = 1, \cdots,$

the estimator for $\boldsymbol{\beta}$ is given by

$$\boldsymbol{\beta}^{k} = \boldsymbol{\beta}^{(k-1)} - \left\{ \nabla^2 \ell(\boldsymbol{\beta}^{(k-1)}) - n\Sigma_\lambda(\boldsymbol{\beta}^{(k-1)}) \right\}^{-1} \left\{ \nabla \ell(\boldsymbol{\beta}^{(k-1)}) - n\Sigma_\lambda(\boldsymbol{\beta}^{(k-1)})\boldsymbol{\beta}^{(k-1)} \right\},$$

once the iteration achieves convergence, we obtain the MLE for $\boldsymbol{\beta}$ is obtained and denoted as $\hat{\boldsymbol{\beta}}^{(k)}$. The components of $\hat{\boldsymbol{\beta}}^{(k)}$ are denoted as $\hat{\beta}_j^{(k)}$, for $j = 1, \cdots, p$. Among these, the non-zero components fulfil the condition

$$\frac{\partial \ell(\hat{\beta}_j^{(k)})}{\partial \beta_j} - np'_\lambda(|\hat{\beta}_j^{(k)}|)\mathrm{sgn}(\hat{\beta}_j^{(k)}) = 0,$$

where $\ell(\hat{\beta}_j^{(k)}) = \ell_{\mathcal{D}^c}(\hat{\beta}_j^{(k)}) + \ell_{\mathcal{D}}(\hat{\beta}_j^{(k)})$.

B Appendix: Tables for *Classification with mislabelled data*

Table B.1: SDs and ESEs for non-zero $\boldsymbol{\beta}_{120\times1}$ coefficient estimates

	n_{cv}	10	20	30	40
β_2	$mLPOCV_{UR}$	0.3415(0.1574,0.2334)	0.3037(0.1605,0.2380)	0.2802(0.1614,0.2393)	0.2539(0.1651,0.2447)
	$mLPOCV_{S}$	0.3371(0.1581,0.2344)	0.3037(0.1605,0.2380)	0.2795(0.1615,0.2394)	0.2363(0.1651,0.2447)
	$mLPOCV_{CD}$	0.3312(0.1574,0.2334)	0.3037(0.1605,0.2380)	0.2795(0.1615,0.2394)	0.2271(0.1651,0.2447)
β_4	$mLPOCV_{UR}$	0.3427(0.1615,0.2395)	0.2741(0.1620,0.2401)	0.2435(0.1643,0.2436)	0.2506(0.1633,0.2420)
	$mLPOCV_{S}$	0.3059(0.1620,0.2402)	0.2804(0.1621,0.2403)	0.2557(0.1643,0.2436)	0.2587(0.1630,0.2417)
	$mLPOCV_{CD}$	0.3000(0.1620,0.2402)	0.2813(0.1621,0.2403)	0.2557(0.1643,0.2436)	0.2587(0.1630,0.2417)
β_5	$mLPOCV_{UR}$	0.2832(0.1739,0.2578)	0.2627(0.1772,0.2627)	0.2817(0.1750,0.2594)	0.2425(0.1744,0.2586)
	$mLPOCV_{S}$	0.2962(0.1737,0.2575)	0.2651(0.1775,0.2632)	0.2872(0.1750,0.2594)	0.2485(0.1738,0.2577)
	$mLPOCV_{CD}$	0.3008(0.1731,0.2566)	0.2651(0.1775,0.2632)	0.2872(0.1761,0.2611)	0.2485(0.1738,0.2577)
β_7	$mLPOCV_{UR}$	0.3174(0.1624,0.2408)	0.2344(0.1682,0.2494)	0.2502(0.1640,0.2432)	0.2345(0.1655,0.2454)
	$mLPOCV_{S}$	0.3347(0.1633,0.2422)	0.2584(0.1686,0.2500)	0.2531(0.1640,0.2432)	0.2417(0.1655,0.2454)
	$mLPOCV_{CD}$	0.3333(0.1633,0.2422)	0.2584(0.1686,0.2500)	0.2531(0.1640,0.2432)	0.2417(0.1655,0.2454)

* For each cell, the value outside the brackets represents SD, while the first and second numbers inside the brackets represent ESE_m and ESE_o, respectively.

* The outcomes obtained from the method described in Section 3.4.2 are denoted as $mLPOCV_{UR}$, those from Section 3.4.3 as $mLPOCV_S$, and the results from Section 3.4.4 as $mLPOCV_{CD}$.

Table B.2: SDs and ESEs for non-zero $\boldsymbol{\beta}_{150\times1}$ coefficient estimates

Case 1: $p = 15$		**Order1**	**Order2**	**Order3**	**Order4**
β_2	$m = 200$	0.3578(0.3222,0.4778)	0.3786(0.3333,0.4941)	0.3646(0.3322,0.4925)	0.3438(0.3167,0.4696)
	$m = 400$	0.2235(0.2605,0.3862)	0.2248(0.2633,0.3904)	0.2209(0.2633,0.3904)	0.2124(0.2628,0.3897)
β_4	$m = 200$	0.3740(0.2765,0.4100)	0.2952(0.2815,0.4174)	0.2965(0.2792,0.4139)	0.3432(0.2738,0.4059)
	$m = 400$	0.1926(0.2136,0.3167)	0.1954(0.2134,0.3164)	0.1951(0.2131,0.3159)	0.1895(0.2129,0.3157)
Case 2: $p = 150$		**Order1**	**Order2**	**Order3**	**Order4**
β_2	$m = 200$	0.3695(0.2890,0.4284)	0.3695(0.2890,0.4284)	0.3695(0.2890,0.4284)	0.3695(0.2890,0.4284)
	$m = 400$	0.2912(0.2261,0.3352)	0.2931(0.2261,0.3351)	0.2875(0.2223,0.3295)	0.2812(0.2208,0.3273)
β_4	$m = 200$	0.3905(0.2486,0.3685)	0.3905(0.2486,0.3685)	0.3905(0.2486,0.3685)	0.3905(0.2486,0.3685)
	$m = 400$	0.2261(0.1907,0.2827)	0.2278(0.1907,0.2827)	0.2213(0.1901,0.2818)	0.2277(0.1873,0.2777)

* For each cell, the value outside the brackets represents SD, while the first and second numbers inside the brackets represent ESE_m and ESE_o, respectively.

Table B.3: SDs and ESEs for non-zero $\boldsymbol{\beta}_{15\times1}$ coefficient estimates

	ρ	0	0.2	0.5
β_2	$\hat{\boldsymbol{\beta}}$	0.206(0.229,0.340)	0.245(0.224,0.332)	0.312(0.335,0.496)
	$\hat{\boldsymbol{\beta}}_{\|\boldsymbol{\eta}_0}$	0.219(0.229,0.339)	0.226(0.222,0.329)	0.099(0.333,0.493)
	$\hat{\boldsymbol{\beta}}_{\|\boldsymbol{\eta}_1}$	0.209(0.231,0.343)	0.208(0.225,0.333)	0.095(0.338,0.500)
	$\hat{\boldsymbol{\beta}}_{\|\boldsymbol{\eta}_0,\boldsymbol{\eta}_1}$	0.219(0.232,0.344)	0.210(0.223,0.330)	0.096(0.332,0.492)
β_3	$\hat{\boldsymbol{\beta}}$	0.203(0.188,0.279)	0.200(0.181,0.268)	0.244(0.192,0.284)
	$\hat{\boldsymbol{\beta}}_{\|\boldsymbol{\eta}_0}$	0.196(0.191,0.283)	0.200(0.183,0.271)	0.234(0.190,0.282)
	$\hat{\boldsymbol{\beta}}_{\|\boldsymbol{\eta}_1}$	0.197(0.188,0.278)	0.200(0.184,0.273)	0.226(0.194,0.288)
	$\hat{\boldsymbol{\beta}}_{\|\boldsymbol{\eta}_0,\boldsymbol{\eta}_1}$	0.246(0.190,0.282)	0.219(0.185,0.274)	0.227(0.191,0.283)

* For each cell, the value outside the brackets represents SD, while the first and second numbers inside the brackets represent $\mathrm{ESE_m}$ and $\mathrm{ESE_o}$, respectively.

Table B.4: SDs and ESEs for non-zero $\boldsymbol{\beta}_{15\times1}$ coefficient estimates

	m	300	400	500
β_2	$\hat{\boldsymbol{\beta}}$	0.211(0.166,0.247)	0.163(0.144,0.214)	0.138(0.129,0.192)
	$\boldsymbol{\beta}^{(0)}$	0.238(0.174,0.258)	0.160(0.148,0.219)	0.122(0.131,0.194)
	$\hat{\boldsymbol{\beta}}^*$	0.089(0.093,0.138)	0.089(0.093,0.138)	0.089(0.093,0.138)
	$\hat{\boldsymbol{\beta}}_{\mathrm{R}}$	-	-	-
	$\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$	1.442(0.004,0.006)	1.431(0.004,0.006)	1.439(0.003,0.004)
β_3	$\hat{\boldsymbol{\beta}}$	0.151(0.151,0.223)	0.138(0.131,0.193)	0.113(0.117,0.173)
	$\boldsymbol{\beta}^{(0)}$	0.205(0.154,0.229)	0.127(0.134,0.199)	0.119(0.120,0.179)
	$\hat{\boldsymbol{\beta}}^*$	0.080(0.084,0.124)	0.080 (0.084,0.124)	0.080(0.084,0.124)
	$\hat{\boldsymbol{\beta}}_{\mathrm{R}}$	-	-	-
	$\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$	0.815(0,0)	0.811(0,0)	0.812(0,0)
β_4	$\hat{\boldsymbol{\beta}}$	0.181(0.154,0.229)	0.150(0.134,0.199)	0.138(0.119,0.177)
	$\boldsymbol{\beta}^{(0)}$	0.206(0.157,0.233)	0.141(0.135,0.200)	0.141(0.121,0.179)
	$\hat{\boldsymbol{\beta}}^*$	0.076(0.084,0.125)	0.076(0.084,0.125)	0.076(0.084,0.125)
	$\hat{\boldsymbol{\beta}}_{\mathrm{R}}$	-	-	-
	$\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$	0.810(0,0)	0.804(0.001,0.001)	0.808(0,0)
β_5	$\hat{\boldsymbol{\beta}}$	0.211(0.171,0.253)	0.158(0.147,0.219)	0.149(0.134,0.198)
	$\boldsymbol{\beta}^{(0)}$	0.212(0.173,0.256)	0.158(0.148,0.219)	0.150(0.135,0.200)
	$\hat{\boldsymbol{\beta}}^*$	0.104(0.093,0.138)	0.104(0.093,0.138)	0.104(0.093,0.138)
	$\hat{\boldsymbol{\beta}}_{\mathrm{R}}$	-	-	-
	$\hat{\boldsymbol{\beta}}_{\mathrm{CD}}$	1.456(0.002,0.004)	1.448(0.003,0.004)	1.444(0.003,0.004)

* For each cell, the value outside the brackets represents SD, while the first and second numbers inside the brackets represent $\mathrm{ESE_m}$ and $\mathrm{ESE_o}$, respectively.

* In some of the trials in a 100-repetition experiment, the method that directly uses the corrupted datasets fails to converge, leading to the inability to generate the SDs, and $\mathrm{ESE_m}$s and $\mathrm{ESE_o}$s of $\hat{\boldsymbol{\beta}}_{\mathrm{R}}$s.

Table B.5: SDs and ESEs for non-zero $\boldsymbol{\beta}_{10\times1}$ coefficient estimates

n		1000	2000	3000	5000
β_2	$\hat{\boldsymbol{\beta}}$	0.2902(0.2475,0.3670)	0.3254(0.2484,0.3683)	0.3760(0.2476,0.3670)	0.3701(0.2314,0.3430)
	$\hat{\boldsymbol{\beta}}_{\lvert\hat{M}_{y\lvert0},\hat{M}_{y\lvert1}}$	0.2977(0.2532,0.3753)	0.3078(0.2520,0.3736)	0.3345(0.2491,0.3693)	0.4733(0.2397,0.3553)
β_3	$\hat{\boldsymbol{\beta}}$	0.2564(0.2148,0.3184)	0.2587(0.2149,0.3186)	0.2643(0.2115,0.3136)	0.3208(0.2093,0.3103)
	$\hat{\boldsymbol{\beta}}_{\lvert\hat{M}_{y\lvert0},\hat{M}_{y\lvert1}}$	0.2776(0.2221,0.3293)	0.2931(0.2187,0.3242)	0.3111(0.2188,0.3243)	0.3519(0.2080,0.3084)
β_5	$\hat{\boldsymbol{\beta}}$	0.3357(0.2449,0.3631)	0.3374(0.2422,0.3591)	0.3106(0.2371,0.3516)	0.3514(0.2312,0.3428)
	$\hat{\boldsymbol{\beta}}_{\lvert\hat{M}_{y\lvert0},\hat{M}_{y\lvert1}}$	0.2945(0.2345,0.3477)	0.3340(0.2322,0.3442)	0.4118(0.2383,0.3533)	0.6075(0.2326,0.3448)

* For each cell, the value outside the brackets represents SD, while the first and second numbers inside the brackets represent $\mathrm{ESE_m}$ and $\mathrm{ESE_o}$, respectively.

* The experiments utilise the same resampled datasets with a fixed size of $m = 300$ while the sizes of the training datasets vary and are denoted by n.

Table B.6: SDs and ESEs for non-zero $\boldsymbol{\beta}_{10\times1}$ coefficient estimates

m		300	600	900
β_2	$\hat{\boldsymbol{\beta}}$	0.3692(0.2426,0.3596)	0.1779(0.1746,0.2589)	0.1490(0.1411,0.2092)
	$\hat{\boldsymbol{\beta}}_{\lvert\hat{M}_{y\lvert0},\hat{M}_{y\lvert1}}$	0.2841(0.2261,0.3352)	0.1681(0.1698,0.2518)	0.1447(0.1399,0.2074)
β_3	$\hat{\boldsymbol{\beta}}$	0.2164(0.1866,0.2767)	0.1366(0.1427,0.2115)	0.1127(0.1188,0.1762)
	$\hat{\boldsymbol{\beta}}_{\lvert\hat{M}_{y\lvert0},\hat{M}_{y\lvert1}}$	0.5799(0.1959,0.2904)	0.2419(0.1421,0.2107)	0.1362(0.1182,0.1753)
β_5	$\hat{\boldsymbol{\beta}}$	0.2602(0.2422,0.3590)	0.1871(0.1704,0.2527)	0.1449(0.1368,0.2029)
	$\hat{\boldsymbol{\beta}}_{\lvert\hat{M}_{y\lvert0},\hat{M}_{y\lvert1}}$	0.2635(0.2248,0.3332)	0.1667(0.1635,0.2424)	0.1684(0.1345,0.1995)

* For each cell, the value outside the brackets represents SD, while the first and second numbers inside the brackets represent $\mathrm{ESE_m}$ and $\mathrm{ESE_o}$, respectively.

* The experiments utilise the same training datasets with a fixed size of $n = 3000$ while the sizes of the resampled datasets vary and are denoted by m.

Table B.7: SDs and ESEs for non-zero $\boldsymbol{\beta}_{150\times1}$ coefficient estimates

$m = 300$, ρ		0	0.2	0.5
β_2	$\hat{\boldsymbol{\beta}}$	0.2836(0.2023,0.2999)	0.2313(0.1976,0.2930)	0.2485(0.1990,0.2950)
	$\hat{\boldsymbol{\beta}}_{\mathrm{SIS}}$	0.2917(0.1866,0.2767)	0.2271(0.1810,0.2684)	0.9149(0.1349,0.2000)
β_3	$\hat{\boldsymbol{\beta}}$	0.3439(0.2348,0.3481)	0.2992(0.2338,0.3467)	0.2984(0.2448,0.3629)
	$\hat{\boldsymbol{\beta}}_{\mathrm{SIS}}$	0.3055(0.2229,0.3304)	0.2595(0.2180,0.3233)	2.6684(0,0)
β_4	$\hat{\boldsymbol{\beta}}$	0.2309(0.2017,0.2991)	0.2462(0.2021,0.2996)	0.2453(0.1979,0.2935)
	$\hat{\boldsymbol{\beta}}_{\mathrm{SIS}}$	0.2480(0.2000,0.2965)	0.2359(0.1920,0.2847)	1.1188(0.1380,0.2047)

* For each cell, the value outside the brackets represents SD, while the first and second numbers inside the brackets represent $\mathrm{ESE_m}$ and $\mathrm{ESE_o}$, respectively.

Table B.8: SDs and ESEs for non-zero $\boldsymbol{\beta}_{350\times1}$ coefficient estimates

$m = 300$, ρ		0	0.2	0.5
β_2	$\hat{\boldsymbol{\beta}}_{\text{SIS}}$	0.2642(0.1880,0.2787)	0.2887(0.1906,0.2825)	0.9086(0.1337,0.1982)
	$\hat{\boldsymbol{\beta}}^*_{\text{SIS}}$	0.1163(0.1144,0.1696)	0.1296(0.1109,0.1644)	0.1380(0.1116,0.1654)
β_3	$\hat{\boldsymbol{\beta}}_{\text{SIS}}$	0.3172(0.2271,0.3367)	0.3303(0.2296,0.3404)	2.6686(0,0)
	$\hat{\boldsymbol{\beta}}^*_{\text{SIS}}$	0.1744(0.1329,0.1971)	0.1606(0.1294,0.1919)	0.1646(0.1372,0.2034)
β_4	$\hat{\boldsymbol{\beta}}_{\text{SIS}}$	0.2496(0.2020,0.2995)	0.2697(0.2016,0.2990)	1.0293(0.1420,0.2105)
	$\hat{\boldsymbol{\beta}}^*_{\text{SIS}}$	0.1213(0.1154,0.1712)	0.1230(0.1108,0.1643)	0.1642(0.1123,0.1664)

* For each cell, the value outside the brackets represents SD, while the first and second numbers inside the brackets represent ESE_{m} and ESE_{o}, respectively.

Table B.9: SDs and ESEs for non-zero $\boldsymbol{\beta}_{3000\times1}$ coefficient estimates

ρ			0	0.2	0.5
$p = 1010$, $m = 400$	β_2	$\hat{\boldsymbol{\beta}}_{\text{SIS}}$	0.2058(0.1592,0.2360)	0.2040(0.1576,0.2336)	1.1019(0.1131,0.1676)
		$\hat{\boldsymbol{\beta}}_{\text{ISIS}}$	0.2232(0.1646,0.2440)	0.2171(0.1607,0.2383)	0.2570(0.1584,0.2348)
	β_3	$\hat{\boldsymbol{\beta}}_{\text{SIS}}$	0.2562(0.1951,0.2892)	0.2671(0.1931,0.2863)	2.6679(0,0)
		$\hat{\boldsymbol{\beta}}_{\text{ISIS}}$	0.2705(0.2027,0.3004)	0.2761(0.1944,0.2882)	0.2953(0.2033,0.3014)
	β_4	$\hat{\boldsymbol{\beta}}_{\text{SIS}}$	0.1682(0.1696,0.2514)	0.2117(0.1665,0.2469)	1.1331(0.1173,0.1739)
		$\hat{\boldsymbol{\beta}}_{\text{ISIS}}$	0.1963(0.1746,0.2589)	0.2232(0.1677,0.2487)	0.2599(0.1665,0.2469)
$p = 3000$, $m = 300$	β_2	$\hat{\boldsymbol{\beta}}_{\text{SIS}}$	0.3026(0.2119,0.3141)	0.2962(0.2033,0.3014)	1.2831(0.1471,0.2181)
		$\hat{\boldsymbol{\beta}}_{\text{ISIS}}$	0.2552(0.1986,0.2945)	0.2065(0.1923,0.2851)	0.3066(0.1977,0.2931)
	β_3	$\hat{\boldsymbol{\beta}}_{\text{SIS}}$	0.3639(0.2562,0.3798)	0.3377(0.2428,0.3599)	2.6686(0,0)
		$\hat{\boldsymbol{\beta}}_{\text{ISIS}}$	0.3514(0.2322,0.3443)	0.3050(0.2310,0.3425)	0.3887(0.2449,0.3631)
	β_4	$\hat{\boldsymbol{\beta}}_{\text{SIS}}$	0.3013(0.2194,0.3253)	0.2718(0.2114,0.3134)	1.3154(0.1454,0.2155)
		$\hat{\boldsymbol{\beta}}_{\text{ISIS}}$	0.2232(0.2003,0.2970)	0.2355(0.1965,0.2914)	0.2530(0.1968,0.2918)

* For each cell, the value outside the brackets represents SD, while the first and second numbers inside the brackets represent ESE_{m} and ESE_{o}, respectively.

Table B.10: SDs and ESEs for non-zero $\boldsymbol{\beta}$ coefficient estimates

	p	350	1500	3000
β_2	$\hat{\boldsymbol{\beta}}_{\text{ISIS}}$	0.3845(0.2026,0.3004)	0.3538(0.2053,0.3044)	0.3066(0.1977,0.2931)
	$\hat{\boldsymbol{\beta}}^*_{\text{ISIS}}$	0.1274(0.1099,0.1630)	0.1172(0.1097,0.1626)	0.1285(0.1094,0.1622)
β_3	$\hat{\boldsymbol{\beta}}_{\text{ISIS}}$	0.4188(0.2578,0.3822)	0.4106(0.2525,0.3743)	0.3887(0.2449,0.3631)
	$\hat{\boldsymbol{\beta}}^*_{\text{ISIS}}$	0.1537(0.1341,0.1988)	0.1493(0.1339,0.1985)	0.1565(0.1335,0.1979)
β_4	$\hat{\boldsymbol{\beta}}_{\text{ISIS}}$	0.3284(0.2090,0.3098)	0.3072(0.2049, 0.3037)	0.2530(0.1968, 0.2918)
	$\hat{\boldsymbol{\beta}}^*_{\text{ISIS}}$	0.1437(0.1096,0.1625)	0.1417(0.1096,0.1625)	0.1381(0.1092,0.1619)

* For each cell, the value outside the brackets represents SD, while the first and second numbers inside the brackets represent ESE_{m} and ESE_{o}, respectively.

C Appendix: Tables for *Online algorithms for streaming data*

C.1 Simulation study

Table C.1: SDs and ESEs for non-zero $\boldsymbol{\beta}_{15\times 1}$ coefficient estimates

	$[a, b]$	$(0.005, 0.65)$	$(0.01, 0.35)$	$(0.01, 0.25)$
$\tilde{\boldsymbol{\beta}}_{\mathrm{one-way}}$	β_2	0.9159(0.0654,0.0969)	0.6733(0.0678,0.1005)	0.4326(0.0715,0.1060)
	β_5	0.9648(0.0659,0.0977)	0.7283(0.0682,0.1011)	0.4341(0.0729,0.1080)
$\tilde{\boldsymbol{\beta}}_{\mathrm{two-way}}$	β_2	0.0955(0.0921,0.1365)	0.0875(0.0913,0.1354)	0.0845(0.0912,0.1353)
	β_5	0.1255(0.0976,0.1448)	0.1170(0.0950,0.1409)	0.1151(0.0948,0.1406)

* For each cell, the value outside the brackets represents SD, while the first and second numbers inside the brackets represent $\mathrm{ESE_m}$ and $\mathrm{ESE_o}$, respectively.

Table C.2: SDs and ESEs for non-zero $\boldsymbol{\beta}_{15\times 1}$ coefficient estimates

	b	10	5	2
$\tilde{\boldsymbol{\beta}}_{\mathrm{one-way}}$	β_2	0.4444(0.0714,0.1059)	0.6733(0.0678,0.1005)	1.2666(0.0643,0.0954)
	β_5	0.4609(0.0725,0.1074)	0.7283(0.0682,0.1011)	1.3036(0.0647,0.0959)
$\tilde{\boldsymbol{\beta}}_{\mathrm{two-way}}$	β_2	0.1741(0.0877,0.1300)	0.0875(0.0913,0.1354)	0.0984(0.0913,0.1354)
	β_5	0.1494(0.0915,0.1357)	0.1170(0.0950,0.1409)	0.1027(0.0962,0.1426)

* For each cell, the value outside the brackets represents SD, while the first and second numbers inside the brackets represent $\mathrm{ESE_m}$ and $\mathrm{ESE_o}$, respectively.

Table C.3: SDs and ESEs for non-zero $\boldsymbol{\beta}_{50\times1}$ coefficient estimates

$p = 15, \rho$		0	0.2	0.5
β_2	$\tilde{\boldsymbol{\beta}}$	0.0845(0.0963,0.1428)	0.0913(0.0935,0.1386)	0.0900(0.0911,0.1351)
	$\tilde{\boldsymbol{\beta}}_{\lambda=0}$	0.0840(0.0970,0.1438)	0.0899(0.0959,0.1422)	0.1013(0.0977,0.1448)
β_3	$\tilde{\boldsymbol{\beta}}$	0.1406(0.1278,0.1895)	0.1373(0.1269,0.1881)	0.1435(0.1276,0.1891)
	$\tilde{\boldsymbol{\beta}}_{\lambda=0}$	0.1331(0.1292,0.1916)	0.1441(0.1286,0.1907)	0.1326(0.1327,0.1968)
β_5	$\tilde{\boldsymbol{\beta}}$	0.1343(0.1098,0.1628)	0.1219(0.1085,0.1608)	0.1225(0.1107,0.1641)
	$\tilde{\boldsymbol{\beta}}_{\lambda=0}$	0.1269(0.1110,0.1645)	0.1268(0.1111,0.1647)	0.1518(0.1181,0.1751)
$p = 50, \rho$		0	0.2	0.5
β_2	$\tilde{\boldsymbol{\beta}}$	0.0991(0.0962,0.1427)	0.0919(0.0930,0.1379)	0.1039(0.0892,0.1322)
	$\tilde{\boldsymbol{\beta}}_{\lambda=0}$	0.3056(0.1505,0.2231)	0.2081(0.1163,0.1725)	0.1694(0.1057,0.1567)
β_3	$\tilde{\boldsymbol{\beta}}$	0.1328(0.1293,0.1916)	0.1454(0.1255,0.1860)	0.1526(0.1254,0.1858)
	$\tilde{\boldsymbol{\beta}}_{\lambda=0}$	0.6621(0.2486,0.3686)	0.3737(0.1643,0.2436)	0.2325(0.1435,0.2128)
β_5	$\tilde{\boldsymbol{\beta}}$	0.1460(0.1099,0.1630)	0.1293(0.1076,0.1596)	0.1398(0.1093,0.1620)
	$\tilde{\boldsymbol{\beta}}_{\lambda=0}$	0.4427(0.1981,0.2937)	0.3137(0.1381,0.2047)	0.2385(0.1308,0.1940)

* For each cell, the value outside the brackets represents SD, while the first and second numbers inside the brackets represent $\mathrm{ESE_m}$ and $\mathrm{ESE_o}$, respectively.

Table C.4: SDs and ESEs for non-zero $\boldsymbol{\beta}_{p\times1}$ coefficient estimates

$p = 10,\ n_{cv}$	10	20	30	40	50
β_2	0.0442(0.0434,0.0643)	0.0646(0.0435,0.0645)	0.0401(0.0435,0.0646)	0.0393(0.0435,0.0645)	0.0396(0.0435,0.0645)
β_3	0.0660(0.0498,0.0739)	0.0739(0.0499,0.0740)	0.0630(0.0499,0.0740)	0.0617(0.0500,0.0741)	0.0622(0.0500,0.0742)
β_5	0.0578(0.0495,0.0734)	0.0733(0.0494,0.0732)	0.0529(0.0494,0.0732)	0.0538(0.0496,0.0735)	0.0553(0.0496,0.0735)
$p = 100,\ n_{cv}$	10	20	30	40	50
β_2	0.1249(0.0414,0.0613)	0.0950(0.0417,0.0619)	0.0905(0.0418,0.0620)	0.0703(0.0431,0.0638)	0.0858(0.0410,0.0608)
β_3	0.1239(0.0481,0.0713)	0.1206(0.0482,0.0715)	0.1388(0.0480,0.0712)	0.0787(0.0446,0.0661)	0.2028(0.0458,0.0679)
β_5	0.1515(0.0469,0.0695)	0.1308(0.0470,0.0696)	0.1501(0.0469,0.0695)	0.0742(0.0447,0.0662)	0.2073(0.0445,0.0660)

* For each cell, the value outside the brackets represents SD, while the first and second numbers inside the brackets represent $\mathrm{ESE_m}$ and $\mathrm{ESE_o}$, respectively.

Table C.5: SDs and ESEs for non-zero $\boldsymbol{\beta}_{100\times 1}$ coefficient estimates

$b = 2,\ n_{cv}$	10	20	30	40	50
β_2	0.4290(0.1279,0.1896)	0.3313(0.1310,0.1942)	0.3173(0.1313,0.1947)	0.4536(0.1270,0.1883)	0.4748(0.1258,0.1865)
β_3	0.3876(0.1486,0.2202)	0.3953(0.1490,0.2209)	0.3742(0.1495,0.2216)	1.0245(0.1259,0.1867)	0.7394(0.1346,0.1996)
β_5	0.4918(0.1432,0.2122)	0.4880(0.1459,0.2163)	0.5303(0.1460,0.2165)	1.1041(0.1164,0.1726)	1.1553(0.1271,0.1885)
$b = 20,\ n_{cv}$	10	20	30	40	50
β_2	0.1249(0.0414,0.0613)	0.0950(0.0417,0.0619)	0.0905(0.0418,0.0620)	0.0703(0.0431,0.0638)	0.0858(0.0410,0.0608)
β_3	0.1239(0.0481,0.0713)	0.1206(0.0482,0.0715)	0.1388(0.0480,0.0712)	0.0787(0.0446,0.0661)	0.2028(0.0458,0.0679)
β_5	0.1515(0.0469,0.0695)	0.1308(0.0470,0.0696)	0.1501(0.0469,0.0695)	0.0742(0.0447,0.0662)	0.2073(0.0445,0.0660)

* For each cell, the value outside the brackets represents SD, while the first and second numbers inside the brackets represent $\mathrm{ESE_m}$ and $\mathrm{ESE_o}$, respectively.

Table C.6: SDs and ESEs for non-zero $\boldsymbol{\beta}$ coefficient estimates

$p = 15$		β_2	β_3	β_6	β_7	β_{11}	β_{13}
$\rho = 0$	$\tilde{\boldsymbol{\beta}}_\lambda$	0.0767(0.0628,0.0931)	0.0742(0.0636,0.0943)	0.0938(0.0675,0.1000)	0.0676(0.0627,0.0929)	0.0937(0.0636,0.0942)	0.1096(0.0676,0.1002)
	$\tilde{\boldsymbol{\beta}}_{\lambda*}$	0.0857(0.0631,0.0936)	0.0922(0.0635,0.0941)	0.1042(0.0684,0.1014)	0.0850(0.0633,0.0939)	0.0752(0.0639,0.0947)	0.0831(0.0680,0.1008)
	C1:$\hat{\boldsymbol{\beta}}^*$	0.0645(0.0629,0.0932)	0.0766(0.0637,0.0945)	0.0742(0.0679,0.1006)	0.0629(0.0629,0.0932)	0.0830(0.0637,0.0944)	0.0738(0.0683,0.1013)
	C2:$\hat{\boldsymbol{\beta}}^*$	0.0622(0.0626,0.0928)	0.0695(0.0635,0.0941)	0.0760(0.0675,0.1000)	0.0620(0.0626,0.0929)	0.0798(0.0635,0.0941)	0.0735(0.0681,0.1009)
$\rho = 0.2$	$\tilde{\boldsymbol{\beta}}_\lambda$	0.0774(0.0637,0.0945)	0.0781(0.0646,0.0958)	0.0967(0.0688,0.1019)	0.0830(0.0640,0.0948)	0.0846(0.0640,0.0948)	0.0870(0.0679,0.1007)
	$\tilde{\boldsymbol{\beta}}_{\lambda*}$	0.0934(0.0641,0.0950)	0.0803(0.0646,0.0958)	0.0797(0.0693,0.1028)	0.0740(0.0644,0.0955)	0.0776(0.0641,0.0950)	0.0986(0.0682,0.1011)
	C1:$\hat{\boldsymbol{\beta}}^*$	0.0934(0.0641,0.0950)	0.0803(0.0646,0.0958)	0.0797(0.0693,0.1028)	0.0740(0.0644,0.0955)	0.0776(0.0641,0.0950)	0.0986(0.0682,0.1011)
	C2:$\hat{\boldsymbol{\beta}}^*$	0.0709(0.0634,0.0941)	0.0686(0.0644,0.0955)	0.0834(0.0686,0.1017)	0.0593(0.0636,0.0943)	0.0706(0.0635,0.0941)	0.0812(0.0677,0.1003)
$\rho = 0.5$	$\tilde{\boldsymbol{\beta}}_\lambda$	0.0667(0.0694,0.1029)	0.0946(0.0721,0.1069)	0.1198(0.0756,0.1121)	0.0875(0.0707,0.1049)	0.1002(0.0690,0.1023)	0.1069(0.0715,0.1060)
	$\tilde{\boldsymbol{\beta}}_{\lambda*}$	0.0889(0.0697,0.1033)	0.0910(0.0717,0.1064)	0.0976(0.0757,0.1123)	0.0729(0.0706,0.1047)	0.0833(0.0665,0.0986)	0.0853(0.0706,0.1046)
	C1:$\hat{\boldsymbol{\beta}}^*$	0.0601(0.0697,0.1033)	0.0773(0.0707,0.1049)	0.0819(0.0742,0.1100)	0.0551(0.0700,0.1037)	0.0695(0.0646,0.0958)	0.0845(0.0692,0.1026)
	C2:$\hat{\boldsymbol{\beta}}^*$	0.0622(0.0689,0.1022)	0.0798(0.0703,0.1042)	0.0818(0.0737,0.1093)	0.0520(0.0695,0.1030)	0.0685(0.0640,0.0949)	0.0808(0.0684,0.1014)
$p = 100$		β_2	β_3	β_6	β_7	β_{11}	β_{13}
$\rho = 0$	$\tilde{\boldsymbol{\beta}}_\lambda$	0.1051(0.0611,0.0906)	0.1095(0.0622,0.0922)	0.1186(0.0658,0.0975)	0.1105(0.0611,0.0905)	0.1280(0.0622,0.0922)	0.1260(0.0658,0.0975)
	$\tilde{\boldsymbol{\beta}}_{\lambda*}$	0.1100(0.0618,0.0916)	0.1043(0.0623,0.0923)	0.1193(0.0662,0.0982)	0.1050(0.0614,0.0911)	0.0950(0.0624,0.0925)	0.1074(0.0661,0.0979)
	C1:$\hat{\boldsymbol{\beta}}^*$	0.0613(0.0626,0.0928)	0.0618(0.0635,0.0942)	0.0795(0.0674,0.1000)	0.0560(0.0629,0.0932)	0.0739(0.0634,0.0940)	0.0677(0.0682,0.1011)
	C2:$\hat{\boldsymbol{\beta}}^*$	0.0636(0.0626,0.0927)	0.0618(0.0635,0.0941)	0.0794(0.0673,0.0997)	0.0559(0.0628,0.0931)	0.0752(0.0634,0.0940)	0.0680(0.0682,0.1011)
$\rho = 0.2$	$\tilde{\boldsymbol{\beta}}_\lambda$	0.1510(0.0606,0.0899)	0.1626(0.0613,0.0909)	0.1354(0.0654,0.0970)	0.1283(0.0615,0.0911)	0.1538(0.0607,0.0900)	0.1633(0.0639,0.0947)
	$\tilde{\boldsymbol{\beta}}_{\lambda*}$	0.1496(0.0614,0.0911)	0.1288(0.0624,0.0925)	0.1235(0.0667,0.0990)	0.1155(0.0621,0.0921)	0.1167(0.0615,0.0912)	0.1468(0.0643,0.0953)
	C1:$\hat{\boldsymbol{\beta}}^*$	0.0678(0.0637,0.0944)	0.0676(0.0648,0.0960)	0.0742(0.0687,0.1019)	0.0537(0.0639,0.0947)	0.0645(0.0636,0.0943)	0.0740(0.0680,0.1009)
	C2:$\hat{\boldsymbol{\beta}}^*$	0.0683(0.0635,0.0941)	0.0700(0.0647,0.0960)	0.0793(0.0687,0.1018)	0.0537(0.0637,0.0945)	0.0645(0.0635,0.0942)	0.0782(0.0679,0.1007)
$\rho = 0.5$	$\tilde{\boldsymbol{\beta}}_\lambda$	0.3055(0.0624,0.0926)	0.3000(0.0625,0.0926)	0.2209(0.0682,0.1011)	0.1886(0.0633,0.0938)	0.2090(0.0589,0.0873)	0.1957(0.0622,0.0923)
	$\tilde{\boldsymbol{\beta}}_{\lambda*}$	0.2770(0.0613,0.0909)	0.2841(0.0616,0.0914)	0.1861(0.0687,0.1019)	0.1707(0.0639,0.0947)	0.1948(0.0588,0.0871)	0.2249(0.0614,0.0910)
	C1:$\hat{\boldsymbol{\beta}}^*$	0.0601(0.0697,0.1033)	0.0773(0.0707,0.1048)	0.0819(0.0742,0.1099)	0.0558(0.0700,0.1037)	0.0695(0.0645,0.0957)	0.0829(0.0691,0.1025)
	C2:$\hat{\boldsymbol{\beta}}^*$	0.0601(0.0697,0.1033)	0.0772(0.0707,0.1048)	0.0819(0.0742,0.1099)	0.0558(0.0700,0.1037)	0.0695(0.0645,0.0957)	0.0829(0.0691,0.1025)

* For each cell, the value outside the brackets represents SD, while the first and second numbers inside the brackets represent $\mathrm{ESE_m}$ and $\mathrm{ESE_o}$, respectively.

Table C.7: SDs and ESEs for non-zero $\boldsymbol{\beta}_{p\times 1}$ coefficient estimates

$p = 15$		β_2	β_3	β_6	β_7	β_{11}	β_{13}
	$\tilde{\boldsymbol{\beta}}_\lambda$	0.1037(0.0906,0.1344)	0.0863(0.0927,0.1375)	0.1075(0.0977,0.1449)	0.0886(0.0915,0.1357)	0.1156(0.0909,0.1347)	0.1033(0.0975,0.1445)
$b = 4$	$\tilde{\boldsymbol{\beta}}_{\lambda*}$	0.0950(0.0908,0.1347)	0.0845(0.0925,0.1372)	0.1028(0.0978,0.1449)	0.0824(0.0915,0.1356)	0.1161(0.0905,0.1342)	0.0983(0.0972,0.1441)
	$\hat{\boldsymbol{\beta}}^*$	0.0989(0.0912,0.1352)	0.1047(0.0928,0.1375)	0.0975(0.0984,0.1458)	0.0822(0.0920,0.1363)	0.1089(0.0905,0.1342)	0.1006(0.0973,0.1443)
	$\tilde{\boldsymbol{\beta}}_\lambda$	0.0742(0.0644,0.0955)	0.0686(0.0654,0.0970)	0.0911(0.0690,0.1022)	0.0610(0.0648,0.0960)	0.0737(0.0633,0.0938)	0.0715(0.0679,0.1007)
$b = 8$	$\tilde{\boldsymbol{\beta}}_{\lambda*}$	0.0732(0.0644,0.0955)	0.0659(0.0652,0.0967)	0.0909(0.0690,0.1022)	0.0650(0.0646,0.0957)	0.0726(0.0630,0.0935)	0.0716(0.0678,0.1005)
	$\hat{\boldsymbol{\beta}}^*$	0.0775(0.0642,0.0951)	0.0731(0.0652,0.0967)	0.0838(0.0694,0.1029)	0.0615(0.0643,0.0954)	0.0771(0.0629,0.0933)	0.0737(0.0679,0.1007)
$p = 150$		β_2	β_3	β_6	β_7	β_{11}	β_{13}
	$\tilde{\boldsymbol{\beta}}_\lambda$	0.1206(0.0897,0.1330)	0.1061(0.0902,0.1338)	0.1177(0.0957,0.1419)	0.1137(0.0894,0.1326)	0.1238(0.0877,0.1300)	0.1239(0.0949,0.1406)
$b = 4$	$\tilde{\boldsymbol{\beta}}_{\lambda*}$	0.1210(0.0903,0.1338)	0.0981(0.0908,0.1347)	0.1167(0.0964,0.1429)	0.1139(0.0894,0.1325)	0.1248(0.0876,0.1298)	0.1230(0.0951,0.1409)
	$\hat{\boldsymbol{\beta}}^*$	0.1169(0.0929,0.1377)	0.1108(0.0942,0.1396)	0.1174(0.0999,0.1482)	0.0968(0.0928,0.1376)	0.1089(0.0916,0.1359)	0.0994(0.0988,0.1464)
	$\tilde{\boldsymbol{\beta}}_\lambda$	0.0889(0.0632,0.0938)	0.0812(0.0639,0.0947)	0.0973(0.0677,0.1004)	0.0867(0.0632,0.0938)	0.0891(0.0617,0.0914)	0.0970(0.0664,0.0984)
$b = 8$	$\tilde{\boldsymbol{\beta}}_{\lambda*}$	0.0949(0.0637,0.0944)	0.0863(0.0645,0.0956)	0.0974(0.0680,0.1008)	0.0870(0.0637,0.0945)	0.0948(0.0619,0.0918)	0.0971(0.0666,0.0988)
	$\hat{\boldsymbol{\beta}}^*$	0.0694(0.0639,0.0947)	0.0681(0.0649,0.0963)	0.0797(0.0690,0.1024)	0.0570(0.0641,0.0950)	0.0724(0.0627,0.0930)	0.0702(0.0674,0.0999)

* For each cell, the value outside the brackets represents SD, while the first and second numbers inside the brackets represent $\mathrm{ESE_m}$ and $\mathrm{ESE_o}$, respectively.

Table C.8: SDs and ESEs for non-zero $\boldsymbol{\beta}_{150\times 1}$ coefficient estimates

		β_2	β_3	β_6	β_7	β_{11}	β_{13}
	$\tilde{\boldsymbol{\beta}}_\lambda$	0.0936(0.0626,0.0928)	0.0796(0.0629,0.0933)	0.0793(0.0669,0.0991)	0.1195(0.0625,0.0927)	0.0875(0.0623,0.0923)	0.0995(0.0662,0.0981)
$b = 8$	$\tilde{\boldsymbol{\beta}}_{\lambda*}$	0.0962(0.0627,0.0929)	0.0818(0.0629,0.0933)	0.0803(0.0670,0.0993)	0.1193(0.0624,0.0925)	0.0887(0.0625,0.0926)	0.0992(0.0664,0.0985)
	$\hat{\boldsymbol{\beta}}^*$	0.0572(0.0643,0.0953)	0.0639(0.0647,0.0959)	0.0653(0.0689,0.1022)	0.0772(0.0639,0.0948)	0.0549(0.0633,0.0939)	0.0704(0.0678,0.1005)
	$\tilde{\boldsymbol{\beta}}_\lambda$	0.0483(0.0396,0.0588)	0.0438(0.0401,0.0594)	0.0466(0.0425,0.0630)	0.0619(0.0394,0.0584)	0.0520(0.0393,0.0583)	0.0577(0.0418,0.0620)
$b = 20$	$\tilde{\boldsymbol{\beta}}_{\lambda*}$	0.0519(0.0396,0.0588)	0.0442(0.0401,0.0595)	0.0491(0.0425,0.0631)	0.0575(0.0395,0.0586)	0.0579(0.0393,0.0583)	0.0575(0.0419,0.0621)
	$\hat{\boldsymbol{\beta}}^*$	0.0354(0.0402,0.0596)	0.0418(0.0408,0.0604)	0.0370(0.0432,0.0641)	0.0379(0.0401,0.0594)	0.0340(0.0398,0.0590)	0.0449(0.0425,0.0630)

* For each cell, the value outside the brackets represents SD, while the first and second numbers inside the brackets represent $\mathrm{ESE_m}$ and $\mathrm{ESE_o}$, respectively.

Table C.9: SDs and ESEs for non-zero $\boldsymbol{\beta}_{1900\times1}$ coefficient estimates

ρ	$n_b = 300$	$\tilde{\boldsymbol{\beta}}_{\text{SIS}-\text{SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\text{SIS}_{\text{V1}}-\text{SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\text{ISIS}-\text{SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\text{ISIS}_{\text{V1}}-\text{SCAD}}$
0.5	β_2	1.1861(0,0)	1.1861(0,0)	0.1032(0.0705,0.1045)	0.0900(0.0709,0.1052)
	β_3	1.4708(0.0117,0.0173)	0.8028(0.0531,0.0787)	0.1341(0.0741,0.1099)	0.1072(0.0744,0.1104)
	β_6	1.1861(0,0)	1.1861(0,0)	0.0964(0.0712,0.1055)	0.0874(0.0714,0.1058)
	β_7	0.7399(0.0539,0.0799)	0.7129(0.0542,0.0803)	0.1058(0.0744,0.1103)	0.0841(0.0746,0.1106)
0.2	β_2	1.1861(0,0)	0.1147(0.0649,0.0963)	0.0824(0.0684,0.1015)	0.0872(0.0684,0.1014)
	β_3	0.2831(0.0629,0.0933)	0.1175(0.0683,0.1013)	0.1056(0.0728,0.1079)	0.1025(0.0727,0.1078)
	β_6	1.1861(0,0)	0.1207(0.0650,0.0964)	0.0860(0.0687,0.1018)	0.0846(0.0687,0.1019)
	β_7	0.2788(0.0630,0.0935)	0.1313(0.0679,0.1006)	0.0953(0.0723,0.1071)	0.0913(0.0723,0.1071)
0.0	β_2	0.0982(0.0667,0.0989)	0.0882(0.0672,0.0996)	0.0943(0.0703,0.1042)	0.0915(0.0703,0.1043)
	β_3	0.0933(0.0700,0.1038)	0.0818(0.0708,0.1049)	0.0895(0.0746,0.1105)	0.0971(0.0746,0.1106)
	β_6	0.1087(0.0662,0.0981)	0.0882(0.0672,0.0997)	0.1074(0.0705,0.1045)	0.0989(0.0705,0.1045)
	β_7	0.0866(0.0697,0.1034)	0.0813(0.0704,0.1044)	0.0946(0.0744,0.1103)	0.0939(0.0743,0.1102)
ρ	$n_b = 600$	$\tilde{\boldsymbol{\beta}}_{\text{SIS}-\text{SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\text{SIS}_{\text{V1}}-\text{SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\text{ISIS}-\text{SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\text{ISIS}_{\text{V1}}-\text{SCAD}}$
0.5	β_2	1.1861(0,0)	0.2746(0.0635,0.0941)	0.0846(0.0707,0.1048)	0.0828(0.0708,0.1050)
	β_3	0.6157(0.0555,0.0823)	0.3696(0.0655,0.0971)	0.1097(0.0745,0.1105)	0.1038(0.0746,0.1106)
	β_6	1.1861(0,0)	1.1861(0,0)	0.0823(0.0718,0.1065)	0.0795(0.0718,0.1064)
	β_7	0.6813(0.0560,0.0831)	0.6636(0.0568,0.0841)	0.1084(0.0749,0.1111)	0.1084(0.0750,0.1111)
0.2	β_2	0.0859(0.0659,0.0978)	0.0871(0.0659,0.0978)	0.1008(0.0681,0.1009)	0.1008(0.0681,0.1010)
	β_3	0.1085(0.0694,0.1029)	0.1068(0.0694,0.1029)	0.1139(0.0717,0.1064)	0.1132(0.0718,0.1064)
	β_6	0.0769(0.0662,0.0981)	0.0756(0.0662,0.0982)	0.0975(0.0687,0.1019)	0.0930(0.0687,0.1019)
	β_7	0.0879(0.0696,0.1032)	0.0851(0.0697,0.1033)	0.1014(0.0722,0.1070)	0.0961(0.0721,0.1070)
0.0	β_2	0.0767(0.0676,0.1002)	0.0758(0.0676,0.1002)	0.0926(0.0708,0.1050)	0.0899(0.0706,0.1047)
	β_3	0.0768(0.0708,0.1049)	0.0768(0.0707,0.1049)	0.1067(0.0742,0.1100)	0.1037(0.0742,0.1100)
	β_6	0.0844(0.0672,0.0996)	0.0840(0.0672,0.0996)	0.0976(0.0703,0.1042)	0.0933(0.0703,0.1042)
	β_7	0.0849(0.0713,0.1057)	0.0844(0.0713,0.1057)	0.1147(0.0741,0.1099)	0.1147(0.0740,0.1097)

* For each cell, the value outside the brackets represents SD, while the first and second numbers inside the brackets represent ESE_m and ESE_o, respectively.

Table C.10: SDs and ESEs for non-zero $\boldsymbol{\beta}_{3000\times1}$ coefficient estimates

ρ		$\tilde{\boldsymbol{\beta}}_{\text{ISIS}-\text{SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\text{ISIS}_{\text{V1}}-\text{SCAD}}$	$\hat{\boldsymbol{\beta}}_{\text{ISIS}-\text{SCAD}}$
0.5	β_2	0.0631(0.0595,0.0882)	0.0646(0.0595,0.0882)	0.0854(0.0576,0.0854)
	β_3	0.0659(0.0625,0.0929)	0.0629(0.0627,0.0930)	0.0896(0.0605,0.0896)
	β_6	0.0674(0.0600,0.0889)	0.0688(0.0600,0.0889)	0.0859(0.0582,0.0863)
	β_7	0.0665(0.0626,0.0928)	0.0672(0.0626,0.0928)	0.0894(0.0604,0.0896)
0.2	β_2	0.0640(0.0574,0.0851)	0.0640(0.0574,0.0851)	0.0807(0.0549,0.0814)
	β_3	0.0773(0.0610,0.0905)	0.0776(0.0610,0.0904)	0.0857(0.0577,0.0856)
	β_6	0.0608(0.0577,0.0856)	0.0608(0.0577,0.0856)	0.0815(0.0550,0.0816)
	β_7	0.0686(0.0608,0.0902)	0.0686(0.0608,0.0902)	0.0856(0.0579,0.0859)
0	β_2	0.0574(0.0585,0.0867)	0.0574(0.0584,0.0867)	0.0843(0.0572,0.0848)
	β_3	0.0714(0.0615,0.0912)	0.0714(0.0615,0.0912)	0.0894(0.0602,0.0892)
	β_6	0.0690(0.0585,0.0867)	0.0658(0.0585,0.0867)	0.0846(0.0574,0.0850)
	β_7	0.0600(0.0617,0.0914)	0.0600(0.0617,0.0914)	0.0896(0.0604,0.0895)

* For each cell, the value outside the brackets represents SD, while the first and second numbers inside the brackets represent ESE_m and ESE_o, respectively.

* Each training dataset consists of $N_b = 2500$ data. In the renewable estimation experiments, each data stream batch has a size of $n_b = 500$.

Table C.11: SDs and ESEs for non-zero $\boldsymbol{\beta}_{3000\times 1}$ coefficient estimates

n_b		$\tilde{\boldsymbol{\beta}}_{\mathrm{ISIS-SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\mathrm{ISIS_{V1}-SCAD}}$	$\hat{\boldsymbol{\beta}}_{\mathrm{ISIS-SCAD}}$
400	β_2	0.0629(0.0672,0.0996)	0.0635(0.0672,0.0996)	0.1009(0.0681,0.1009)
	β_3	0.0895(0.0707,0.1047)	0.0860(0.0707,0.1047)	0.1070(0.0720,0.1068)
	β_6	0.0823(0.0676,0.1003)	0.0823(0.0677,0.1004)	0.1023(0.0687,0.1019)
	β_7	0.0798(0.0709,0.1050)	0.0805(0.0708,0.1050)	0.1076(0.0724,0.1073)
500	β_2	0.0729(0.0667,0.0989)	0.0741(0.0667,0.0989)	0.1009(0.0681,0.1009)
	β_3	0.0787(0.0703,0.1042)	0.0742(0.0705,0.1045)	0.1070(0.0720,0.1068)
	β_6	0.0740(0.0676,0.1002)	0.0721(0.0676,0.1002)	0.1023(0.0687,0.1019)
	β_7	0.0787(0.0703,0.1042)	0.0828(0.0703,0.1042)	0.1076(0.0724,0.1073)
1000	β_2	0.0743(0.0668,0.0991)	0.0741(0.0669,0.0991)	0.1009(0.0681,0.1009)
	β_3	0.0809(0.0703,0.1042)	0.0765(0.0702,0.1040)	0.1070(0.0720,0.1068)
	β_6	0.0808(0.0675,0.1001)	0.0802(0.0674,0.0999)	0.1023(0.0687,0.1019)
	β_7	0.0805(0.0705,0.1045)	0.0835(0.0701,0.1039)	0.1076(0.0724,0.1073)

* For each cell, the value outside the brackets represents SD, while the first and second numbers inside the brackets represent $\mathrm{ESE_m}$ and $\mathrm{ESE_o}$, respectively.

* Each training dataset consists of $N_b = 2000$ data.

Table C.12: SDs and ESEs for non-zero $\boldsymbol{\beta}_{3000\times 1}$ coefficient estimates

N_b		$\tilde{\boldsymbol{\beta}}_{\mathrm{ISIS-SCAD}}$	$\tilde{\boldsymbol{\beta}}_{\mathrm{ISIS_{V1}-SCAD}}$	$\hat{\boldsymbol{\beta}}_{\mathrm{ISIS-SCAD}}$
1000	β_2	0.1164(0.0971,0.1439)	0.1197(0.0973,0.1442)	0.1346(0.0925,0.1371)
	β_3	0.1476(0.1024,0.1517)	0.1476(0.1024,0.1517)	0.1407(0.0964,0.1430)
	β_6	0.1020(0.0982,0.1456)	0.1114(0.0983,0.1457)	0.1362(0.0930,0.1378)
	β_7	0.1309(0.1023,0.1517)	0.1244(0.1022,0.1515)	0.1411(0.0964,0.1429)
1500	β_2	0.0846(0.0778,0.1154)	0.0848(0.0779,0.1155)	0.1086(0.0743,0.1102)
	β_3	0.0984(0.0820,0.1216)	0.0949(0.0820,0.1216)	0.1145(0.0783,0.1161)
	β_6	0.0793(0.0786,0.1166)	0.0819(0.0787,0.1166)	0.1101(0.0752,0.1115)
	β_7	0.0804(0.0821,0.1217)	0.0812(0.0820,0.1215)	0.1146(0.0778,0.1154)
2000	β_2	0.0729(0.0667,0.0989)	0.0741(0.0667,0.0989)	0.1009(0.0681,0.1009)
	β_3	0.0787(0.0703,0.1042)	0.0742(0.0705,0.1045)	0.1070(0.0720,0.1068
	β_6	0.0740(0.0676,0.1002)	0.0721(0.0676,0.1002)	0.1023(0.0687,0.1019)
	β_7	0.0787(0.0703,0.1042)	0.0828(0.0703,0.1042)	0.1076(0.0724,0.1073)
2500	β_2	0.0631(0.0595,0.0882)	0.0646(0.0595,0.0882)	0.0854(0.0576,0.0854)
	β_3	0.0659(0.0625,0.0929)	0.0629(0.0627,0.0930)	0.0896(0.0605,0.0896)
	β_6	0.0674(0.0600,0.0889)	0.0688(0.0600,0.0930)	0.0859(0.0582,0.0863)
	β_7	0.0665(0.0626,0.0928)	0.0672(0.0626,0.0928)	0.0894(0.0604,0.0896)

* For each cell, the value outside the brackets represents SD, while the first and second numbers inside the brackets represent $\mathrm{ESE_m}$ and $\mathrm{ESE_o}$, respectively.

* In the renewable estimation experiments, each data stream batch has a size of $n_b = 500$.

C.2 Real data analysis

Table C.13: Comparison of misclassification rates (%) of various classifiers

Testing Batch: $\mathcal{B}_{b+1}$																	
$(b+1)$	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Batch size	351	424	398	443	387	418	409	407	461	367	384	416	363	386	380	374	396
$C_{\hat{\beta}^*_{b,\lambda=0}}$	5.4131	2.8302	2.5126	4.7404	2.5840	2.1531	3.6675	1.2285	3.2538	2.7248	4.1667	0.7212	2.4793	2.8497	2.8947	2.1390	1.2626
$C_{\tilde{\beta}_{b,\lambda=0}}$	5.4131	2.8302	2.5126	4.7404	2.5840	2.1531	3.6675	1.2285	3.2538	2.7248	4.1667	0.7212	2.4793	2.8497	2.8947	2.1390	1.2626
$C_{\hat{\beta}^*_b}$	5.4131	2.8302	2.5126	4.7404	2.5840	2.1531	3.6675	1.2285	3.2538	2.7248	4.1667	0.7212	2.4793	2.8497	2.8947	2.1390	1.2626
$C_{\tilde{\beta}_{b,\lambda}}$	5.4131	2.8302	2.5126	4.7404	2.5840	2.1531	3.6675	1.2285	3.2538	2.7248	4.1667	0.7212	2.4793	2.8497	2.8947	2.1390	1.2626
$C_{\tilde{\beta}_{b,\lambda*}}$	5.4131	2.8302	2.5126	4.7404	2.5840	2.1531	3.6675	1.2285	3.2538	2.7248	4.1667	0.7212	2.4793	2.8497	2.8947	2.1390	1.2626
Testing Batch: $\mathcal{B}_{b+1}$																	
$(b+1)$	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35
Batch size	351	364	340	295	251	281	358	312	318	319	266	206	307	348	353	353	297
$C_{\hat{\beta}^*_{b,\lambda=0}}$	4.5584	2.7473	4.1176	4.0678	2.7888	3.9146	0.5587	2.2436	2.5157	6.5831	3.0075	3.3981	1.3029	2.5862	1.1331	5.0992	1.3468
$C_{\tilde{\beta}_{b,\lambda=0}}$	4.5584	2.7473	4.1176	4.0678	2.7888	3.9146	0.5587	2.2436	2.5157	6.5831	3.0075	3.3981	1.3029	2.5862	1.1331	5.0992	1.3468
$C_{\hat{\beta}^*_b}$	4.5584	2.7473	4.1176	4.0678	2.7888	3.9146	0.5587	2.2436	2.5157	6.5831	3.0075	3.3981	1.3029	2.5862	1.1331	5.0992	1.3468
$C_{\tilde{\beta}_{b,\lambda}}$	4.5584	2.7473	4.1176	4.0678	2.7888	3.9146	0.5587	2.2436	2.5157	6.5831	3.0075	3.3981	1.3029	2.5862	1.1331	5.0992	1.3468
$C_{\tilde{\beta}_{b,\lambda*}}$	4.5584	2.7473	4.1176	4.0678	2.7888	3.9146	0.5587	2.2436	2.5157	6.5831	3.0075	3.3981	1.3029	2.5862	1.1331	5.0992	1.3468
Testing Batch: $\mathcal{B}_{b+1}$																	
$(b+1)$	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52
Batch size	286	371	281	309	231	219	234	216	191	181	259	216	223	260	220	242	242
$C_{\hat{\beta}^*_{b,\lambda=0}}$	3.4965	1.0782	1.4235	2.5890	4.3290	6.8493	4.7009	4.6296	3.6649	8.2873	3.8610	6.4815	2.6906	1.9231	0.9091	2.8926	1.6529
$C_{\tilde{\beta}_{b,\lambda=0}}$	3.4965	1.0782	1.4235	2.5890	4.3290	6.8493	4.7009	4.6296	3.6649	8.2873	3.8610	6.4815	2.6906	1.9231	0.9091	2.8926	1.6529
$C_{\hat{\beta}^*_b}$	3.4965	1.0782	1.4235	2.5890	4.3290	6.8493	4.7009	4.6296	3.6649	8.2873	3.8610	6.4815	2.6906	1.9231	0.9091	2.8926	1.6529
$C_{\tilde{\beta}_{b,\lambda}}$	3.4965	1.0782	1.4235	2.5890	4.3290	6.8493	4.7009	4.6296	3.6649	8.2873	3.8610	6.4815	2.6906	1.9231	0.9091	2.8926	1.6529
$C_{\tilde{\beta}_{b,\lambda*}}$	3.4965	1.0782	1.4235	2.5890	4.3290	6.8493	4.7009	4.6296	3.6649	8.2873	3.8610	6.4815	2.6906	1.9231	0.9091	2.8926	1.6529
Testing Batch: $\mathcal{B}_{b+1}$																	
$(b+1)$	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69
Batch size	256	225	241	269	237	236	210	190	203	178	216	206	218	209	210	207	206
$C_{\hat{\beta}^*_{b,\lambda=0}}$	2.3438	3.5556	3.3195	1.4870	5.9072	4.2373	6.1905	4.7368	3.4483	3.9326	3.2407	3.3981	3.6697	5.7416	4.2857	3.3816	2.9126
$C_{\tilde{\beta}_{b,\lambda=0}}$	2.3438	3.5556	3.3195	1.4870	5.9072	4.2373	6.1905	4.7368	3.4483	3.9326	3.2407	3.3981	3.6697	5.7416	4.2857	3.3816	2.9126
$C_{\hat{\beta}^*_b}$	2.3438	3.5556	3.3195	1.4870	5.9072	4.2373	6.1905	4.7368	3.4483	3.9326	3.2407	3.3981	3.6697	5.7416	4.2857	3.3816	2.9126
$C_{\tilde{\beta}_{b,\lambda}}$	2.3438	3.5556	3.3195	1.4870	5.9072	4.2373	6.1905	4.7368	3.4483	3.9326	3.2407	3.3981	3.6697	5.7416	4.2857	3.3816	2.9126
$C_{\tilde{\beta}_{b,\lambda*}}$	2.3438	3.5556	3.3195	1.4870	5.9072	4.2373	6.1905	4.7368	3.4483	3.9326	3.2407	3.3981	3.6697	5.7416	4.2857	3.3816	2.9126
Testing Batch: $\mathcal{B}_{b+1}$																	
$(b+1)$	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84		
Batch size	218	209	206	201	157	208	194	219	142	180	182	164	165	172	178		
$C_{\hat{\beta}^*_{b,\lambda=0}}$	4.5872	3.8278	0.9709	3.4826	3.1847	2.4038	3.0928	5.9361	7.0423	2.7778	4.9451	3.6585	2.4242	2.9070	1.6854		
$C_{\tilde{\beta}_{b,\lambda=0}}$	4.5872	3.8278	0.9709	3.4826	3.1847	2.4038	3.0928	5.9361	7.0423	2.7778	4.9451	3.6585	2.4242	2.9070	1.6854		
$C_{\hat{\beta}^*_b}$	4.5872	3.8278	0.9709	3.4826	3.1847	2.4038	3.0928	5.9361	7.0423	2.7778	4.9451	3.6585	2.4242	2.9070	1.6854		
$C_{\tilde{\beta}_{b,\lambda}}$	4.5872	3.8278	0.9709	3.4826	3.1847	2.4038	3.0928	5.9361	7.0423	2.7778	4.9451	3.6585	2.4242	2.9070	1.6854		
$C_{\tilde{\beta}_{b,\lambda*}}$	4.5872	3.8278	0.9709	3.4826	3.1847	2.4038	3.0928	5.9361	7.0423	2.7778	4.9451	3.6585	2.4242	2.9070	1.6854		

* $C_{\hat{\beta}^*_{b,\lambda=0}}$ denotes the results obtained from the offline method without variable selection, while $C_{\tilde{\beta}_{b,\lambda=0}}$ represents the results obtained from the renewable estimation method without variable selection. On the other hand, $C_{\hat{\beta}^*_b}$ denotes the results obtained from the offline method incorporating SCAD, while $C_{\tilde{\beta}_{b,\lambda}}$ and $C_{\tilde{\beta}_{b,\lambda*}}$ represent the results obtained from the penalised renewable estimation methods introduced in Section 4.2.2 and Section 4.4.1, respectively. The misclassification rates of each classifier remain the same across all cases.

Table C.14: Comparison of misclassification rates (%) for various classifiers

Testing Batch: $\mathcal{B}_{b+1}$																	
$(b+1)$	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Batch size	191	243	202	246	212	234	211	231	251	214	210	235	216	221	208	223	241
$C_{\hat{\beta}^*_b}$	5.7592	3.7037	1.9802	4.4715	1.4151	1.7094	3.7915	0.8658	3.5857	3.7383	3.8095	0.8511	2.7778	4.5249	4.3269	1.3453	0.4149
$C_{\tilde{\beta}_{b,\lambda}}$	6.2827	4.9383	1.4851	3.6585	1.4151	1.7094	3.3175	0.8658	3.5857	2.8037	3.8095	0.8511	2.7778	3.6199	4.3269	1.3453	0.4149
$C_{\tilde{\beta}_{b,\lambda^*}}$	6.2827	4.9383	1.4851	3.6585	1.4151	1.7094	3.3175	0.8658	3.5857	2.8037	3.8095	0.8511	2.7778	3.6199	4.3269	1.3453	0.4149
Testing Batch: $\mathcal{B}_{b+1}$																	
$(b+1)$	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35
Batch size	210	215	199	159	143	159	196	165	147	169	150	122	153	194	177	182	152
$C_{\hat{\beta}^*_b}$	3.8095	3.7209	3.0151	3.7736	4.1958	5.0314	1.0204	2.4242	2.0408	2.9586	3.3333	4.0984	0.6536	3.0928	1.1299	6.0440	0.6579
$C_{\tilde{\beta}_{b,\lambda}}$	3.8095	3.7209	3.0151	3.7736	4.1958	4.4025	1.0204	1.8182	2.0408	2.9586	3.3333	4.0984	0.6536	3.0928	1.1299	5.4945	0.6579
$C_{\tilde{\beta}_{b,\lambda^*}}$	4.2857	3.7209	3.0151	4.4025	4.1958	5.0314	1.0204	1.8182	2.0408	2.9586	3.3333	4.0984	0.6536	3.0928	1.1299	5.4945	0.6579
Testing Batch: $\mathcal{B}_{b+1}$																	
$(b+1)$	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52
Batch size	163	222	145	143	85	73	98	77	71	70	100	83	100	109	85	107	116
$C_{\hat{\beta}^*_b}$	3.0675	1.3514	1.3793	3.4965	7.0588	10.9589	5.1020	6.4935	1.4085	17.1429	5.0000	8.4337	3.0000	1.8349	1.1765	5.6075	2.5862
$C_{\tilde{\beta}_{b,\lambda}}$	3.0675	1.3514	1.3793	2.7972	7.0588	10.9589	5.1020	6.4935	1.4085	17.1429	5.0000	7.2289	3.0000	2.7523	1.1765	5.6075	2.5862
$C_{\tilde{\beta}_{b,\lambda^*}}$	3.0675	1.3514	1.3793	2.7972	7.0588	10.9589	5.1020	6.4935	1.4085	17.1429	5.0000	8.4337	3.0000	1.8349	1.1765	5.6075	2.5862
Testing Batch: $\mathcal{B}_{b+1}$																	
$(b+1)$	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69
Batch size	123	111	114	132	116	116	104	92	99	90	106	95	114	99	103	112	98
$C_{\hat{\beta}^*_b}$	2.4390	6.3063	4.3860	0.7576	5.1724	2.5862	9.6154	6.5217	6.0606	4.4444	2.8302	2.1053	5.2632	7.0707	5.8252	4.4643	2.0408
$C_{\tilde{\beta}_{b,\lambda}}$	2.4390	6.3063	4.3860	0.7576	5.1724	2.5862	9.6154	6.5217	6.0606	4.4444	1.8868	2.1053	5.2632	7.0707	4.8544	5.3571	3.0612
$C_{\tilde{\beta}_{b,\lambda^*}}$	3.2520	6.3063	4.3860	0.7576	5.1724	2.5862	9.6154	6.5217	6.0606	4.4444	1.8868	2.1053	6.1404	7.0707	4.8544	4.4643	3.0612
Testing Batch: $\mathcal{B}_{b+1}$																	
$(b+1)$	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84		
Batch size	110	99	105	96	74	97	78	91	65	78	85	76	77	70	86		
$C_{\hat{\beta}^*_b}$	5.4545	4.0404	0.9524	7.2917	4.0541	2.0619	3.8462	5.4945	7.6923	3.8462	5.8824	5.2632	3.8961	2.8571	3.4884		
$C_{\tilde{\beta}_{b,\lambda}}$	5.4545	4.0404	0.9524	7.2917	2.7027	2.0619	3.8462	5.4945	6.1538	3.8462	5.8824	5.2632	3.8961	4.2857	3.4884		
$C_{\tilde{\beta}_{b,\lambda^*}}$	6.3636	4.0404	0.9524	7.2917	4.0541	2.0619	3.8462	5.4945	6.1538	3.8462	5.8824	5.2632	3.8961	2.8571	3.4884		

* $C^*_{\hat{\beta}_b}$ denotes the results obtained from the offline method incorporating SCAD, while $C_{\tilde{\beta}_{b,\lambda}}$ and $C_{\tilde{\beta}_{b,\lambda^*}}$ represent the results obtained from the penalised renewable estimation methods introduced in Section 4.2.2 and Section 4.4.1, respectively.