A generic data representation for predicting player behaviours

Hanting Xie

PhD

University of York

Computer Science

September, 2017

Abstract

A common use of predictive models in game analytics is to predict the behaviours of players so that pre-emptive measures can be taken before they make undesired decisions. A standard data pre-processing step in predictive modelling includes both data representation and category definition.

Data representation extracts features from the raw dataset to represent the whole dataset. Much research has been done towards predicting important player behaviours with game-specific data representations. Some of the resulting efforts have achieved competitive performance; however, due to the game-specific data representations they apply, game companies need to spend extra efforts to reuse the proposed methods in more than one products. This work proposes an event-frequency-based data representation that is generally applicable to games. This method of data representation relies only on counts of in-game events instead of prior knowledge of the game. To verify the generality and performance of this data-representation, it was applied to three different genres of games for predicting player first-purchasing, disengagement and churn behaviours. Experiments show that this data-representation method can provide a competitive performance across different games.

Category definition is another essential component of classification problems. As labelling method that relies on some specific contions to distribut players into classes can often lead to imbalanced classification problems, this work applied two commonly used appraoches, i.e., random undersampling and Synthetic Minority Over-Sampling Technique (SMOTE), for rebalancing the imbalanced tasks. Results suggested that undersampling is able to provide better performance in the cases where the quantity of data is sufficient whereas the SMOTE has more chances when the dataset is too small to be balanced with the undersampling approach. Besides, this work also proposes a new category-definition method which can maintain a distribution of the resultant classes that is closer to balanced. In addition, the parameters used in this method can also be used to gain insight into the health of the game. Preliminary experimental results show that this method of category definition is able to improve the balance of the class distribution when it is applied to different games and provide significantly better performance than random classifiers.

Contents

List of Tables

List of Figures

Acknowledgements

I would first like to thank my thesis supervisor Prof. Daniel Kudenko and Dr. Sam Devlin of the University of York. They helped me so much not only on my research but also constantly giving me tons of suggestions to my PhD life and career skill development.

I would also like to thank Prof. Peter Cowling who gave me lots of helpful advices on my first publication during my PhD.

I would also like to acknowledge Mr. Alex Whittaker at Inspired Gaming Ltd. who gave me permissions to the data of their games for finishing my PhD Research. Similarly, Mr. Neil Hutchinson at Bigbit Ltd. who granted me permissions to work with the data of their games, too.

Finally, I must express my very profound gratitude to my wife and my parents for providing me with continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them. Thank you.

Declaration

I declare that this thesis is a presentation of original work and I am the sole author. This work has not previously been presented for an award at this, or any other, University. All sources are acknowledged as References.

Chapter 1

Introduction

1.1 Motivations and Hypotheses

The digital-game industry is one of the fastest growing industries in the world, with over 1500 new products published annually (Bauckhage et al., 2012). Due to the limitations of past technologies, data analytics was not commonly involved in the process of developing games. Most games were played off-line, such that their data could hardly be collected. This fact, however, has dramatically changed in recent decades because of the rapid spread of the Internet. Nowadays, because most games are played with the Internet connected across a variety of devices, an enormous number of player behaviours can be collected via telemetry methods. However, since the raw data collected is often noisy and massive, it is hard to extract useful information.

Data analytics has been heavily used in recent years to address this issue. Instead of providing only statistical descriptions of players to inform better decision making, data mining, an important element of data analytics, can make forecasts of important behavioural trends such that risks (e.g., the disengaging trends of some players) and opportunities (e.g., the trend that some players will become paying users) may be noted in advance (Yannakakis et al., 2013).

With the rapid growth of both data-driven game development and data-mining technology, several works have applied data-mining approaches to such tasks as anomalous behaviour detection (Ahmad et al., 2009; Kang et al., 2012; Laurens et al., 2007; Mitterhofer et al., 2009; Woo et al., 2012), player preference modelling (Charles and Black, 2004; Hunicke and Chapman, 2004; Missura and Gärtner, 2009; Pedersen et al., 2010; Shaker et al., 2010; Togelius et al., 2007, 2011; Yannakakis and Togelius, 2011), player-preference based AI characters (Aiolli and Palazzi, 2008; Bakkes et al., 2009; Bauckhage et al., 2003; Cowling et al., 2015), player-disengagement prediction (Borbora et al., 2011; Borbora and Srivastava, 2012; Debeauvais et al., 2014; Drachen et al., 2016; Hadiji et al., 2014; Kawale et al., 2009; Runge et al., 2014; Tarng et al., 2009) and player-purchasing prediction (Pluskal and Šedivý, 2014; Sifa et al., 2015). Approaches introduced in these works effectively predict the targets in the games in which they conducted the experiments; however, the common problem is the data-representation methods applied in these works can hardly be made generic. In the process of data mining, the data-representation stage aims to transform the raw dataset into a vectorised information format which is ready for training machine-learning models. In this format, each vector (also referred to as a *feature* in data mining) represents one or a combination of more attributes in the original unprocessed dataset. Therefore, decisions concerning how to construct these features are often important and can affect the performance of resultant classifiers. In most of the methods proposed in these works, the limitation of generality on data representation comes either from the game-specific vectors selected or from a lack of availability of some features in new games. Specifically, this happens when predictions are made based on game-specific data representations which use game events that appear only in a specific type of game (e.g., how many enemies killed in a war game), whereas the latter issue arises when some features are general enough but are not tracked in some games.

To solve the generality issue of the existing methods, this work introduces (as the main contribution) a generic data-representation method called *event-frequency-based data representation*, which can be migrated to build feature spaces in different games without any prior knowledge of them. Instead of extracting specific information from the game dataset for each player, this approach takes counts of all of the in-game events that happen to (or are generated by) this player as the data representation. In this way, the actual meanings of the events become less important. This is also why this data-representation method can be seamlessly migrated to different game products with minor modifications.

Apart from this issue, which exists in the stage of creating the data representation for training classifiers, another common issue that can be found in many existing works is the imbalanced dataset. Because the predictions of players' future behaviours is a classification task in machine learning (e.g., whether a player will be leaving the game in the near future or not), a labelling method is needed to describe the task and to distribute players into their corresponding classes for predictive purposes. While labelling methods often use some specific conditions to distinguish players, an imbalanced class distribution can easily be created if the majority of players can satisfy the defined conditions. Unfortunately, an imbalanced classification task may lead to imbalanced classifiers.

In this work, two existing approaches are used to solve the imbalance issues found in three commercial games. Experiments show that both methods are able to help only in some cases. This suggests that, when a dataset is biased by some labelling method, a balancing method might help but that stable improvement is not guaranteed. In addition, some labelling methods might sometimes also lead to small datasets which can hardly be used for training higher-dimensional classifiers. In this study, when the predictive purpose is to forecast churn behaviours, the dataset labelled by the churn-labelling method is not only small (lacking data samples); it also shows a distribution bias in some games. To deal with this type of issue, this work applied two existed balancing methods and reviewed their abilities for dealing with the bias issues in this research. This work also offers a new labelling method called disengagement over varying dates. This labelling method is close to the concept of churn, which also aims to predict players' disengagement behaviours, but it can maintain an approximately balanced distribution while labelling. In addition, this method also comes with two parameters that can be used to provide insight into the health of a game. Preliminary results for investigating the ability of this research can be found in Appendix A.

This section explains the current status of this research area and the challenges it faces. A basic introduction has been given of the research motivations and main contributions of this study. To investigate and verify the main contributions, several experiments were conducted that made extra contributions in different parts of the study. The next section identifies all the contributions made by this work.

1.2 Contributions

1.2.1 Main Research Hypotheses

As introduced in Section 1.1, this study offers a generic data-representation method for predicting player behaviours that is designed to work across different games. To investigate its utility, the following research hypothesis was proposed: ***Event-frequency-based data representation can be used to predict player behaviour with supervised learning to provide a significantly better performance than random guess and competitive performance while being compared to other state-of-the-art methods, where applicable.*** Detailed explanations of this hypothesis can be found in Section 4.5.1.

1.2.2 Contribution Summary

To test both of the main research hypotheses proposed in Section 1.2.1, several contributions have been made during this research. A summary is given in this section to introduce their content.

Event Frequency-based data Representation
As the main contribution of this work, a generic data-representation method is proposed which can form the feature space only on the counts of events created (or experienced) by players. Because counts of events are content irrelevant, this data-representation method can be smoothly migrated to a wide range of different game products for predicting player behaviours. This corresponds to first main research hypothesis stated in Section 1.2.1. Details of it can be found in Section 4.5.1.

Player First Purchase Behaviour Prediction
To test the hypothesis, this work applies event-frequency-based data representation to predict players' first-purchase behaviours in three different commercial games. It achieved significantly better performance than a random classifier. Details of this experiment can be found in Chapter 5.

Disengagement Labelling Method A new labelling method named 'disengagement' is proposed in this study which represents the disengaging behaviour of players. Unlike the commonly applied labelling method called 'churn', this method focuses on predicting players' disengaging trends instead of their exact leaving actions. Developers would have more time to retain players by using extra care because the player has not yet decided to leave the game. Further details of this labelling method can be found in Section 6.1.1.

Player Disengagement/Churn Behaviour Prediction
The event-frequency-based data-representation method are used to predict the players' disengagement and churn behaviours to verify the hypothesis. Experiments show comparisons of classifiers trained with this data-representation method and another state- of-the-art game-specific data-representation method. Further information about the experiments is offered in Chapter 6.

An Evaluation of Two Popular Class Balancing Methods
Imbalanced class distribution is another issue that is often seen not only in game data-mining problems but also in other data-mining research areas. This work discusses the general causes of imbalanced class distribution using two methods that are commonly

used to solve this type of issue: *random undersampling* and *SMOTE (synthetic minority over-sampling technique)*. Statistical comparisons are given to show that the random undersampling and SMOTE methods would help improve the performance of classifiers. Details of these comparisons can be found in Chapter 7

Disengagement Over Varying Dates Labelling method
Based on experiments in which datasets labelled with the churn definition are not large enough and are sometimes biased for training higher-dimensional data representation, this work proposes a new, alternative labelling method that aims to use all data samples while maintaining an approximately balanced dataset. This alternative can help to solve two common issues discovered in this research: imbalanced and small data samples. In addition, parameters optimised for balancing in this method can be used as indicators of the game's health. Preliminary results have shown classifiers trained in tasks balanced by this method can perform significantly better than random classifiers. Further details of it are covered in Appendix A.

1.3 Outline

This thesis is organised as follows:

1. First,a basic introduction to machine learning and data mining is given. At the same time, several types of metrics are explained that are collected in the game context.

2. Next, a literature review is given of several studies that have attempted to predict various player behaviours. The data-representation methods applied in these works are discussed in detail.

3. Afterwards, before diving into experiments, the research methods are discussed. This section includes a global view of all the consequent experiments.

4. To investigate the utility of event-frequency-based data-representation methods for predictive tasks, it is first used to predict decisions regarding first purchases in three commercial games of different genres.

5. In addition, this data-representation method is utilised to predict players' disengagement. During this experiment, some limitations of the event-frequency-based data-representation methods are discussed.

6. After the event-frequency-based data-representation method has been tested, another common issue–the imbalanced dataset, which was discovered while conducting these experiments–is introduced. Some explanations are given that will enable us to work out the possible cause of it, and some existing methods are applied to rebalance the dataset. A possible solution called *disengagement over varying dates* is also introduced as an alternative to deal with this type of problem. Experiments show that classifiers trained under balanced datasets can perform significantly better than a random classifier.

7. Finally, the conclusion is given to summarise the contribution of this work. Based on some limitations introduced earlier, further work that may be conducted is also introduced.

Chapter 2

Modelling with Data Mining

From games (Mahlmann et al., 2010) and films (Saraee et al., 2004) to serious areas like earthquake prediction (Otari and Kulkarni, 2012) and medical diagnosis (Soni et al., 2011), an increasing number of areas have entered the century of fast growing information. A very large amount of data is generated every second. According to Hilbert (Hilbert, 2013), in an average minute in 2012, around 2,000 search queries were received by Google, nearly 700,000 pieces of content were shared by Facebook users and almost 100,000 tweets were produced by Twitter users. Although data generated in these areas can occur in different formats, a common problem they face is that the data has been grown so massive that cannot be easily analysed. As mentioned in Chapter 1, data mining is considered as one of the most reliable ways for digging meaningful patterns from a very large amount of data.

This chapter offers an introduction to how and what type of game data is commonly collected in state-of-the-art methods. Next, the basics of data mining (including machine learning) are reviewed. This introduction covers most important aspects of how data is processed before and during analysis. ***Main points in this chapter:***

- ◆ an overview of the data in games,
- ◆ descriptions about data cleaning,
- ◆ review of the dimensional reduction methods, and
- ◆ review of the machine learning based data mining process.

2.1 The 'Data' in Data Mining

As discussed above, since databases have become extremely large, it has never been more difficult to extract valid information from them. Data mining, also referred to as *knowledge discovery from data* (KDD), is a promising technology that aims to summarize latent patterns/models and discover unsuspecting relationships in databases by applying both statistical approaches and machine-learning algorithms (Han and Kamber, 2011; Hand et al., 2001). As datasets have become increasingly massive, it is important to make sure that the data quality is good enough before a data-mining method is applied. This section discusses the data types that are generally collected and used for behaviour modelling. Next, it discusses basic data-relevant concepts and how data can be cleaned and simplified.

2.1.1 Game Telemetry

This section offers a basic explanation is given of what data types are generally used for player behaviour-predictive modelling. According to Drachen (Drachen et al., 2013), game telemetry is a method whereby developers can collect game data from player clients over a distance. In the modern game industry, it has become the most widely used method for collecting game data (Zoeller, 2013) from free-to-play games and even 'AAA' titled games. This is because a game-telemetry system can help a game company not only track the real-time performance of a game but also record players' historical behaviours to storage databases. As introduced by Weber (Weber et al., 2011a), the conventional way to implement the system is to integrate a client or basic collection logical codes into games and transmit data to a collection server. Depending on the preferences of different companies, the data collected might be either formatted or kept raw in the server for the convenience of further analysis. While transmitting data to collection servers, because exhaustively recording player behaviours' in every frame could easily lead to very large cost and heavy transition pressures, the widely used approach (e.g, used by Google Analytics, Yahoo Flurry, Game Analytics, Unity, Upsight and Game Sparks) is to transmit data based on the firings of in-game events. In other words, the game data is only be transmitted in accord with pre-defined, in-game events. Generally, these events are in-game actions or behaviours, chosen by developers, which should be able to reflect any update on the status of games and milestones players achieved. Therefore, it is crucial to design an optimal and informative game-telemetry system. In most cases, the development of a game-telemetry system is comprised of at least two elements: game metrics, data storage.

2.1.2 Game Metrics

To perform game analysis, data collected from telemetry would normally be transformed into a more interpretable format defined as *game metrics*. For example, in a shooting game, some samples of game metrics are logs of a player's shootings behaviours, hitting rate, weapon preferences, average number of bullets used for killing an enemy and so on. This can be different in a racing game in which metrics can include players' movements, final positions, items used, etc. In other words, game metrics can vary based on game type and game content. Therefore, generic game data that can be found in all types of games is generally limited.

As introduced by Drachen (Drachen et al., 2013), game metrics can include three types: *player metrics*, *process metrics* and *performance metrics*. Of these, process metrics and performance metrics are more beneficial for adjusting system development and performing bug fixes than for game-analytics purposes. *Player metrics* stands for all data related to players' behaviours and gameplay tracks. This is the main sort of metric that game analytics is applied to. The definitions of *player metrics* can be further split into subcategories (shown in Figure 2.1). The definitions are as follows:

Customer Metrics

Transactions are commonly seen in modern free-to-play games, and it is necessary to track the details of each transaction. Associated with player identity, this type of log could include in-game/out-game payments for game time, products and any items in the game. An example application of this can be found in the work of (Lim and Harrell, 2013) which uses players' preferences for hats (an item) in the game, *Team Fortress 2* (Valve, 2007), to investigate taste performances.

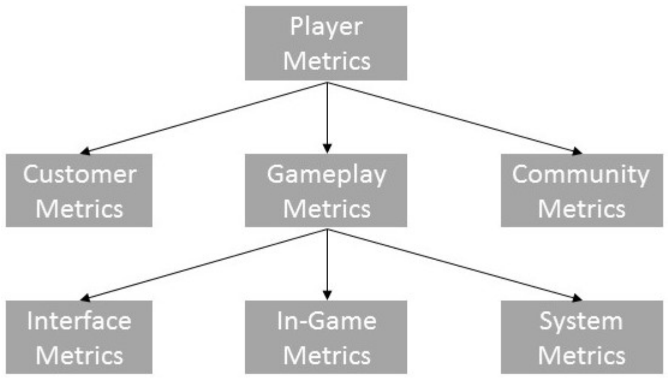

Figure 2.1: The Subcategories in Player Metrics (Drachen et al., 2013)

Community Metrics

This type of metric is able to reflect social interactions among players in a game. It is common for modern games to integrate some social networks, whether third-party platforms (e.g., Facebook and Twitter) or game-specific platforms (e.g., guilds in *World of Warcraft*). These metrics help researchers/developers gain a better understanding of the behaviours in a social group rather than of individuals. For example, Thurau and Bauckhage (2010) investigated the evolution patterns of guilds (a form of community) in the game, *World of Warcraft* (Blizzard, 2004). At the same time, these metrics can also be used to find similar styles among players in games. As an example of this, Kirman and Lawson (2009) discuss how social communities in online games could be utilised for identifying play styles.

Gameplay Metrics

Gameplay metrics is an important sub-category which represents most general in-game behaviours. This can be further broken down into *interface metrics*, *in-game metrics* and *system metrics*.

Interface metrics describe how players interact with the interfaces of a game. For example, in the main menu of games, metrics are collected concerning which options are used most by players and what purposes they use them for.

In-game metrics show how players behave in a game. This category is comprised of any metrics that are related to players' behaviours during gameplay. It can help to reflect players' skills, activities and preferences. For example, Pederson studied how to model players' experience in *Infinite Mario Bros* (Markus-Persson, 2008) by using in-game metrics such as death rate, jump times and so on (Pedersen et al., 2009).

System metrics record the system status corresponding to operations by players. Level settings provide an example. Shaker et al. (2010) looked into how to generate game levels dynamically. He relies on a model which maps the level designs of games into player experiences.

Game metrics collected could be analysed directly by simple visualisation and basic statistical methods (Tychsen, 2008). However, to gain a deeper understanding of those metrics, the preferable way is to apply data-mining methods on them to discover latent patterns and build predictive models.

2.1.3 Data Storage

Game metrics collected can be transmitted either to local servers or to some third-party services. In recent years, several third-party services such as Google Analytics, Yahoo Flurry, Game Analytics, Unity and Game Sparks have come to provide an easy way to store data on their servers. The benefits of this storage method are that some basic data visualisations can be done through these services for quickly gaining initial knowledge of game metrics. But, to achieve predictive purposes, the data has to be further analysed using data-mining approaches.

2.1.4 Data Attributes

This section explains one of the most basic concepts in data mining: the 'attribute'. According to Han and Kamber (2011), data attributes are data fields, each of which represents a single characteristic/feature of the dataset. A dataset is comprised of data instances (points), and each data instance has a corresponding value for every data attribute. For example, to describe the features of a dataset of balls, radius, weight, colour and shape are generally used as data attributes. In this case, a football would be a data instance the properties of which can be described with these attributes.

Although data attributes can be very different in varying datasets, they can mainly be categorised into four types: nominal, ordinal, interval-scaled and ratio-scaled (Stevens, 1946).

Nominal
A nominal data attribute can be considered a categorical variable in which there is no order among options. For example, the class a student belongs to is a nominal attribute, because a class name is a choice from all classes and it is meaningless to compare the name 'Class A' to 'Class B'. A binary data attribute is a special case of a nominal variable which is sometimes considered a separate type. It is merely a nominal variable with two choices–for instance, gender (male or female).

Ordinal
An ordinal data attribute is also a categorical variable but relationships exist among the choices. For example, to answer the question, 'Are you happy with your salary?', the answers could be A) Excellent, B) Good, C) Ok, D) Bad. These four options are choices/categories, but it is meaningful to say that Option A is more positive than B.

Interval-Scaled
An interval-scaled data attribute is a general numerical variable. There are only two constraints on this type of value. First, it is meaningless to calculate the division between two interval-scaled values. Temperature is a good example of this. It is meaningful to state the difference between 40°C and 20°C is 20°C; however, it is confusing to claim that 40°C is twice 20°C, as it is not true that 40°C is two times warmer than 20°C. In addition, a zero value for an interval-scaled variable is nothing special relative to other values; e.g., 0°C is a valid temperature, and it is warmer than -1°C.

Ratio-Scaled
A ration-scaled variable is similar to an interval-scaled one except that it is meaningful for divisional calculation; zero value means the attribute does not exist. Height is a good example of ratio-scaled measurement. It is not only meaningful to compare two

Type	Example
Nominal	Equal Choices: ‘Class A’ and ‘Class B’, ‘Apple’ and ‘Banana’
Ordinal	Grades: ‘A+, A, A-, B+’, ‘Very Low, Low, Medium, High, Very High’
Interval-Scaled	Temperature: 110°C’ and 120°C’, Time on the clock: ‘1:30’ and ‘2:15’
Ratio-Scaled	Weight: 20 kg and 10 kg, Height: 50 cm and 10 cm

Table 2.1: Types of Attributes

heights but also reasonable to state that 100 m is twice 50 m. For instance, a building of 100m is exactly two times taller than one of 50 m. When it is said that the height of something is zero, it means it has no height.

Table 2.1 gives examples of every single data attribute which can help to further explain the differences.

2.1.5 Data Cleaning

To reach an acceptable efficiency of collection during runtime, most data is stored in informal formats along with unpredictable coincidences and random errors. For this reason, metrics from game telemetry can be both noisy and incomplete. These problems can be introduced at any time during either the process of data generation or collection. Thus, a cleaning process is sometimes needed before the analysis to ensure that the data can achieve a certain standard of reliability. In general, the process of data cleaning is about detecting outliers and missing values in the database. This section introduces some basics about data cleaning will be introduced.

Outlier Detection

As a common start for cleaning data, outlier detection is often needed to filter out data points that do not follow the data distribution. Outliers are outlying observations which might be generated by coincidence or error during data collection.

Several methods are used for outlier detection. The easiest to implement and most widely used method for univariate outlier (single dimensional outliers) detection is to use k times the standard deviation distance from the mean as a criterion (Seo, 2006). This can be referred to simply as the standard-deviation method. It tries to label a data point as an outlier if the point is located outside of the region $[mean - k * std, mean + k * std]$.

However, this criterion assumes that the data follows a normal distribution, which is not feasible for many practical problems. A modification method which works on data with any distributions is called a boxplot. It was produced by John Tukey in 1977 (Tukey, 1977). Instead of using mean as the centre of the data space, it splits the dataset into three quartiles (Q_1, Q_2, Q_3) and uses $(Q_1 + Q_3)/2$ as the centre. Thus, an outlier is called an *extreme outlier* if it lies outside of the region $[Q_1 - 3 * IQR, Q_3 + 3 * IQR]$. It is called a *mild outlier* if it lies outside of the region $[Q_1 - 1.5 * IQR, Q_3 + 1.5 * IQR]$, where IQR is called interquartile range which equals to $Q_3 - Q_1$.

Although these easy-to-implement methods work well for a univariate situation, they are not reliable for multivariate outlier detections. This type of outlier detection is necessary when an analysis depends on more than one independent variable. In this case, even if a data point is not an outlier in each dimension of a dataset individually, the combination of several dimensions may change the situation (Acuña and Rodriguez, 2004). To solve it, two approaches are typically applied: a *minimum-volume ellipsoid (MVE)* estimator and a *minimum-covariance determinant (MCD)* estimator.

The MVE estimator is fairly easy to understand, as it aims merely to minimise the area of an ellipsoid which can cover h points out of n (all points) where $n/2 \leqslant h < n$. The h stands for the final inlier points (Van Aelst and Rousseeuw, 2009). The MCD estimator looks for h inlier points whose classical covariance matrix shows the lowest determinant. It is claimed by Rousseeuw and Driessen (1999) that MCD is a good replacement MVE as it can not only provide a better statistical performance that MVE while being calculated but also gives more precise robust distances (Butler et al., 1993).

Missing Value Imputation

In practical data analysis, it is common to encounter missing values (Little, 1992). Several approaches have been invented to handle this problem. Generally, methods for missing value imputation can be considered in two branches: *single imputation* and *multiple imputation* (Donders et al., 2006).

The most commonly used single-imputation methods include the *missing indicator* and *overall sample mean*. Missing indicator methods utilise a separate column of dummy numbers (0/1) to indicate whether the current line of data is missing or not. This column is taken into consideration as an individual data attribute during statistical modelling (Groenwold et al., 2012). However, this method has been proven to lead to biased estimates of the correlations between independent variables and outcomes (dependent variables) (Donders et al., 2006; Groenwold et al., 2012). Different from it, the overall sample mean replaces the missing value simply with the mean of the values in the same column (data attribute). However, like the missing-indicator method, this method has been claimed to provide biased correlation (Donders et al., 2006).

Due to the limitations of these simple methods, *multiple imputation* has become increasingly popular and is claimed to be a good approach which can help to decrease the effects brought by missing data (Fox-Wasylyshyn and El-Masri, 2005; Yuan, 2010). The problem with single imputation (i.e., normal method for filling in blanks) is that it achieves final analytics purposes by assuming that the numbers are 'true' or similar to 'true'. This may easily lead to a problem if the estimated values vary considerably from the actual values. To solve it, multiple-imputation method first uses several different estimation methods to form many 'possible' datasets. The final analytical result is an average of analyses from these 'possible' datasets. This tries to reduce the variances caused by value imputations and provides a more reliable result.

2.1.6 Data Reduction and Data Representation

Data collected from practical spaces is often of large quantity and large dimension, which can cause plenty of time spent or mislead an analysis. To solve this problem, in the area of data mining, data reduction is an indispensable tool that is needed before any analysis (Han and Kamber, 2011; Wei, 2010). Data reduction is a technique which aims to reduce the complexity of data while keeping the result of the analysis almost unchanged. A series of data-reduction processes is also sometimes referred to as the *data representation method*, after these data reduction processes have been applied, the ready-to-use data is now a representation of the original dataset. In data mining, the main strategies used to perform data reduction are *dimensional reduction* and *numerous reduction*.

Dimensional Reduction

The dimension of the raw dataset is generally high, which may lead to very large consumption of computing resources while performing any data analysis. In data mining, three main branches of approaches can be used in this situation: *attribute subset selection*, *attribute construction* and *data compression* (Han and Kamber, 2011).

The idea of attribute-subset selection is simple. The method is often used when the analytical objective is more likely to be correlated only with a subset of all data attributes. For example, when predicting the winning rate of a player, it is more reasonable to consider behaviours of the player rather than his/her name though the name is also a valid attribute in the dataset. However, sometimes the relationships between the independent variable and the analytical purpose are rather complex. To deal with this issue, a branch referred to as *feature selection* is more helpful, as it utilises various machine-learning algorithms (e.g., linear regression, support vector machine, etc.) for estimating the relationship factors (e.g., coefficients) between each independent variable and the analytics purposes. Based on the estimations, variables with higher estimated relationships can be considered more important factors while lower ones may be dismissed accordingly (Barbar'a et al., 1997).

Attribute construction aims at generating new and replacing old attributes based several raw attributes. For the same example of predicting the winning rate, while observing players' behaviours, rather than focusing on their total health reduction and the number of levels played during the last day, it is probably more meaningful to combine them and observe the average health reduction over levels (which equals the division of these raw attributes) during the last day.

When the number of data attributes is much larger than the number of observations (data points) and a subset or combination of attributes cannot be easily decided (which happens when the relationships are complicated), a more advanced approach called data compression is needed. Rather than using the original data attributes, data-compression methods aim to discover new attributes/features which can form new data spaces and perform analysis in the new data space instead. The new data space generated is mostly a transformation of the original data space. The most commonly used approaches are *Principal Components Analysis (PCA)* (Sammut and Webb, 2011) and *Discrete Wavelet Transform (DWT)* (Barbar'a et al., 1997; Qu et al., 2003).

The idea of PCA is based on *singular value decomposition (SVD)*, which tries to project the original raw data matrix into the direction that maximises the data-projection variances. It can also be considered rotating the axis of the original data in the direction in which the data is spreading most broadly. After the rotation, data is represented in new attributes (axis) in the new data matrix, where these new attributes are combinations of the original ones. A simplified situation can be seen in Figure 2.2, where the y' and x' are the new attributes discovered.

Like PCA, the DWT method also performs rotations on the original axis (attributes); however, the direction now is not decided by the data-projection variance. In PCA, the rotation is done via calculations over the whole dataset. Different from it, in DWT, the transform is first calculated by decompressing data (by splitting signals to 'main trends' and 'details') along each axis (column/attribute) into orthogonal wavelets and

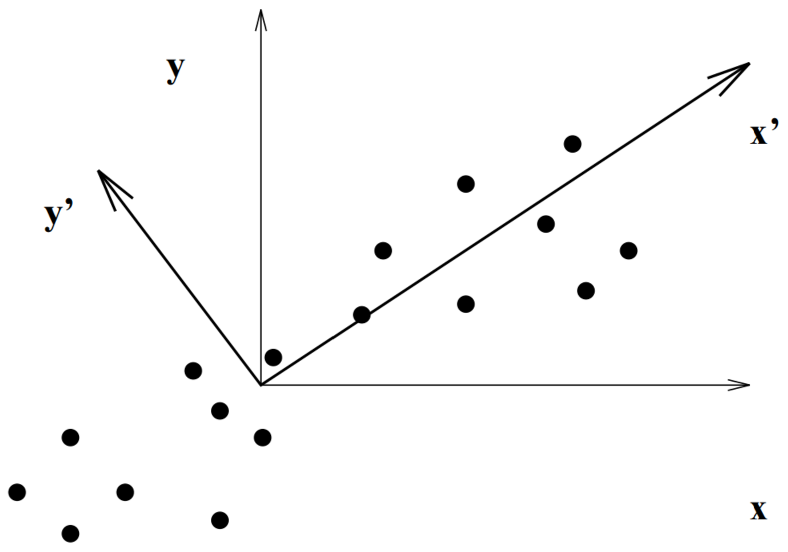

Figure 2.2: The idea of SVD (Barbar'a et al., 1997)

then find the common optimal transformations with the best wavelet coefficients along each direction. Since the calculation is done on each data axis separately, the efficiency is faster than PCA (Qu et al., 2003).

Numerosity Reduction

Numerosity reduction is most needed when a dataset is too large to be processed with limited storage resources and computation time. To decrease the size of the database, it simply tries to use a smaller set of data to represent the original dataset. Two categories of methods are being used in this case: *parametric methods* and *non-parametric methods* (Han et al., 2006)

Parametric methods generally try to fit the original data into some models (keep outliers at the same time) and then only store a combination of the parameters of the best-fitted model for representing the dataset. *Regression* and *Log-Linear models* are most commonly used. The regression method is a set of methods which aims to find correlations between one (or more) independent variable(s) $(x_1, x_2, ..., x_n)$ and a dependent numerical variable (y).In the simplest version, which is called linear regression, the fitted model can be mathematically shown as in Formula 2.1 where the b stands for the coefficients and c represents a constant number. Based on the regression result, the parameters of the model can be used as a representation of the whole dataset (Barbar'a et al., 1997; Han et al., 2006). The log-linear models work at estimating parameters that can provide the maximum probabilities that can reconstruct the original, high-dimensional data points from some smaller subset of lower-dimensional attributes (Han et al., 2006; Smith, 2004). These parameters are mostly estimated using max-likelihood methods. Thus, this method can be used for numerosity and dimensional reduction at the same time (Barbar'a et al., 1997). These parameters can finally be used for representing the original dataset.

$$y = b_1 * x_1 + b_2 * x_2 + ... + b_n * x_n + c \tag{2.1}$$

Without considering models, the non-parametric methods group data points into clusters and use cluster information to represent the original large-volume dataset. The most widely used methods are *Histogram*, *Clustering* and *Sampling*. A histogram shows the distribution of data by binning together data points with the same values along a selected attribute. As a result, each bin in a histogram represents multiple original data points. Similar to this, clustering randomly generates centres which merge all data points around them into clusters by unsupervised machine-learning algorithms (details are introduced in Section 2.2). In this case, each cluster corresponds to several original, similar data points. Similarly, the sampling approach tries to utilise a smaller sample set or subset to represent the original dataset without changing its distribution. It generally picks out data points by some random algorithms, including *random pick with replacement (take out)*, *random without replacement*, *random pick from clusters* and *stratified random pick*. These methods are suitable for different situations. The objective in applying them is to ensure that every data point shares the same probability to be selected.

2.2 Machine-learning based Data Mining

After being transformed by several pre-processing methods, the data becomes actionable for various analyses. As mentioned before, among many data-analysis approaches, data mining plays a key role in uncovering hidden patterns from data in depth. While basic data-visualisation methods and statistical tests work on describing the transformed data itself, data mining is specialised at extracting models from hidden correlations among attributes, which can be used for multiple purposes.

Working as the core engine of data mining, machine learning is a popular artificial-intelligence (AI) branch which has been widely used in multiple fields. From spam-email filtering to online shopping recommendations, NPC controlling in games and transaction prediction in economics, machine learning has shown its power across many areas. Machine learning can be described as a collection of algorithms or systems that can enhance their knowledge and performance with experience (Flach, 2012). From the perspective of use, it can also be explained as a set of methods that can automatically detect patterns in data and then predict the future or perform other decision making based on the pattern recognised (Murphy, 2012). It is so-called ‘machine learning’ because the process of its experience-based training is similar to the learning behaviour of humans. Based on the types of tasks that can be solved, machine-learning algorithms can mainly be categorised into four classes: supervised learning, unsupervised learning, learning association rules and reinforcement learning (Alpaydin, 2004; Flach, 2012; Hua et al., 2009; Murphy, 2012). Among them, the first three areas are directly related to data mining, thereby, their definitions will be briefly introduced in the following sections.

2.2.1 Supervised Learning

A typical supervised learning problem aims to work out correlations between several independent data attributes and a labelled target-dependent variable. The problem can be further split into two subcategories based on how the target is labelled. A problem with a categorical variable as the target is called classification, whereas it is called regression when the target is a number to be estimated (Caruana and Niculescu-Mizil, 2006). Figure 2.3 shows the basic structure of the whole process of supervised learning. Practically, supervised learning is usually applied for predictive modelling situations. For example, predicting the weather

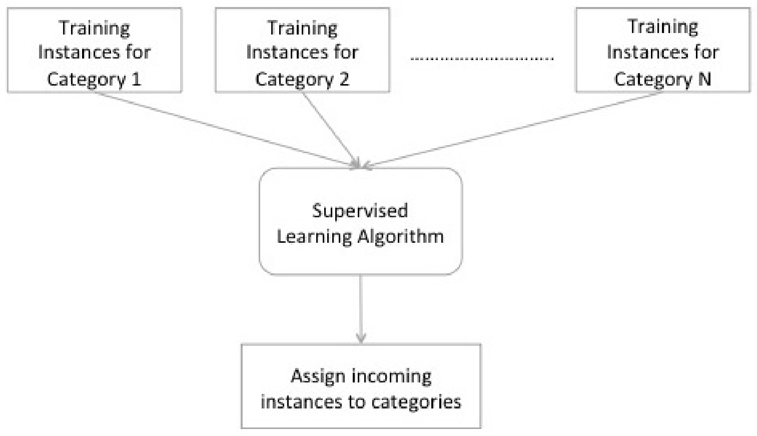
Training Instances for Category 1
Training Instances for Category 2
Training Instances for Category N
Supervised Learning Algorithm
Assign incoming instances to categories

Figure 2.3: Supervised Learning

tomorrow based on the recent climate conditions is a classification problem (Olaiya and Adeyemo, 2012) and foreseeing the price of a specific stock based on its transaction history is a regression purpose (Kannan et al., 2010).

Since this study is based on supervised learning, classical algorithms such as *Decision Tree*, *Logistic Regression* and *Support Vector Machine* are briefly introduced here.

Decision Tree

A decision tree is a model for supervised-learning problems. A standard decision tree is a hierarchical tree-like data structure following a divide-and-conquer strategy (Alpaydin, 2010). It is one of the most interpretable models in data mining. It is helpful for understanding what features could lead to which terminal nodes (categories) (Apté and Weiss, 1997).

Logistic Regression

Logistic regression is an important statistical-modelling method. According to Hosmer and Lemeshow (Hosmer and Lemeshow, 2004), it aims at discovering the best-fitting parametric model which could correctly describe the relationship between multiple independent variables (also referred to as covariates) and a dependent variable (categories in supervised learning). It is similar to a linear-regression algorithm in which the difference is the output of a logistic regression model is dichotomous rather than continuous. A logistic regression model is useful for solving many classification problems but cannot be easily interpreted.

Support Vector Machine

Support vector machine is a widely used algorithm for solving both classification and regression problems. As introduced by Campbell and Ying (Campbell and Ying, 2011), it first uses bounded-data instances in each class as support vectors and then maps them into a higher dimension via some kernel functions (e.g., Gaussian kernel). Afterwards, the algorithm looks for a hyperplane which can separate all canonical hyperplanes formed by support vectors. Although an SVM model can be used to achieve a promising accuracy, it can only be used as a black box because the resultant model is not easily interpretable.

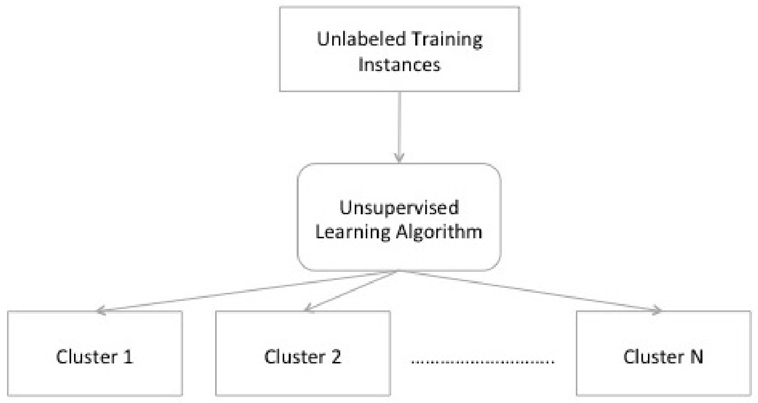

Figure 2.4: Unsupervised Learning

Besides the algorithms introduced above, another algorithm that is also widely applied in Machine Learning is the *Neural Networks*. This algorithm has been developed dramatically recently and shows its potential to be used across many applications. However, the limitation of this method is that its performance is correlated not only with the choices of its parameters, but also with the topology of the network. As Admuthe et al. (2009) discussed in their work, working out both choices is crucial to a success of a network. Thereby, this research did not include it as one of the methods to avoid possible topology sensitive results.

2.2.2 Unsupervised Learning

In data mining, another general purpose is to work out some potential patterns or intrinsic similarities from a dataset. In this case, the algorithm aims at gathering similar data points into clusters so that each of them represents a specific type of style or behaviour pattern (Ghahramani, 2004). The situation of unsupervised learning is shown in Figure 2.4. Because this type of problem is not for prediction, no labelled targets are needed for the training (i.e., it is unsupervised). Algorithms including k-means, DBSCAN (density-based spatial clustering of applications with noise), affinity propagation (etc.) are available for processing data under different situations. An example of an unsupervised learning problem is one that finds play styles of players in games by their behaviours (Bauckhage et al., 2015).

2.2.3 Mining Association Rules

Another challenging task of data mining is to discover rules (patterns) that are frequently followed in datasets. Generally, a rule follows the format of 'A happened, then B happens' where A and B could be single items or sets of items (Agrawal et al., 1993). This idea sounds simple, but it requires a very large amount of computing resources when a large dataset is given. The Apriori algorithm is commonly used for efficiently generating rules from large datasets. An example of a learning-association rules problem is the recommendation system. This type of system has become increasingly important in the last few decades because it is able to discover specific purchasing rules. For example, in Table 2.2, it is easy to see that there is a strong relationship between 'mobile phone' and 'protection case'; thus, a rule like 'customer bought mobile phone, then will buy protection case' will be possibly true if it covers at least a specific ratio (threshold) of the whole dataset. Based on rules like

Customers	Item A	Item B	Item C
Customer A	**Mobile Phone**	**Protection Case**	
Customer B	Earphone	**Protection Case**	**Mobile Phone**
Customer C	**Mobile Phone**	Mobile Charger	**Protection Case**

Table 2.2: Example of item set

this, individual recommendations can be made to different customers by their transaction histories (Sarwar et al., 2000).

2.3 Summary

This chapter discussed how data can be traced using game-telemetry methods. Afterwards, the types of game metrics were introduced to show that it is important to cover the types based on data-analytics objectives. Subsequently, the pre-processing of data–including de-noising and reduction–was covered. This part includes several methods for both dimensional and numerosity reduction. Finally, the basic idea of how data mining (with the core of machine learning) can be used to solve different types of problems was reviewed.

The next chapter covers the important literature that has been generated so far on the subject of game data mining. At the same time, the contributions and possible limitations are discussed in detail.

Chapter 3

Game Data Mining

Although the history of applying data mining to games is not as long as it is in other industries (Yannakakis, 2012), thanks to the development of data-mining technologies, several successful works have achieved promising results for a variety of specific purposes in different stages of developing a modern game. The area is nowadays called *game data mining*(Drachen et al., 2013).

Main points in this chapter:

- ◆ introduction to literature related to anomaly behaviour detection,
- ◆ introduction to literature related to player-style modelling,
- ◆ introduction to literature related to player-style based AI,
- ◆ introduction to literature related to player preference learning and procedure content generation,
- ◆ introduction to literature related to player disengagement prediction, and
- ◆ introduction to literature related to player purchasing prediction.

Of all research topics in game data mining, *player behaviour modelling* is one of the most basic and important concepts. It aims to detect or summarise patterns of actions that players have performed in games to achieve a better lifetime value (LTV) and to improve game development. These models can be either descriptive (for visualising the status of the dataset) or predictive models. This work is focused on the latter sort. By applying machine-learning techniques, predictive models can generally help to predict possible trends (e.g., player engagement, purchase decisions, etc.). Based on the predictive results, some pre-emptive measurements can be made for enhancing some expected trends and preventing some unexpected trends. If interpretable algorithms such as decision trees are applied, the resultant model might also be able to provide the developers with some insights of their players.

Since the purposes of this study is to come up with a generic data representation that can achieve competitive results for predicting both disengagement and first-purchase problems, data representations that have been used in relevant work is introduced in detail in the literature review. Via discussion, some limitations of the current selection of data representation in game data mining may become clearer. Grouped by different predictive purposes, research from the five main commonly researched areas is covered: *modelling for anomalous-behaviour detection*, *player-style modelling*, *player-preference learning*, *preference-based AI*, *player-disengagement modelling* and *player-purchase modelling*. These areas sometimes overlap with each other.

3.1 Modelling for Anomalous-behaviour detection

Based on the game metrics that have been introduced, several works with different predictive purposes in games are introduced. In games, especially online games, it is essential to prevent players from performing illegal operations which would affect the balance of the system and the experiences of other players. These illegal behaviours can either be cheating or potential sexual assaults in chat logs. However, because of the large quantity of player data, it is generally impossible to recognise these deviant players manually. Game data mining based *behaviour detection* is a technology that could recognise specific behaviour patterns from player behaviours.

A general application of this area is to try to detect bots in multi-player, online, role-playing games (MMORPG). Bots often exist in MMO games nowadays, because players can easily get important resources (mostly game-currency related) without even taking the time to play the game. These bots have been considered large threats to all MMO games, as they can cause a serious inflation problem in the game economy (Mitterhofer et al., 2009). Several efforts have been made to detect these bots. Among them, data-mining based approaches have been achieving competitive results. Research by Mitterhofer et al. (2009) and Kang et al. (2012) has tried to identify bots by observing their anomalous behaviour in games. Both works construct player behaviour models based on the history of the behaviours of players. However, their selections of data representations are not the same.

Of the three sources of literature to be introduced in this section, both Mitterhofer et al. (2009) and Kang et al. (2012) tend to focus on a specific direction of play behaviours. Mitterhofer et al. (2009) observe the movement patterns of players (or bots) and utilise the paths for training the classification models. This is reasonable, as most bots rely on some specific algorithms to make a move, whereas players' movements are more random. However, this solution is highly specific, as the game it investigates must at least contain movement behaviours. MMO board games such as Hearthstone (Blizzard, 2014) constitute extreme counterexamples, as there is no movement involved in the game at all. This limitation also applies to the work by Kang et al. (2012) as the predictive model built by them utilizes the players' chats log, which is yet another game-specific concept. In their study, this idea for detecting bots has been nicely followed out, as bots should chat slightly while 'playing' games and their chat contents should be less logical than real human beings in a game with a chatting system. Instead of filtering out bots only, many researchers have also investigated the detection of gold-farmer players in MMO games. Gold farming refers to players who mine virtual resources in the game and sell them for real-world money. As introduced by Ahmad et al. (2009), although this behaviour is not illegal and can bring fewer threats than bots, it can still destroy the working order of the ecosystem in a game. To prevent this issue from occurring in games, previous research relies on pure statistics (Laurens et al., 2007). Few predictive models have been built. For improvement, Ahmad et al. (2009) propose a feature space that can be used to identify a player as a potential gold farmer. The features selected by them are broader than those of Mitterhofer et al. (2009) and Kang et al. (2012). It includes 16 different attributes from four various directions: i.e., player character, aggregate, demographic and temporal features. On the basis of these features, the researchers applied several classification algorithms for achieving predictive purposes. Though Ahmad et al. (2009) provide a broader selection of features, most of them are still specific to some types of game and therefore can hardly be extended to other games. Apart from protecting the ecosystem of a game, another important application of game data mining in anomalous-behaviour detection is to secure players' accounts. With the development of online games, the security issue has become increasingly important in recent decades, as the accounts of players are often precious because they may either be associated with items purchased in-game or have consumed a lot of playing time. In addition, if the player has used the same combination of ID and password in other contexts, the loss of their accounts may bring unexpected extra threats. Woo et al. (2012) employ a different set of features, including the RFM (how recently, how frequently and how much) of purchasing behaviours and the login behaviours associated with IP addresses. However, the IP address information used in the work may not be available in many games, as it is related to players' private information.

As can be seen, several studies have tried to use game data-mining approaches to detect players' specific behaviours. Most of these works achieve competitive results for detecting the illegal behaviours. However, due to their selections of features, the data representations introduced in this section can hardly be migrated to another game–especially a game of a different type. In fact, this issue is not limited to this area; it also applies to the study area introduced in the next section.

3.2 Player Style Modelling

Because human beings differ in behaviour, it is rather difficult to satisfy individual players. To generate personal content or understand the most valuable customer groups, it is crucial to construct preference models in games. *Player Style Modelling* is an area that aims to understand the preferable way that players apply to play games. Models of player-styles can be used in several different ways, including adapting game settings for specific players (Charles and Black, 2004; Hunicke and Chapman, 2004) and generating new content for players (Missura and Gärtner, 2009; Togelius et al., 2007, 2011). This section reviews several kinds of literature in this area.

Since player-styles are relatively abstract concepts, these styles are generally created by various machine-learning algorithms for unsupervised learning (Baumgarten, 2010; Drachen et al., 2009, 2012; Gow et al., 2012; Thawonmas et al., 2006). Though the algorithms considered in existing studies are similar, the data representations are different. Using pre-defined play styles in an online game with closed space filled and several landmarks, Thawonmas et al. (2006) successfully discovered five different movement patterns by treating players' movement trails as a feature space for discovering different players' preferences. This selection of feature space is similar to that in Mitterhofer et al. (2009); the difference is in the way the players' trails were represented. Therefore, both works share the same limitations, as these features work only for games in which players can freely move in some specific space. Other works have enlarged the selection of features. Both works by (Drachen et al., 2009, 2012) select several game-specific features from three commercial games: *Tomb Raider:Underworld* (Eidos-Interactive, 2008), *Tera: Rising* (Bluehole, 2011)and *Battlefield 2: Bad Company 2* (EA, 2010). They successfully discovered some typical types of users based on visualisations. These selected features mostly cover two aspects: players' social activities and players' performance. On top of their work in 2009, Gow et al. (2012) and Baumgarten (2010) believe that, because these styles are generated from players across different scenarios, they may be influenced by complex regularities. Both of these two works used *linear discriminant analysis (LDA)* based on selected game-specific features to find the optimal vectors and transform the original data space into a lower-dimensional space with only the important features.

Therefore, players' locations under the reduced data representation space can reflect their preferences and can be grouped by a further clustering procedure.

As can be found in the literature that aims to detect player-styles with unsupervised learning approaches, the limitations of features that we observed in the area of anomalous-behaviour detection still hold. Most studies in this area rely on more complex feature spaces than those seen in anomalous-behaviour detection. The features selected are more dependent on player-performance metrics that vary in different games types. Data representations used in these works can hardly be migrated to others, as the performance of players is mostly related to the gameplay of games, and investigations can hardly be made without knowing their content. In game data mining, once players' preferences have been discovered, in addition to apply further visualisation analysis to understand player groups, another study might try to make AI characters learn from these styles to interact with players in a more realistic manner. The next section introduces studies that are relevant to player-style-based AI.

3.3 Player Style-based AI

Based on the models built for player styles, rather than just understanding players' behaviours or preferences, the studies introduced in this section looks further into how these models can be utilised for creating AI-based NPCs (non-player characters). In a game's world, an NPC can be either an opponent or other virtual characters involved in the game world. NPCs, especially opponents, are important actors that can make the game more interesting and challenging. Unfortunately, compared with the great changes happening with the graphic technologies of games, current virtual character controls still mostly rely on scripted behaviours or simple decision trees (Aiolli and Palazzi, 2008). By using player-behaviour models, it is feasible to create AI-based NPC behaviours which can make games more challenging and attractive.

For example, like the studies introduced in Section 3.2 and using the unsupervised learning method K-means, Bakkes et al. (2009) automatically generated clusters of similar player behaviours with ten self-selected features from an RTS game named *Sprint* (Spring-Engine, 2008). After the player-style model is built and a current player is classified into the nearest cluster, corresponding predefined corresponding strategies are applied to the AI behaviours. Instead of using the similar unsupervised approaches introduced in Section 3.2, Aiolli and Palazzi (2008); Bauckhage et al. (2003); Cowling et al. (2015) applied supervised learning approaches to achieve similar targets. As one of the most interpretable of supervised-learning algorithms, a decision tree was applied by Cowling et al. (2015) for generating a human-strategy model from gameplay histories of a leading card game *Spades* (AI-Factory, 2011). The features used for building the decision tree are abstractions of different possible moves in the specific Spades games. The strategies described in the tree can act as guidelines for AI players so that they will make choices similar to those human beings would make and behave more like real players. Rather than using discovered player preferences as suggestions, both Aiolli and Palazzi (2008) and Bauckhage et al. (2003) attempt to determine a state-action mapping function from games based on players' gameplay logs with machine-learning algorithms. While the choices of features made in the work by Aiolli and Palazzi (2008) are similar to those made in the work by Cowling et al. (2015) in a board game (Randolph, 1980), their selections of feature space include not only possible moves but also initial positions and situations of pieces before capture. Similarly, to discover an efficient mapping function in the FPS game *Quake II*, Bauckhage et al. (2003) created neural networks for mapping the players' field of view and velocity in the game to their next actions.

In summary, based on models of player behaviours, both supervised and unsupervised learning models can be applied for learning some functions that map the status of agents to their next actions. Among all the studies introduced in this section, the data representations applied are mostly based on players' specific behaviour in the types of games to which the method was applied. This discovery matches what we have seen in two previous research areas. Player-styles modelling approaches introduced in these two sections aim at finding players' potential types or styles while playing a game, these styles can be considered as the preferences that players follow across the game. However, this is different from investigating players' preferences on specific choices in different situations in game. To learn and predict player's preferences, another research area named 'player-preference learning' has attracted increasingly attentions, and it will be covered in this next Section.

3.4 Player Preference Learning and PCG (Procedure Content Generation)

Preference learning has become increasingly important during the last decade not only in the game research area (Abou-Zleikha and Shaker, 2015; Pedersen et al., 2010; Shaker et al., 2010; Yannakakis et al., 2009; Yannakakis and Togelius, 2011) but also in many other research directions (Chu and Ghahramani, 2005; Fürnkranz and Hüllermeier, 2010). In the game context, the applications of it mainly focus on working out the personal preferences on games content so that developers could use the models to enhance the game content for improving players' experience (Pedersen et al., 2010; Shaker et al., 2010; Yannakakis and Togelius, 2011).

There are generally three different types of preference learning tasks, they are label ranking, instance ranking and object ranking respectively (Abou-Zleikha and Shaker, 2015). In game context, the commonly seen researches (Abou-Zleikha and Shaker, 2015; Martínez et al., 2010; Pedersen et al., 2010; Shaker et al., 2010; Yannakakis et al., 2009; Yannakakis and Togelius, 2011) are focused on object ranking that aims at predicting the players' preference over pairwise of variations of games based on some known pairwise preference relationships. The pairwise preference relationships (referred to as player experience) in these works are represented either by the 2-alternative forced choice protocol with the affective state 'Fun' (Yannakakis et al., 2009) or 4-alternative forced choice protocol with the affective states *Engagements*, *Frustration*, *Challenge*, *Predictable* and *Anxious* (Abou-Zleikha and Shaker, 2015; Martínez et al., 2010; Pedersen et al., 2010; Shaker et al., 2010; Yannakakis and Togelius, 2011). The performance for each affective state is verified by the number of correctly classified pairs. Among the research works, Yannakakis et al. (2009), Martínez et al. (2010), Pedersen et al. (2010) and Abou-Zleikha and Shaker (2015) are focused on the methods that can be used for predicting pairwise preferences. Yannakakis et al. (2009), Pedersen et al. (2010) and Martínez et al. (2010) applied ANN (with Genetic Algorithm) in their works as the main algorithm to achieve the prediction whereas Abou-Zleikha and Shaker (2015) applied a modified random forest (with Grammatical Evolution) as the classifier. As for the data representations, all the three mentioned research works applied game specific data representations in their testing games. The features used by Yannakakis et al. (2009) are specific gameplay logs in the game *Bug-smasher* including the press of tile (a location in the game), the pressure force on the tile, the tile's colour, the game speed, and the entropy of bugs' visits (enemy in the game). Both Pedersen et al. (2010) Abou-Zleikha and Shaker

(2015) were using the game *Super Mario Bros* as the test game. The difference in their selection of data is that, in the work by Pedersen et al. (2010), game-specific features are classified into controllable features and gameplay features so that the controllable features can be further used in the future for procedurally generating game content. Instead, in the work by Abou-Zleikha and Shaker (2015), there were 36 feature selected in total of five types fully represent players' gameplay, which is similar to the gameplay features selected in the work by Pedersen et al. (2010). Data representation applied in the work by Martínez et al. (2010) is also specifically for the game *Maze ball*, the selected features include five categories, i.e., players' performance, time of staying in specific location, space distance among objects, players' input and the visibility of object from camera.

Different from them, several other research works in the preference learning research area focus on how these preference predictive model can be utilised for creating procedure generated game content (Shaker et al., 2010; Yannakakis and Togelius, 2011). In both research works, genetic algorithms were applied for maximising the specific aspects of players' experience (represented by 4-alternative forced choice protocol) by generating preferable contents based on the predictions from preference learning models. Data representation applied in the work by Shaker et al. (2010) is claimed to be modifiable to work in similar shooting games because it includes some common concepts in this type of game such as *number of times player shoots*, *entropy of objects placement*, *number and concentration of enemies/items/obstacles*. Instead, the work by Yannakakis and Togelius (2011) is a review of how preference learning models can be built and utilised for PCG. In this work, several different ways of data representations were covered, including the gameplay data (including how players interact with games and performance) for model-free methods and the physiological data (electrocardiography, galvanic skin response, etc) for model-based methods. While discussions were made on model-free methods, Yannakakis and Togelius (2011) introduced that the gameplay data used is often comprised of some general features (such as time spent, performance) with many game-specific features.

Works introduced in this section achieved certain accuracies towards predicting player's preferences of choosing games and some also utilised the resultant models for PCG. However, similar to other works reviewed, their extractions of features are still game-specific in most cases. Even though Yannakakis and Togelius (2011) mentioned some features such as time spent or performance may be general across games, their availabilities are depending on the data collections of games. Until now, the studies that have been introduced aim to analyse player behaviours in games to improve the game content with machine-learning methods. Apart from this, in the game industry, a basic index for assessing the success of a game is the engagement of its players. The following sections investigate another important research area for player-disengagement prediction.

3.5 Player Disengagement Modelling

Player engagement, sometimes referred to as retention, is a measurement of how players engage with a game. As it is one of the most fundamental objectives game companies would like to achieve, it has become increasingly attractive in both research in the game industry and in academia. Most studies are focused on three different directions: i.e., player disengagement/churn prediction, play-time prediction and interest retention. Since this area is related to the basic benefits of game companies, several studies from both academia and industry have been published. This section discusses most of these works and their selections of data-representation methods.

Player-disengagement prediction generally aims at forecasting the result of a binary classification problem: i.e., is a player going to leave the game or not. A player-labelling method commonly used in this area is called *churn*. According to Runge et al. (2014), this term refers to players who have entirely stopped playing games. However, while being used, as Hadiji et al. (2014) suggested, instead of labelling players simply by observing whether he or she played the game on some day_0, a flexible window d_{rlx} is given in case this user returns in the next d_{churn} (e.g., seven) days. This is for solving the problem of bias, as there are many churners without a flexible window. At the same time, rather than filter out churners and take the rest of the non-churners (still highly biased), they attempt to consider only player behaviours made during a specific period before the current date (i.e., the test date). The length of this period was designed as two d_{churn} days (e.g., two weeks), and it takes the middle of this two d_{churn} days as the cut-off date. For example, if d_{churn} equals to seven, the length of the period to be considered is two weeks before the current date, and the cut-off date is the end of the first week. Based on the soft-labelling method mentioned before, the resultant class distribution is better balanced. However, according to their tables of class distribution, even after applying this approach, the percentage of positive examples still ranges from 0.883 to 0.998 for 14 of 20 cases–except for the remaining six cases in which the percentage of positive examples ranges from 0.564 to 0.618. While predicting churn in the game *Race Team Manager* (Bigbit-Ltd, 2014), a similarly bias was also observed. To solve it, another contribution (which is introduced in Chapter 7) is to come up with a new disengagement-labelling method that is able to maintain an approximately balanced distribution of resultant classes without losing any samples for predicting behaviour trends.

To discover reliable models, various approaches with different choices of data representations have been investigated for a range of games (Borbora et al., 2011; Borbora and Srivastava, 2012; Debeauvais et al., 2014; Drachen et al., 2016; Hadiji et al., 2014; Kawale et al., 2009; Runge et al., 2014; Tarng et al., 2009). In the modern gaming industry, the operation models of games range from free-to-play to subscription, because of this, the feature space that can be extracted from games for representing disengagement/churn may also vary. For example, Kawale et al. (2009); Runge et al. (2014); Weber et al. (2011b) selected a range of game-specific features to achieve a competitive predictive performance. In the work by Runge et al. (2014), their choices of attributes include *rounds of games played*, *performance accuracy* and *invites sent* which are measurements from a casual game. By representing players' disengagement with the number of rounds they played in the game, Weber et al. (2011b) converted the disengagement prediction into a regression problem. The choice of features applied in the work by Weber et al. (2011b) is similar but broader than that of Runge et al. (2014). Several game specific features from four aspects were included: i.e., *mode preference features*, *control usage features*, *performance features* and *playcalling features (manual or gameflow playcalling)*, etc. From a special point of view, Kawale et al. (2009) took social influences into consideration while trying to predict churn. The features they chose are different from those of Runge et al. (2014) and Weber et al. (2011b). During their experiment, they used undirected graphs to represent players' social relationships and then conducted some weighted calculations for extracting both group-engagement and social-influence features from the graph. The resultant features were utilised for creating predictive models with several different machine-learning algorithms. However, the results were not as competitive as other approaches. Game-specific data representations can also be found in the work by Borbora et al. (2011). In a study of the MMO game, *EverQuest II* (Entertainment, 2004), Borbora et al. (2011) utilised 14 game-specific motivational features such as *Number of quests participant in*, *Number of monster kills*, *Number of deaths* (etc.)

to predict churners. However, the following year, they started to realise the generality issue and came up with another work for upgrading this selection of in-game attributes into more comprehensive and general features (Borbora and Srivastava, 2012). Instead of relying directly on in-game attributes, they first applied a K-means algorithm for clustering both churner and non-churner players into groups. By comparing their observations over groups from two classes, they were able to determine churners' special behaviours in terms of their engagement, persistence and enthusiasm. All three terms are different functions of session-related features: i.e, *number of sessions per week*, *sum of session length per week* and *sum of inter-session length per week*. The selection of features in their work is close to that of Hadiji et al. (2014). Both studies stress the importance of session-related features. As for comparison, the features selected by Hadiji et al. (2014) are *number of sessions*, *number of days after installation*, *current absence time*, *average playtime per session*, *average time between sessions*, *playtime model parameters* (the parameters of player based power-law function), *retention value* (a function from average from days of play), *premium user flag*, *predefined spending category*, *number of purchases* and *average spending per session*. Similar to the work of Borbora and Srivastava (2012), most of these features are game-content irrelevant. However, their availability depends highly on how the game data is collected. This is because many features selected here are based on session and exact-time information, and such data is sometimes not available if game data is collected via some third-party data-collection frameworks. For example, Google Analytics is the choice for many game companies, and its framework uses an event-fired system (information is sent to the server when some predefined event happens in the game). Thus, to gain exact time information, developers have to include these time messages in some initializing-session and ending-session events. However, for mobile games or web games, because players can frequently return to the homepage or switch to different web-pages, the ending event is not fired for sending the required information. In addition, the number of days after a player's installation can only be easily gained if it was recorded as a constant event. Otherwise, it cannot be calculated if the available dataset does not contain the full history of players or the data collection was not integrated when the game launched.

Another general issue with this area is the requirement of data volume. Though these works achieved competitive results for predicting disengagement, according to their experimental settings, these experiments are able to achieve good accuracy only when more than half a year of players' history is available. According to Drachen et al. (2016), this is probably acceptable for some very successful MMO games but lacks values for newly released games or free-to-play games. In newly released games, gameplay history is usually short. Thus there will not be sufficient data available for conducting these experiments. In addition, the life cycles of free-to-play games are shorter and churning happens more often and earlier than in other types of games, as there is no loss if players quit. To deal with these practical issues, Drachen et al. (2016) tried to predict player disengagement/churn in an early stage by applying a heuristic-based decision tree. They tried to make the judgement by using features from very early-stage feature windows: i.e., a session, a day or a week from installations. For their study of the game *Jelly Splash* (Wooga-Games, 2013), for data representations, they selected attributes from both installation information (e.g., devices, geographic location), gameplay features (e.g., total days, total rounds), session information (e.g., average time between sessions) and game-specific information (e.g., average moves, average stars). This method can generate reasonable accuracy (ranges from 0.613 to 0.786, depending on the length of feature window) considering the fact that it relies on a short period after installation. In fact, to gain better prediction results, most works would use a longer but acceptable gameplay history: i.e. ranges from half a month to two months (Hadiji et al., 2014; Kawale et al., 2009; Runge et al., 2014; Weber et al., 2011b).

As introduced, many works show promise for predicting disengagement. However, like other research areas, a common problem that is that the features used in the experiments cannot be migrated into other games: i.e., they are not generic enough. This drawback may limit the business value of the studies if they are to be applied for commercial use. This is because, for every new game product, the feature space has to be redesigned for finding the available game-specific features. As Drachen et al. (2016) mention, free-to-play game companies–especially medium or small ones–cannot afford to spend much on data analytics. To constantly migrate existing data analytics products for every new product would be too difficult for them. This is one of the main problems this study aims to solve. Section 4.5.1 introduces a new data-representation method called 'event-frequency-based data representation', which can offer better generality. Based on this data-representation method, case studies are given for predicting disengagement in three different types of game. Details are given in Chapter 6. Apart from disengagement behaviours, yet another important factor in evaluating the successful operation of a game is in-game purchases. The next section discusses relevant studies of both predictions of player purchasing behaviours and revenue.

3.6 Player-purchase Modelling

In game data mining, a study area that is straightforwardly related to a games' profit concerns the prediction of players' payment behaviours and associated revenue. However, little research has been done in this area in recent years. Studies for predicting purchase are mainly focused on two aspects, i.e., purchase decision prediction (either from non-paying or paying users) and revenue prediction. In this section, both areas are going to be introduced with their corresponding studies.

The term *conversion rate* in games stands for the ratio of paying users to all players. By applying three standard algorithms (SVM, random forest and decision tree), both Pluskal and Šedivý (2014) and Sifa et al. (2015) studied how a non-paying user can be converted into a paying player following different strategies. In modern free-to-play games, a common strategy is to send offer information to players to encourage their desire to purchase. This type of information frequently appears at the end of a game round or a level upgrade. However, Pluskal and Šedivý (2014) claim that it is both annoying and inefficient to push advertisements to every player. Instead, they aim to predict whether a player will make a purchase after finishing each game level and only push advertisements to potentially paying users. Their choices of data representation include 13 different game-specific features (e.g., the number of matches, the number of trophies, etc.) over one recent month. Rather than treat this as a binary classification problem, they upgraded their prior model to a regression model which aims to predict the exact number of hours before a player is going to make a purchase after each game session (Pluskal and Šedivý, 2014). Based on the predictions, offer information can be delivered to players at the most efficient time. Instead of focusing on the suitable time for pushing offer notifications, Sifa et al. (2015) prefer to predict which players are likely to become premium (paying) players over the whole game lifetime (i.e., whether his/her first purchase will happen). According to Kim (2012), first purchase is an important concept, for once a user starts to pay, it is easier for him/her to make further purchases. This is why a similar predictive target was also included as a case in this study. The relevant experiments are further discussed in Chapter 5. Sifa et al. (2015) built both a classification model and a regression model. A classification model can be used to predict whether a player will make his/her purchase while a regression model can be used to estimate how many purchases a player will make. Their selection of data representation is broader than that of Pluskal and Šedivý (2014), as not only some game-specific features are included: Players' demographic and game-session related information is also taken into consideration. Alves et al. (2014) implemented a framework for predicting player revenue (after 180 days of gameplay) based on spending in the very early stage of playing (the first seven days). Using a large amount of data, they picked several hundred features to represent the dataset in their framework. These features came from six categories including appending action, gameplay, game-progression, social interaction, success metrics and game-setting preferences. Although some of these selected features can be found in other games, most are game specific.

Sifa et al. (2015), while trying to build the classification model, encountered another issue: high skew, as shown by the distribution between positive examples (paying users) and negative examples (non-paying). This is commonly seen in free-to-play games, as a biased dataset can lead to unreliable classifiers, thereby generating low accuracies for forecasting. To solve this problem, they applied a technology called the synthetic-minority over-sampling technique (SMOTE), which generates fake data in the minority class side for balancing the classification. This is one of the most widely used algorithms in different areas of data mining. However, it also brings some potential problems which will be further discussed in this research. The details are discussed in Chapter 7, which also considers possible solutions.

Like what has been discussed concerning player-disengagement prediction, data representations applied in this area also lack sufficient generality: i.e., most of the features are game specific and can hardly be immigrated to other games. The event-frequency-based data representation is used to provide promising prediction results with a better generality in Chapter 5. Another thing to note here is that, similar to what was mentioned in Section 3.5, bias can also occur in this research area. Unlike Hadiji et al. (2014), Sifa et al. (2015) use a technology called 'SMOTE' for balancing datasets for classification problems. As discussed, both approaches encounter limitations when faced with either a highly biased or highly dimensional dataset. An alternative solution that can deal with complex datasets from this work is explained in Chapter 7.

3.7 Generic Feature Extraction

Research works introduced so far have been investigating how player models can be built for a wide range of predictive purposes. However, as can be seen, although most selections of data representations differ, most nevertheless suffer from a common issue: i.e., the lack of generality. This limitation comes from either the features that are selected based on game contents or the features that are not tracked. It can be less risky when game-specific features are applied to their corresponding games. However, this problem quickly grows and brings extra costs when the data representation must be extended to work with other games. To solve this problem, this works has come up with a generic feature extraction approach named 'event frequency based data representation' that only relies on the counts of events that happened in games. The details of this generic data representation will be further explained in Chapter 4. Before then, in this section, some research works have been made to tackle the generality issue will be reviewed and will be compared to the method of this work.

There are generally three different ways that the state of the art research works attempted to provide methods with better generality, they are *relying only on general features* (Camilleri et al., 2017; Shaker et al., 2015; Togelius and Yannakakis, 2016), *transfer learning*(Shaker and Abou-Zleikha, 2016) and *general feature extraction*(Martínez and Yannakakis, 2011) respectively. The former two methods can be considered as feature engineering methods whereas the later one utilises feature mapping between games to port the existed model.

As the most straight-forward approach solution to the generality issue, works by Togelius and Yannakakis (2016), Camilleri et al. (2017) and Shaker et al. (2015) suggested generality can be achieved by modelling with only features that can be found across games. In the work by Togelius and Yannakakis (2016), they reviewed the generality limitation of the research works on game AI (e.g., Non-player-characters in games) and suggested that, the latest research should attempt to achieve three types of generality, i.e., *game generality*, *task generality* and *user/designer/player generality.* The *game generality* requires methods are generally applicable across different games; this is also the focus of the work presented in this thesis. To model behaviours in a general manner, general features such as *winning*, *losing*, *achieving rewards* and *progression or tension curves* are suggested to be used. Although the purposes are different, the features applied in the works by Shaker et al. (2015) and Camilleri et al. (2017) are similar to each other. While trying to build a model for predicting player experience, Shaker et al. (2015) firstly applied game-specific features to predict player experience in two different games (i.e., *Super Mario Bros* and a first-person shooter game: *Sauerbraten*) and then extracted features that can be commonly found between the two games, i.e., *The number of enemies*, *The number of enemies hit*, *the average time spent in each life* and *the number of rewards received.* While sharing some features used in the work by Shaker et al. (2015), Camilleri et al. (2017) further abstracted and expanded the feature space to include *goal-oriented/opposed events*, *distance travelled*, *time spent moving* and *time spent since start of the game (or including tutorial).*

Results from Shaker et al. (2015) and Camilleri et al. (2017) indicated that generic features selected across games can achieve high predictive accuracy. However, as mentioned by Shaker et al. (2015), a limitation of selecting common features as the data representation is that the availability of features are highly dependent on the genres of games. In other words, features found in football games may not be available in music games. Taking the games used in this research as examples (Samples of game data can be found in Table 4.1, 4.2 and 4.3), while using features suggested by Togelius and Yannakakis (2016), the progression or tension curves can only be found in *I Am Playr* because the other two games *Lyroke* and *Race Team Manager* are not tracking this information. Features suggested by Shaker et al. (2015) and Camilleri et al. (2017) are largely missing from the three games, for instance, *distance travelled*, *The number of enemies* and *The number of enemies hit* are only available in game *Race Team Manager*; *the average time spent in each life* and *time spent moving* do not match the concept of all three games; *time spent since start of the game (or including tutorial)* is not tracked in all three games. The methods applied in these previous works are different from this research, as in this research features are extracted from all the available events and, thereby, will not suffer from the availability issue.

To deal with the availability issue, transfer learning is also an alternative solution. Instead of finding common features across games, transfer learning aims at mapping feature space from a game to another so that the trained model can become applicable in a different one (Shaker and Abou-Zleikha, 2016). Although the work by Shaker and Abou-Zleikha (2016) is the only research that applied for player behaviour modelling, transfer learning approaches have been applied in several other areas for solving problems such as generality and lack of data Pan and Yang (2010). In the work by Shaker and Abou-Zleikha (2016), they applied both supervised transfer learning and unsupervised transfer learning to validate if transfer learning methods can ensure an acceptable performance in the games *Super Mario Bros* and *Sauerbraten*. In the supervised transfer learning scenario, model is trained in one game and migrated to another. In the unsupervised learning transfer learning, features of both games are projected into a third space via (Principle Content Analysis), the output of the trained model will be projected into that space and then transformed into the space of the other game. In both case, performance achieved are better than random guess which indicated that the transfer learning method did take use of the shared low-level information between the two games. Although the results achieved are promising, a general limitation with transfer learning is that the source and target needs to be related (Tan et al., 2015). In other words, the further the content of the game are away from each other, the lower amount of information might be carried over. Because of this, transfer learning may not be applicable in all cases for achieving better generality. Although transfer learning is not applied in this work, as a general method, it can also be applied upon the generic data representation method applied in this work for tacking problems where the dataset is small. Further information will be explained in the Section 8.3.

The work that is mostly close to this research is done by Martínez and Yannakakis (2011) who utilised sequence mining approaches for creating models for player-preference learning in games. Although the research area in their work is slightly different from player behaviour prediction, its purpose for applying the sequence mining method is also aimed at providing a generic data representation. Their sequence mining method first builds up the feature space by finding frequent events from three pre-defined event groups (i.e., *Performance Events*, *Navigation Events*, *Physiological Events*) and then merges them into higher-level patterns when those events happen within a time range. For each pattern to be used, the pattern also needs be supported by at least a specific number of appearances. Similar to the works done by Martínez et al. (2010); Pedersen et al. (2010); Yannakakis et al. (2009), an ANN model was applied to predict players' preference.

Comparing this feature extraction method to the event frequency based data representation (details in Section 4.5.1) proposed in this research, although both of them intend to model player behaviours based on basic events in games, there are two different aspects, they are:

- The sequence mining approaches applied by Martínez and Yannakakis (2011) extracts events from three pre-defined event groups (*Performance Events*, *Navigation Events*, *Physiological Events*) whereas the event frequency based data representation utilises all raw events from gameplay logs without known information.

- The sequence mining approaches focuses on learning sequence patterns where as the event frequency based data representation learns from the occurrences of the events only.

The first difference gives the event frequency based data representations better generality. However, given that all the raw data are included, it also will lead to higher dimensional models than the ones built by the sequence mining method which relies on pre-defined groups of events only. The second difference shows that the learning interests of both methods are not the same, this research focused on learning the player behaviours based on what and how many times player did. However, future research may also investigate if a sequence mining method on raw data can also help to work out important behaviour patterns in games. A possible research example is covered in Section 8.3.

The literature introduced in this section aim at generalise the methods for modelling player behaviour for predicting different purposes. Three categories of methods for increasing the generality of models have been introduced. They are all able to generalise the predictive models, but each of them still has limitations in different aspects. In summary, the methods that utilises only general features may only work on games of similar genres. Similarly, transfer learning works for mapping features across games, but the method assumes that games should share some low-level commons. The sequence mining method is the most generic one, but it was applied on pre-defined event categories instead of raw data, thereby, it may also suffer from the availability of events depending on how data is collected in games. Among these approaches, because the sequence mining approach is the closest one to the contribution of this research, and it was also compared to the event frequency based data representation. Both methods take different views on the directions for investigating player behaviours, in this research, the event frequency based data representation focuses on what players did and their frequencies instead of sequence patterns. But, in future research, other extensions of this data representation may be investigated as will be discussed in Section 8.3.

3.8 Summary

This chapter considers and revises several research areas in game data mining. As can be seen, data-mining technology has been used widely in this area for various predictive purposes. A common problem with most methods applied in player behaviour prediction is their abilities of being generalised. Although some attempts have been done in the preference learning area to generalise the player behaviour models, limitations of them have been reviewed in their corresponding sections. To deal with this issue, a more generic data representation will be proposed which builds feature space directly from the raw data. In the next chapter, this method will be described in details, and at the same time, a glance at the research method of work will be given.

Chapter 4

Player Modelling with Data Mining

Previous chapters discuss how general data-mining technology can be used to model various player behaviours and what areas existing works have investigated. Among them, player-purchase and disengaging-behaviour modelling have attracted much attention from both industrial and academic researchers. However, as explained in the previous chapter, the generality of existing data representations have prevented many otherwise successful methods from being widely applied. To solve this issue, a new data-representation method named *event frequency based data representation* is introduced as the main contribution of my research. This data-representation method aims to provide better generality than existing approaches while providing competitive performance.

This chapter begins by presenting an overview of my experiment. On the basis of this overview, details of experimental information–including game data sources, labelling methods, data representations, algorithms and evaluations–are explained. While data representations are introduced, the main contribution of this study–event frequency-based data representation–is explained in detail.

Main points in this chapter:

- a top-level blueprint of this research,
- introduction to the dataset from games that were used in this research,
- descriptions of labelling methods,
- introductions to data representations,
- explanation of the *event frequency based data representation*,
- introduction of classifier algorithms applied,
- summary of Evaluation methods used, and
- introduction to K-fold cross validation.

4.1 A Glance at The Research

This work is comprised of several experiments for different investigative purposes. Event-frequency-based data representation acts as its kernel. To understand how all the experiments were conducted and the relationships among them, an overview of the research structure is given in this section.

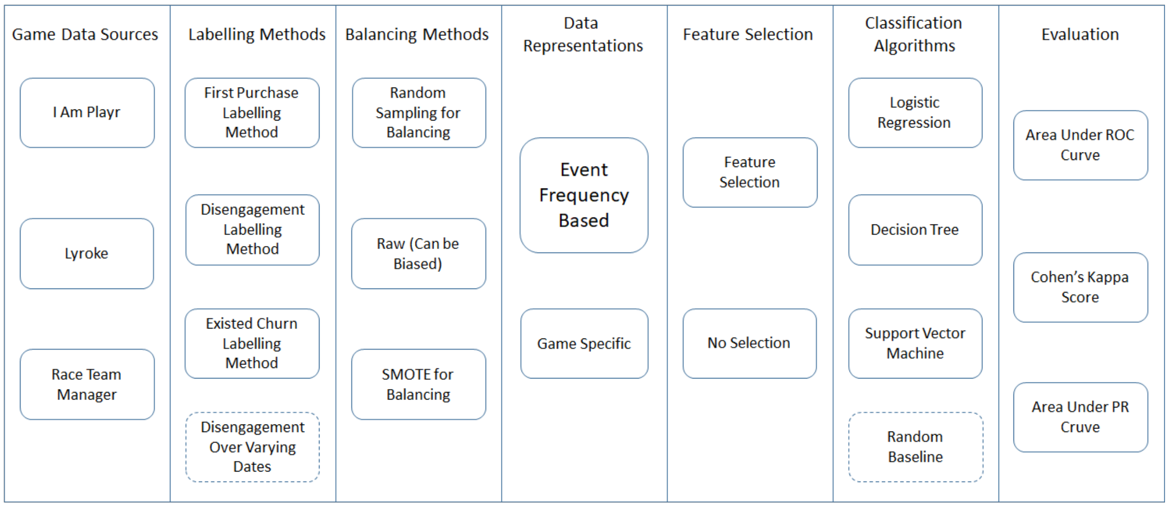

Figure 4.1: Structure of research experiments

See Figure 4.1. The top-level blueprint displays six main stages through which an experiment would proceed during the process of player modelling. These six stages are placed from left to right according to the order in which they are encountered in the procedure of modelling. As can be seen, the stages are *game data*, *labelling methods*, *balancing methods*, *data representations*, *feature selection* and *evaluation*. There are several choices in each stage which stand for different experiment settings for the corresponding stage. Therefore, by following the order from left to right, a complete experiment in this work is a path which links elements in different stages together. In the following chapters, which explain the details of the experiments, this figure is reused to show the current path being introduced. In the next several sections, information about each stage in the blueprint is introduced one stage after another.

4.2 Game-data Sources

This section introduces the game data that is used in our study. As Chapter 1 indicates, free-to-play games have developed dramatically in the last few decades, and player modelling has become increasingly popular in this industry. Therefore, data from these freemium games is of good value for research. Fortunately, by collaborating with *WeR Interactive* and *Bigbit Ltd.*, player-behaviour data from three commercial free-to-games was made available for investigation.

4.2.1 *I Am Playr*

Developed by *WeR Interactive*, *I Am Playr* (Figure 4.2)is a popular, commercial, first-person association-football game published free-to-play on *Facebook*, *IOS* and *Android*. During the game, players take control of a professional football player who is signed by a fictional football club named ‘River Park F.C.’. The basic content of this game is about simulating the player’s life, which includes playing matches, training and other entertaining events. During a match, the game is described by scrolling text until there is a chance to score a goal. Like usual free-to-play games, this game offers several items for purchase: for instance, shoes, cars and

Figure 4.2: Screenshot of I Am Playr

Table 4.1: *I Am Playr* Data Example

Attribute Name	**Descriptions**	**Examples in data**
Timestamp	The Unix Time	‘1388534450669’
s	Anonymous User ID	‘cb2cc1d11089db88’
v	Values related to action	‘1’
l	The week in game that player is currently in	‘3’
st1	Level 1 details of event	‘Item’
st2	Level 2 details of event	‘Equip’
st3	Level 3 details of event	‘Boots’
st4	Level 4 details of event	‘*IAmHelios*’

other luxuries. The data investigated contains gameplay logs of 89,057 players from January and February of 2014.

The data of *I Am Playr* was first hosted by an analytics-service provider named *Kontagent* (now *Upsight*) (Upsight, 2007) which then migrated to *Google Analytics* (Google, 2005). As introduced in Section 2.1.1, both of the providers apply event-firing based systems. For my experiment, the data was retrieved from APIs provided by both providers and parsed into an actionable format. An example of the format can be found in Table 4.1. The event described is that the player is equipped with an in-game item: a pair of boots called *IAmHelios*.

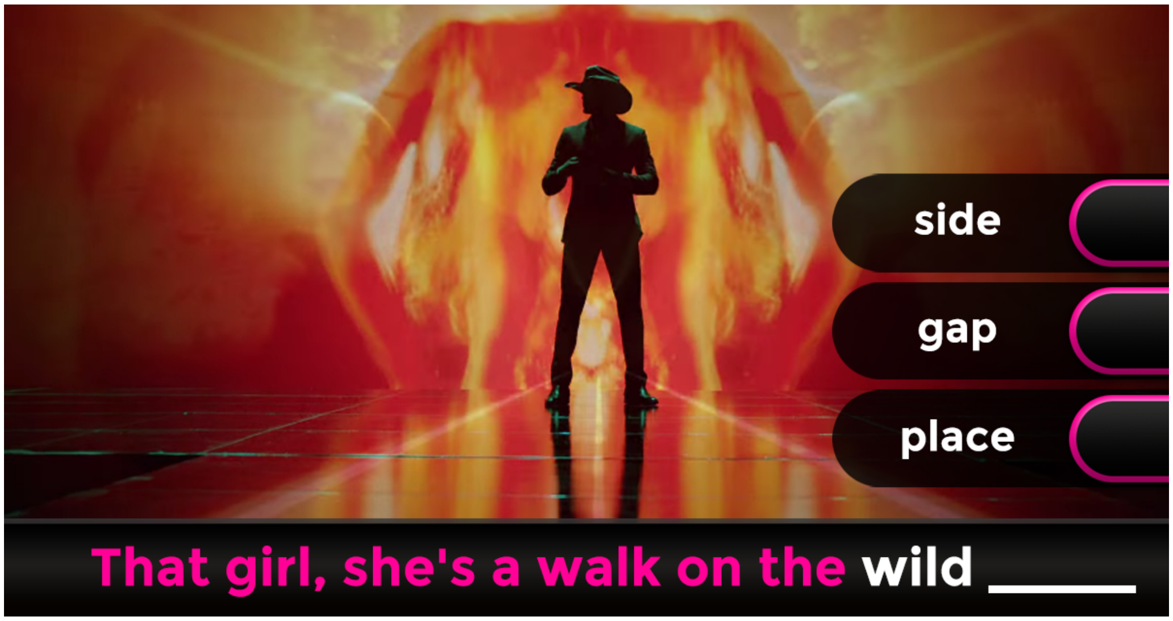

Figure 4.3: Screenshot of Lyroke

Table 4.2: *Lyroker* Data Example

Attribute Name	Descriptions	Examples in data
Time	The time	'2014-03-01-00-19-36'
s	Anonymous User ID	'551f72edc26e6f7e'
v	Values related to action	'1'
st1	Level 1 description of event	'Gameplay'
st2	Level 2 description of event	'IncorrectAnswers'
st3	Level 3 description of event	'TaintedLove'

4.2.2 *Lyroke*

Lyroke (Figure 4.3)is another commercial game investigated in experiments of this research. This game is also a quality product published by *WeR Interactive*. It is available on various platforms including *Facebook*, *IOS* and *Android*. Unlike I Am Playr, the main content of *Lyroke* concerns guessing missing lyrics while a song is being played. There are several modes. In tournament mode, players play a randomly selected song. In challenge mode, a player can directly challenge friends with songs. The in-game purchases include new songs and power-up items. Power-up items are used for reducing the game difficulty; for instance, a 'bomb' can directly display the answer, and a 'clock' can provide more time. The data investigated is gameplay logs of 280,338 players from March and April of 2014.

Like the *I Am Playr* dataset, the dataset used from *Lyroke* was first stored in *Upsight* and then moved to *Google Analytics*. Likewise, the data was retrieved from provided APIs and parsed into actionable formats. Table 4.2 shows an example of the ready-to-use format, which means the player selected a wrong answer to the current challenge for the song 'Tainted Love'.

Figure 4.4: Screenshot of Race Team Manager

Table 4.3: *Race Team Manager* Data Example

Attribute Name	**Descriptions**	**Examples in data**
User ID	Anonymous User ID	'0E61685B-75D5-42A5-A4CA-DC905BF055E8'
Country	The country of the user	'United Kingdom'
Time Stamp	The date of current event	'18/09/2015'
App version	The app version running	'2.3'
EventCategory	Description of event category	'Race'
EventLabel	Description of event info	'CornerOvertake'
EventAction	Description of event action	'Succeed'
EventValue	Values related to event	'0'

4.2.3 *Race Team Manager*

Race Team Manager (Figure 4.4) is a free-to-play game developed by Bit Ltd. It is available across all mobile platforms. The game was picked as the 'editor's choice' after its first launch on the *App Store*. Its gameplay allows players to take the role of the manager of a team who can control how racing cars should drive to pass, avoid collisions, reduce tire-replacing time and adjust driving styles. This experiment used full gameplay logs of 113,872 players between October 2015 and January of 2016.

The data from this game was collected by *Google Analytics*. Through the API, data was retrieved and parsed into a similar format, which was used for *WeR Interactive* games for easier pre-processing. An example event in the game can be found in Table 4.3. This example shows that a player successfully performed a pass around the corner in the race.

4.3 Labelling Methods

The second stage in player modelling is labelling. Labelling is essential to binary classification problems which categorises data samples into two groups for different predictive purposes: e.g., players need to be labelled as disengaging and non-disengaging players in a disengagement-prediction task. Therefore, this stage is decided by predictive purposes. For instance, if used to predict players' disengaging behaviours, the labelling method named 'churn' can be a good candidate. In this work, disengagement and first-purchase prediction act as the main predictive targets. Therefore, labelling methods related to these purposes are applied. In addition to the existing labelling methods, three new labelling methods which aim to solve either disengagement or first-purchase related problems are introduced in this work. Details of the labelling methods are given in corresponding chapters.

4.4 Balancing Methods

Of all the predictive tasks discussed in this work, one notable issue is that the datasets which result from any labelling method are often biased. For example, in a first-purchase prediction problem, players are labelled according to whether they have made any purchases in the past. It often happens that most users are non-paying ones, such that only a small number of people can be paying players. In machine learning, the training of classifiers in biased situations may lead to unexpected performances. According to López et al. (2013), the impact mainly comes from two aspects. First, when a dataset is biased, it will be difficult to identify the minor class correctly because some exceptional and significant cases might exist in the minor class that are not even covered by the training data. Second, most classification algorithms have been designed for balanced datasets. Because of this, they are not necessarily able to provide reliable performances. Once the frequency of examples from the minor class is small enough, algorithms will tend to ignore these examples and classify all of them as the major class to achieve better prediction results.

Therefore, some extra efforts have been made to generate balanced classification problems. Two widely used methods including random under-sampling and SMOTE. These methods are applied to balance the distribution between classes generated by different labelling methods. Because of some existing limitations with these balancing methods, this work offers a new labelling method for generating balanced classes for disengagement prediction. Its details are discussed in Chapter 7.

4.5 Data Representation

The next stage in the blueprint is data representation. Several methods for representing the data have been reviewed in Chapter 3. However, as explained, most of the current methods involve a lack of the generality such that they cannot be applied across games. This section introduces, as the main contribution of this research, a new data-representation method named event-frequency-based data representation.

4.5.1 Event Frequency-based data representation

The limitation on generality in game metrics mostly comes from two aspects: game-specific and availability. The first aspect is easy to understand: I.e., a data representation that is

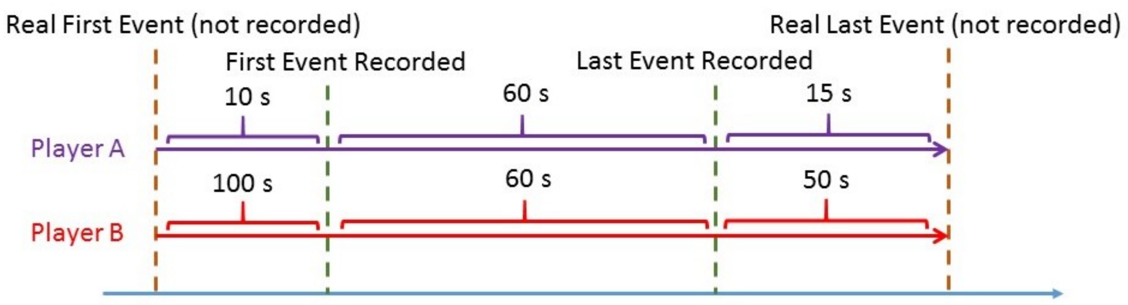

Figure 4.5: Problem for calculating session length

The game-session lengths of players A and B are different. However, since the time between the first and last recorded events are the same, their game-session lengths will be considered the same.

formed by game-content features can hardly be applied to a game with different content– especially for games in other genres. For instance, player behaviours in football games can be very different those in a music game. The second reason is mostly based on the design of the data-collection system of a game. For example, *the length of a game session* is reasonably generic across games (Hadiji et al., 2014), but this may not be collected if the data-collection system is not designed to handle it. When session information is not directly available, an alternative way to calculate the session length is to identify the difference between the ‘last’ and the ‘first’ event in the same session. However, this may become unreliable when the example shown in Figure 4.5 happens.

Thus, to solve both problems, a new data-representation method needs to be both game-content irrelevant and able to take whatever game event is available. Although many efforts have been made to achieve various predictive purposes (e.g., disengagement and purchase) in the area of game data mining, as discussed in Section 3.5 and 3.6, the most widely investigated approaches are unable to provide data-representation methods which are generic enough to be migrated to different games without adaptation. To cope with this issue, a new generic data-representation method is introduced as the main contribution of my research in this section.

A possible solution is to use counts of the appearance of each event to represent the dataset for individual players. This is inspired by a similar model widely used in text mining (Zhang et al., 2010). The model is called ‘bag of words’: Words in an article are enumerated and considered a single dimension (feature) (Cormack, 2007). This approach makes sense, because the frequency of words can reflect some information about the whole article. Likewise, in the context of games, it is possible that the frequency of events a player performs or experiences can also hide valuable patterns. Furthermore, the use of event-frequency can provide good enough generality, as it is content irrelevant and can use any event that happens in the game.

To verify this conjecture, the main contribution of this work –event-frequency based data representation– relies only on the number of occurrence of events in a game. Table 4.5 gives a subset example of the feature space that was built in the game I Am Playr while predicting players’ disengagement behaviours. In this example, the three randomly selected events are “LevelUpOffer-MissOut–Unset”, “Video-Played–MD10a-playrmp4” and “Player-Trophy-UnlockItem-IAmTyphon”. Their definitions can be found in Table 4.4. Taking *Player 1* as an example, he/she missed 10 times of the level-up offer named ‘unset’, has

Table 4.4: Example event explanation

Event Name	Explanation
LevelUpOffer-MissOut–Unset	The number of times that player missed a level up offer called 'unset'
Video-Played–MD10a-playrmp4	The number of times that player played a video (generally when a milestone is reached)
Player-Trophy-UnlockItem-IAmTyphon	The number of times that player unlocked a trophy named 'IAmTyphon'

Table 4.5: Event Frequency Data Representation

	LevelUpOffer-MissOut–Unset	**Video-Played–MD10a-playrmp4**	**Player-Trophy-UnlockItem-IAmTyphon**
Player 1	10	1	1
Player 2	9	1	1
Player 3	8	1	0
Player 4	3	1	1
Player 5	9	0	1

played the video 'MD10a' and has unlocked the trophy 'IAMTyphon'. As only the counts of the events are used, their actual meanings become less important in this data representation. This is also why this data-representation method can be applied across different games for multiple predictive purposes. Therefore, the hypothesis of my research is as follows:

Event-frequency-based data representation can be used to predict player behaviour with supervised learning to provide a significantly better performance than random guess and competitive performance while being compared to other state-of-the-art methods, where applicable.

In this hypothesis, to be more precise, "Method A provides a significantly better performance than Method B' represents the situation where, in all cases of an experiment, the *p-values* resulting from two-tailed t-tests conducted between A and B are less than 0.01, as well as the *t-values* and *effect-sizes* are both positive. Instead, "Method A provides a competitive performance while being compared with Method B" stands for the situation that, in most cases of an experiment, A can either provide significantly better performance than B or there is no significant difference (*p-value* is larger than 0.01) found between A and B.

4.5.2 Game Specific Data Representation

To compare event-frequency-based data representation for predicting players' disengagement behaviours, this work also implements a game-specific data-representation method introduced by Runge et al. (2014). The details of this data-representation method are introduced in Chapter 6.

As for predicting players' first purchases, to the best of my knowledge, only a few works (Sifa et al., 2015) have aimed at the same predictive purpose. Unfortunately, most of the features used in the data representation are not available in the game datasets used in this study. Therefore, to predict players' first-purchase behaviours, a random classifier was used as the baseline for comparison with event frequency-based data representation.

4.6 Feature Selection

Though event-frequency-based data representation is able to provide good generality thanks to its features, it has its limitations. Highly dimensionality is the largest issue with this data-representation method, as there are often thousands of events (features) to be analysed simultaneously in a good game. For example, in the game I Am Playr, 2,635 unique events are experienced or generated by players. High-dimensional data can have two negative impacts: i.e., an extra-complex model, and overfitting (Guyon and Elisseeff, 2003). An extra-complex model cannot be understood by human beings; therefore, it is difficult to visualise and analyse an extra-complex model. The latter impact, overfitting, is a general issue in machine learning that often affects the reliability of predictive models. A detailed explanation of overfitting is given in Section 4.9. To solve these problems, dimensional reduction, introduced in Section 2.1.6, can act as the solution. Feature selection is extensively used in dimensional reduction to decrease the dimension of the represented dataset by keeping only the most relevant features (relative to the current predictive target). In this work, a tree-based ensemble method called *random forest* was applied to all cases for computing the importances of features and helping to determine the most relevant ones. This algorithm has been widely applied in machine learning for feature selection in various areas (Genuer et al., 2010).

In addition to the basic statistical methods introduced in Section 2.1.6, another group of widely used feature-selection methods are tree-based ensemble methods. This group includes methods such as *tree-bagging* and *random forest*. Unlike the approaches introduced before, these methods are machine-learning algorithms which are also capable of solving general classification problems. Among them, tree-bagging is most naive tree-based ensemble approach. During its construction procedure, it simply trains several decision-tree classifiers (introduced in Section 2.1.6) on *random samples with replacement (which means that, examples can be selected more than once)* from the given training set Breiman (1996). As for the classification, for an unseen example, it votes among predictions from all separate classifiers and uses the major choice. When training separate decision trees, rather than select the next splitting node from all feature candidates, a random forest classifier selects only from a subset of all feature candidates (called feature bagging) Breiman (2001). This is to solve the problem that some important features will be overly selected in most decision trees to be aggregated in the tree-bagging method.

After pre-processing, the importances of each feature are ranked and normalised according to their criteria (e.g., 'gini' and 'information gain'), as decided by the trees, so that the more valuable features can be easily extracted by giving a threshold. In this work, the threshold is set to be the mean of all features' importances: That is, the attributes whose importance score is larger than the mean value of all are selected. The implementation of random forest is based on the scikit-learn package in Python 2.7.

4.7 Classification Algorithms

In this work, for easier comparison with other works, classification algorithms such as decision tree, logistic regression and SVM are used as the main classifiers. These classifiers are all

Table 4.6: Parameter Selection Ranges for Algorithms Applied

	Parameters	**Selection Range**
Decision Tree	criterion	entropy, gini
	min_samples_split	random integer from [2, 20]
	min_samples_leaf	random integer from [2, 20]
Logistic Regression	C	random float from an exponential probability density distribution with a scale (1/lambda) of 100
Support Vector Machine	kernel	(Gaussian) radial basis function kernel
	C	random float from an exponential probability density distribution with a scale (1/lambda) of 100
	gamma	random float from an exponential probability density distribution with a scale (1/lambda) of 0.05

widely applied in the game-data mining area for several predictive purposes. For instance, they are used in research work including but not limited to Ahmad et al. (2009); Borbora et al. (2011); Drachen et al. (2016); Hadiji et al. (2014); Kang et al. (2012); Lee et al. (2014); Runge et al. (2014); Woo et al. (2012). The definitions of these algorithms were introduced in Section 2.2.1. Because these algorithms make predictions based on some main hyper-parameters. The pre-defined ranges of the parameters can be found in Table 4.6. During the training process, in each round of the search, a set of parameters is randomly taken within the given range, and three-fold cross validation is used for evaluating the quality of the classifier with the current selection of parameters. The content of k-fold cross-validation is introduced in Section 4.9 with explanation, as needed.

All algorithms and the random search applied in the experiments are based on implementations provided by a *Python (version 2.7)* library called *Scikit-learn (version 0.18)* (Pedregosa et al., 2011).

4.8 Evaluation Methods

This section describes how the experiments conducted in this work were evaluated. In this first part, the confusion matrix–which provides the basis of all applied measurements–is given in Table 4.7.

4.8.1 Confusion Matrix

The evaluation methods applied in my study are *ROC (receiver operating characteristic)*, *PRC (precision-recall-curve)* and *Cohen's kappa*. Since all measurements are based on the concept of the confusion matrix, some explanations about the matrix and its related low-level measurements are introduced in this section.

A confusion matrix can be considered a display of results from a classifier which is able to show the difference between predicted and actual classes (Visa et al., 2011). An ordinary confusion matrix is comprised of four measurements. Their definitions are offered in Table 4.7. In this example, there are 50 positive and 50 negative examples in the dataset. As can

Table 4.7: Confusion Table

	Predicted Positive	**Predicted Negative**
Actual Positive	40 (True Positives)	10 (False Negatives)
Actual Negative	13 (False Positives)	37 (True Negatives)

be seen from the example, the 40 samples which are correctly categorised as positive are called *true positives*; the rest of the ten samples, which are wrongly labelled as negative, are called *false negatives*. Likewise, the 37 samples which were correctly labelled as negative are *true negatives*, and the rest of the 13 samples are *false positives*.

Three commonly used low-level measurements can be calculated from this matrix: *precision*, *recall (true-positive rate)* and the *false-positive rate*. Their formulas can be found from Equation Group 4.1. As can be seen, precision and recall focus on displaying the classifier's ability towards predicting positive samples only whereas the false positive rate also considers whether this classifier wrongly predicts many negative samples as positive. These measurements usually act as important components in top-level evaluation methods that are widely used.

$$\begin{aligned}
&\text{Precision} = \tfrac{True\ Positives}{True\ Positives + False\ Positives}\\
&\text{Recall (True Positive Rate)} = \tfrac{True\ Positives}{True\ Positives + False\ Negatives}\\
&\text{False Positives Rate} = \tfrac{False\ Positives}{False\ Positives + True\ Negatives}
\end{aligned} \tag{4.1}$$

4.8.2 Top-level Measurements

In machine learning, the top-level evaluation methods are often chosen for various purposes. Among all widely used measurements, accuracy (shown in Equation 4.2) is the most basic and best known. It measures the ratio of correctly classified samples (both positive or negative) in all samples. It is able to give a reflection of how a classifier performs globally, but it is less informative in several cases.

One of the commonly seen problems is that, when the data is biased, accuracy is not able to reflect the real performance of a classifier. For example, in a player-purchasing prediction problem, if 90% of players are non-paying users, though the resultant accuracy can be quite high, a less meaningful classifier may have been generated. This is because classifiers are able to reach high accuracy simply by labelling every sample non-paying users, though this is generally not acceptable. In problems like this, an accuracy of 70% with an expected accuracy (random guess) of 40% is generally better than an accuracy of 80% with an expected accuracy of 70%. To correctly show the performance of classifications in a biased situation, a widely used statistical test called Cohen's kappa is introduced in this work to achieve an 'upgraded accuracy'. Originally proposed for a different purpose, Cohen's kappa was first introduced by Cohen as a measurement for calculating inter-raters agreements (Cohen, 1960). A Cohen's kappa score k ranges within $[-1, 1]$. A positive k indicates that two observers agree with each other by the degree of k, whereas on the contrary, observers disagree with each other by the degree of k. In the case of classification, suppose that the actual classes of instances are taken as one observer while the predicted classes of them are another observer. Thus, the calculation of agreement between these two observers can be considered the performance measurement of the model. Cohen's kappa can be imagined simply as the performance normalised by its distribution baselines. Cohen's kappa comprises two parameters: p_o and p_e, where p_o is the accuracy and p_e is the random-guess baseline based on its data distribution. The random-guess baseline p_e normalises the accuracy p_o on imbalanced datasets. Because a Cohen's kappa score measures how the classification performance is better than a random guess, its score will always be 0 for a random classifier. Equation 4.3 shows how it is calculated, *ap*, *pp*, *an* and *pn* stand for 'actual positive proportion', 'predicted positive proportion', 'actual negative proportion' and 'predicted negative proportion', respectively. The variables *tp*, *fn* *fp* and *tn* here stand for the number of 'true positives', 'false negatives', 'false positives' and 'true negatives', respectively.

Accuracy

$$Accuracy = \frac{tp + tn}{tp + fp + tn + fn} \tag{4.2}$$

Cohen's Kappa

$$\begin{aligned}
&total = tp + tn + fp + fn \\
&positive_{actual} = \frac{tp + fn}{total} \\
&positive_{predicted} = \frac{tp + fp}{total} \\
&negative_{actual} = \frac{fp + tn}{total} \\
&negative_{predicted} = \frac{fn + tn}{total} \\
&\boldsymbol{p_e} = positive_{actual} \cdot positive_{predicted} \\
&+negative_{actual} \cdot negative_{predicted} \\
&\boldsymbol{p_o} = \frac{tp + tn}{total} \\
&\boldsymbol{k} = \frac{p_o - p_e}{1 - p_e}
\end{aligned} \tag{4.3}$$

Although Cohen's kappa is able to provide a more reliable accuracy, it does not distinguish the performance for predicting samples from either class. Because of this, it is less informative when the positive and negative classes do not share the same importance; thus, some trade-offs need to be made between them. In other words, given a high Cohen's kappa score, it is difficult to determine whether the classifier is good at predicting one class only (the major class in the biased situation) or has an average performance for predicting both sides. However, in practical problems, it often appears that only one class is more important; this class may often change in different periods. For example, in the churn prediction problem, while the game is in its first launching period, it is generally more important to retain new supporters. During this period, the accuracy measure for predicting non-churners is generally more helpful. On the contrary, after the game has been operated for a while and has gained some loyal supporters, it would be more helpful to predict the churners more accurately to not lose the existing players. In this case, the AUPRC (area under precision-recall curve) would be a better candidate (Saito and Rehmsmeier, 2015). A precision-recall (PR) curve is formed by the precisions and recalls calculated from confusion matrices (introduced in Section 4.8.1)

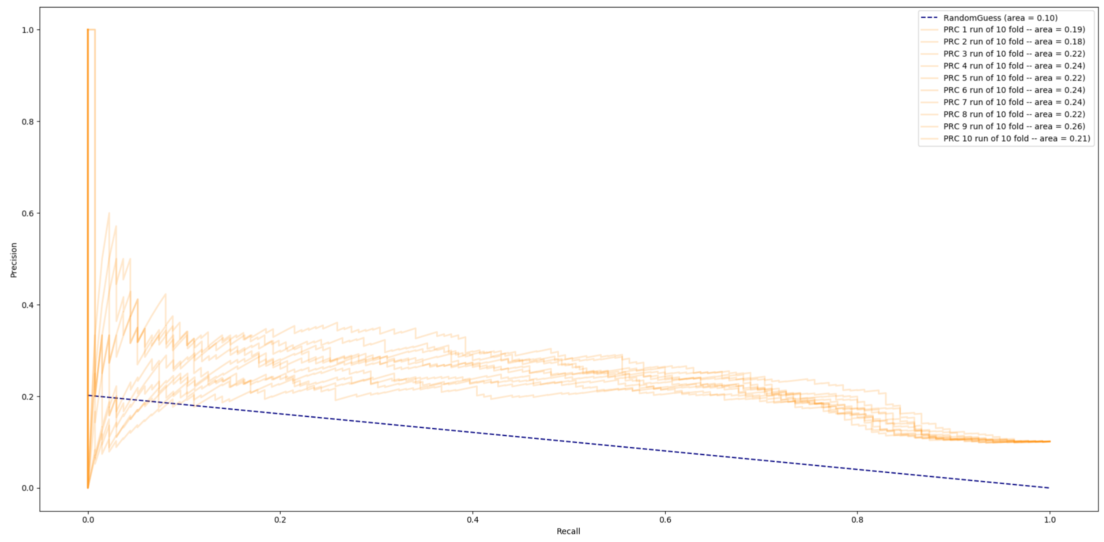

Figure 4.6: PRC curves for 10-fold cross validation while predicting disengagement with the event frequency based data representation

while varying the internal cut-off decision thresholds used by classifier algorithms (Davis and Goadrich, 2006). Area under the curve is often used as the performance of the classifier. The maximal area (perfect classifier) under a PR curve is 1. The cut-off decision threshold is a standard probability used in the algorithm for deciding whether a sample should be positive or negative. For example, in a churn prediction problem, for a sample to be predicted when the score generated by an algorithm is 0.6, the classifier may label the sample as positive, because 0.6 is larger than 0.5. In this case, 0.5 is the cut-off decision threshold we used here. However, 0.5 may not be the perfect number in all cases. Sometimes a trade-off will be made to ensure that the classifier is more serious to make a decision to label a sample as positive. For this purpose, the threshold should be increased. An example of a group of PR-curves can be found in Figure 4.6 where each PR curve shows the performance of a classifier trained in each fold of the 10-fold cross validation while predicting disengagement in I Am Playr. Each point that comprises the curve represents a cut-off decision threshold. Because the PR curve is comprised of precisions and recalls, it is a good indicator for the performance of predicting positive samples, but for the same reason, the PR curve is correlated with the class distribution of the dataset, and its area of a random baseline curve increases along with the proportion of positive samples in the training dataset. In this work, while the predictive task is imbalanced, the percentage of positive examples is used in all examples as the area under the PR curve of a random classifier. This is a commonly used approximation suggested by both Keilwagen et al. (2014) and Saito and Rehmsmeier (2015).

Though AUPRC can be reliably applied for measuring performance where the positive class is more important, it is not a good candidate for measuring the comprehensive performance of the classier for predicting both classes (Davis and Goadrich, 2006)–unless it was applied twice by exchanging the positive and negative classes. In this work, to show the performance of the proposed event-frequency-based data representation, evaluations of AUPRC for taking both classes as the positive class are shown for each experiment. Cohen's kappa works better in this case, as it shows a single score which does not distinguish either class. However, as explained by Jeni et al. (2013), Cohen's kappa is affected by a dataset

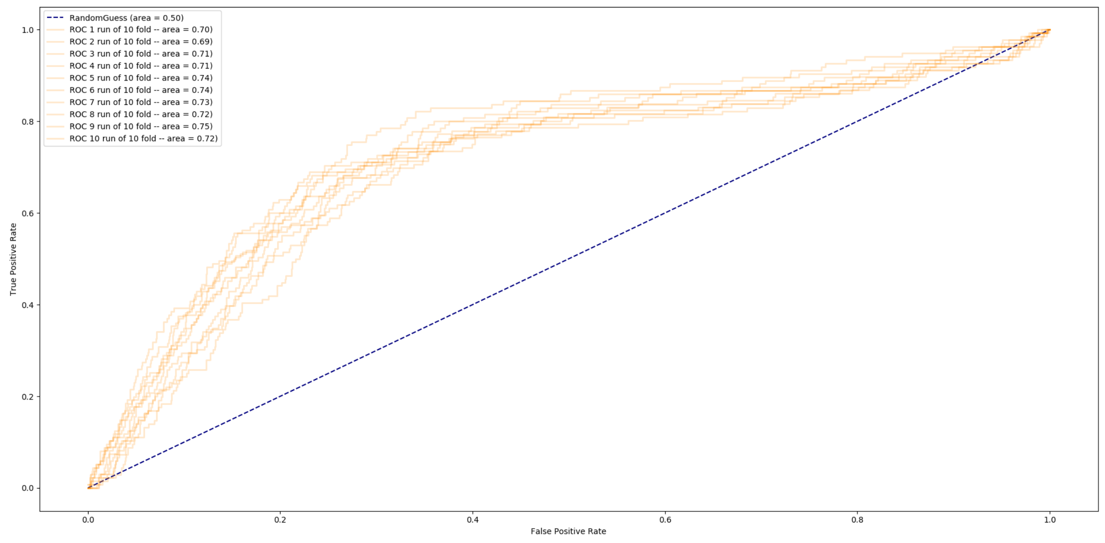

Figure 4.7: ROC curve for 10-fold cross validation while predicting disengagement with the event frequency based data representation

that is biased on either side. Therefore, it is not good at reflecting a classifiers' performance in imbalanced situations. To show a comprehensive measurement of the performance of classifiers for predicting both classes in a manner that is not affected by biased situations, the AUROC (area under receiver operating characteristic curve) often turns out to be an optimal choice. A ROC curve is formed by true-positive and false-positive rates, where the area under this curve comprehensively measures the general performance of this classifier under different cases. An example of a group of ROC curves can be found in Figure 4.7 where there are 10 ROC curves generated from 10-fold cross validation for predicting players' disengagement behaviours in I Am Playr. The maximal area (perfect classifier) under a ROC curve is equal to one. Because the true-positive rate focuses only on the correct prediction of positives for all positive samples, and because the false-positive rate concerns only the wrong prediction of positives for all negative samples, both of the measurements are 0.5 for a random classifier. Therefore, the AUROC of a random classifier is always equal to 0.5 which does not change with the skew of the dataset. However, because AUROC is not sensitive to the data distribution, it may not show a significant difference for the same classifier being tested under experiments conducted under different data distributions (Jeni et al., 2013; Saito and Rehmsmeier, 2015). In this case, measurements such as AUPRC and Cohen's kappa can help to determine the difference (Davis and Goadrich, 2006). Table 4.8 gives a brief summary of all three measurements.

4.9 Overfitting and K-fold Cross-validation

Overfitting is a common issue in machine learning which can seriously affect the reliability of models built. It often leads to a trained model that works great on the training dataset but does not have the generality needed to solve any unseen problems. Figure 4.8 shows a simple regression problem which can help to explain the problem. In this figure, black dots are training data that classifiers see during the training of the model; white dots are testing data unknown while training. In a regression problem, the task is to find a line which can

Table 4.8: A brief summary of measurements being applied in this work

Measurement Name	**Imbalanced Data Impact**	**Random Classifier Performance**
Area Under PR curve	Yes, performance is positively correlated with proportion with positive examples	Can be approximated as ratio of positive examples
Cohen's Kappa	Yes, performance decreases with bias to any class	0.0
Area Under ROC curve	No	0.5

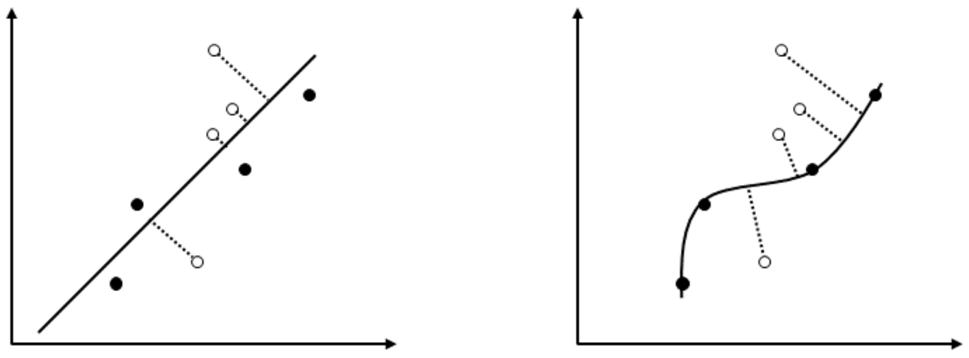

Figure 4.8: Overfitting Problem

minimise the sum of distances from all data points to the line. The left sub-figure uses a linear (less complex) model to fit in the data points; the right one uses a non-linear (more complex) model. By looking at the training data (black dots) only, the more complex model is able to pass all training-data points and has no error whereas the simpler model is worse. However, when both models are used to predict unseen data points (white dots), the simpler model has a lower error than the complex one. In this case, the complex model is said to be affected by overfitting. Therefore, as can be seen from this example, more complex (higher dimensional) models are more likely to face an overfitting issue, as they can cover more cases in training examples including these outliers. However, for the same reason, in a larger dataset (the ratio between 'normal data' and outliers is generally higher), more complex models can generally give better predictions.

Overfitting happens when machine-learning algorithms capture both needed information and misleading random noises at the same time (Lee et al., 2006). As has been discussed in the above example, this situation often appears when the data is not quantitatively sufficient compared to the complexity of the model to be trained. This is because the lack of data points can confuse the classifier about what the 'normal' data points are and what the outliers are, as their quantities are small. Assuming that, in an extreme situation in which the amount of real information is the same as the outliers, an algorithm may hardly decide which is the best path to follow. For this reason, the resultant classifier is able to display a good performance only in the training set; it shows bad generality when facing other unseen samples. In other words, the performance shown by the classifiers trained in this case is unreliable.

To get rid of the overfitting impact and show the reliable performance, K-fold cross-validation is a widely used method of evaluation (Arlot et al., 2010). This method splits the whole dataset into K pieces, and repeats similar experiments K times on different training and testing sets. The algorithm it follows can be found in Algorithm 1. Notice that, in each repeated experiment, the training set and testing set are independent of each other: That is, the classifiers trained are always tested by unseen samples. In this work, a commonly used 10-fold cross-validation was applied to every experiment.

Algorithm 1 K–Fold Cross Validation

```
1: procedure K–Fold Cross Validation
2:     Split Dataset into $k$ pieces
3:     $resultSet = \{ \}$
4:     for each piece $i \in k$ do
5:         $testingSet = i$, $trainingSet = k \setminus i$
6:         Train classifier with $trainingSet$
7:         Test classifier with $testingSet$ and store in $currentResult$
8:         $resultSet = resultSet \cup currentResult$
9:     end for
10:    Calculate averaged result from elements in $resultSet$
11: end procedure
```

As was introduced in Section 4.6, several experiments in this work use feature selection as an essential part. In this case, a special modification of K-fold cross-validation has to be made. Since all experiments were performed with 10-fold cross-validation, feature selection should be applied for each of the ten repeated experiments separately instead rather than to the whole dataset. This is because, if the feature selection was done before splitting, all data points would attend the selection, during this process, some data which only meant to be known by the testing set after splitting will also be included by mistake. This may break the idea of k-fold cross-validation, which tries to keep the testing set unseen from the view of the training set at any sub-experiment and can sometimes cause bias. Therefore, Algorithm 1 for k-fold cross-validation was modified to Algorithm 2 when feature selection was enabled.

4.10 Summary

This chapter provides an overview of the experiments that were conducted in this work. There are seven stages to each experiment: *game data sources*, *labelling methods*, *balancing methods*, *data representations*, *feature selection*, *classification algorithms* and *evaluation methods*. The basic information has been introduced respectively for each stage of an experiment. In summary, Algorithm 3 depicts how classifiers are trained in the experiments of this work. In this algorithm, Training Set X is ready for the training process, which has been processed by any data representation, and the label Y is chosen based on different labelling methods. Based on the experimental procedure introduced here, the case studies conducted

Algorithm 2 K–Fold Cross Validation with Feature Selection

```
1: procedure K–Fold Cross Validation with Feature Selection
2:     Split Dataset into $k$ pieces
3:     $resultSet = \{ \}$
4:     for each piece $i \in k$ do
5:         $testingSet = i$, $trainingSet = k \setminus i$
6:         Get $selectedFeatureSet$ via Feature selection based on $trainingSet$
7:         Filter $trainingSet$ to $trainingSetSelected$ by only keeping $selectedFeatureSet$
8:         Filter $testingSet$ to $testingSetSelected$ by only keeping $selectedFeatureSet$
9:         Train classifier with $trainingSetSelected$
10:        Test classifier with $testingSetSelected$ and store in $currentResult$
11:        $resultSet = resultSet \cup currentResult$
12:    end for
13:    Calculate averaged result from elements in $resultSet$
14: end procedure
```

Algorithm 3 The Simplified Complete Modelling Process

```
1: procedure Train(RawData R)
2:     Get ready-to-use Dataset X by selected data representation from R
3:     Get label list Label Y with selected labelling methods from R
4:     Split Dataset X and Label Y into $k$ pieces
5:     $resultSet = \{ \}$
6:     for each piece $i \in k$ do
7:         $testingSet = i$, $trainingSet = k \setminus i$
8:         Get $selectedFeatureSet$ via Feature selection based on $trainingSet$
9:         Filter $trainingSet$ to $trainingSetSelected$ by only keeping $selectedFeatureSet$
10:        Filter $testingSet$ to $testingSetSelected$ by only keeping $selectedFeatureSet$
11:        Apply balancing method in $trainingSetSelected$
12:        Select the algorithm as the training classifier
13:        Perform random search (3 Fold Cross Validation) on $trainingSetSelected$ to
           find best hyper-parameters for the classifier
14:        Set the classifier with the best hyper-parameters
15:        Train the classifier with $trainingSetSelected$
16:        Test the classifier with $testingSetSelected$ and store in $currentResult$
17:        $resultSet = resultSet \cup currentResult$
18:    end for
19:    Calculate averaged result from elements in $resultSet$
20: end procedure
```

in the following chapters with event frequency-based data representation for predicting different purposes will start to be discussed. In the next chapter, the case studies start with a popular predictive target, first purchase. Experiments conducted for predicting first purchases will help to investigate both the generality and performance that can be brought by applying event-frequency-based data representation.

Chapter 5

Predicting First Purchase

The previous chapter introduced the basic approach to my experiments. From this chapter onwards, experiments performed to investigate both the generality and performance of event-frequency-based data representations are discussed. As was introduced in Chapter 3, revenue-related predictive purposes have attracted many studies in the game-analytics area. In this chapter, as one of the most important revenue related predictive purpose, the player's first-purchasing behaviour is used as the predictive target. The first part of this chapter reviews the state-of-the-art in this area. Next, the results from the experiment of predicting first purchases with event-frequency-based data representation are displayed and discussed. As mentioned in Section 4.6, since event-frequency-based data representation is a highly dimensional approach, feature selection is added for dimensional reduction to see if a less-complex model can be generated without losing significant accuracy.

Main points in this chapter:

- ◆ introduction to the problem of first purchase
- ◆ case studies for predicting first purchase, and
- ◆ experiments for investigating the effect caused by feature selection.

5.1 First Purchase

Game revenue comes from different sources depending on the business model. In the modern game industry, two main types of monetization are typically used: i.e., fixed pricing (including subscription for online games) and 'freemium' pricing strategies (Marchand and Hennig-Thurau, 2013). In the fixed-pricing case, except for its paid basic content, modern games usually provide extra content as an in-game purchase, such as downloadable content (DLC). These purchases can be considered in-game purchases. On the other hand, the main content of freemium games is often free of charge. However, in-game items, such as special skins and powerful items are the sources of revenue. This type of strategy can usually be found in mobile games and web games, in which the in-game purchases act as the main sources of revenue. As can be seen, no matter which business model a game runs, in-game purchasing behaviours are important–especially for the 'freemium' games. Because it is the most revenue-related topic, purchasing-behaviour prediction is important to any company, because once a predictive model is built, developers are able to determine the important potential purchases in their games so that special care can be taken of these players for achieving better revenue.

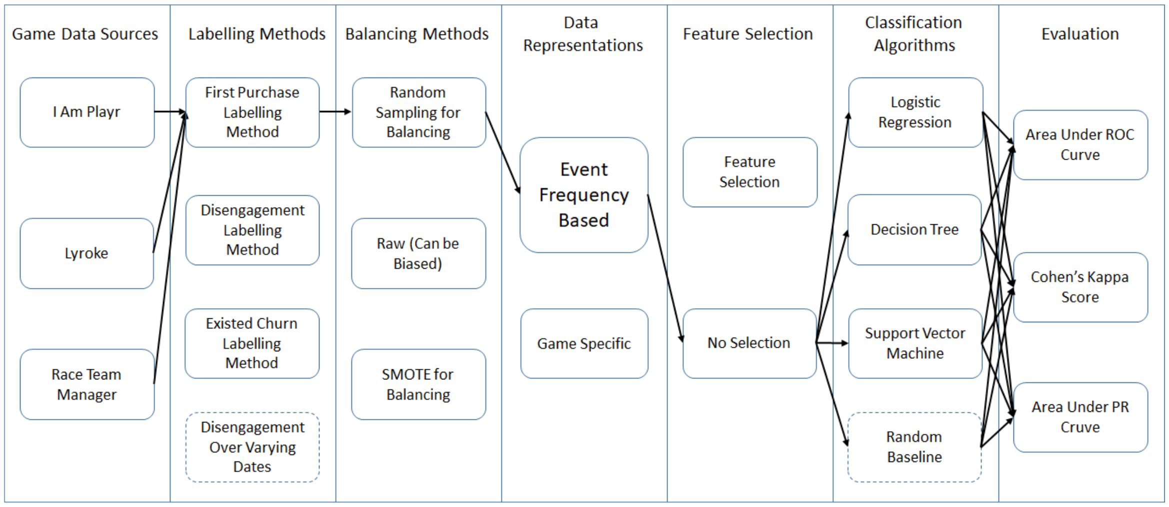

Figure 5.1: Experiment of First Purchase Prediction

As described in Section 3.6, many efforts have been made to predict purchase behaviours. However, except for a recent study by Sifa et al. (2015), most studies in this area are not focused on players' first-purchase decisions. As discussed in Section 4.5.2, we attempted to conduct an experiment that would allow us to compare event-frequency-based data representation (introduced in Section 4.5.1) with the features used by Sifa et al. (2015). However, because most of the features they chose were not available for testing in our game datasets, the comparison was not successfully made. Details of the availability issue are further explained in Section 5.2. First purchase is a special and important behaviour among all purchasing behaviours. This is because the first purchase is the point at which a non-paying player becomes a paying one. According to Kim (2012), once a player has made his/her first purchase, it is very often the case that he or she will start paying for more items. To investigate whether the first purchase can be successfully foreseen, event-frequency-based data representation is utilised.

5.2 First-purchase prediction

In this section, the experiment of predicting players' first purchase with event frequency-based data representation is going to be shown and discussed. This experiment follows the procedure shown in Figure 5.1.

5.2.1 Experiment Information

Game Data Sources

To investigate the generality of event-frequency-based data representation, it was applied to the three games in different genres introduced in Section 4.2: *I Am Playr* , *Lyroke* and *Race Team Manager* .

Labelling method

In this work, the first-purchase prediction is considered a binary classification problem: That is, players are simply labelled as either paying or non-paying users in accord with whether they have purchased any item in the game.

Table 5.1: The availability of the game-specific features used in the study by Sifa et al. (2015) in the three games of this research

	I Am Playr	***Lyroke***	***Race Team Manager***
Country	NA	NA	Yes
Device	NA	NA	Yes
Move Count	ND	ND	ND
Active Opponents	ND	ND	Yes
Logins & Game rounds	Yes	Yes	Yes
Skill-1,2,3	ND	ND	ND
Reached Goals	Yes	Yes	ND
World Number	ND	ND	ND
Number of Interactions	ND	ND	ND
Number of Purchases	Yes	Yes	Yes
Amount Spent	Yes	Yes	Yes
Playtime	NA	NA	NA
Last Inter-session Time	NA	NA	NA
Last Inter-login Time	NA	NA	NA
Inter-login time distribution	NA	NA	NA
Inter-session time distribution	NA	NA	NA
Correlation on time	NA	NA	NA
Mean and Deviation on Time	NA	NA	NA
Country Segments	NA	NA	NA

Balancing Method

As mentioned in Section 4.4, it is often in first-purchase prediction that the distribution of resultant classes after labelling is biased to one side. To solve this problem, we attempted to apply two approaches for balancing the class distribution. This chapter considers the first method, random undersampling, which was applied for balancing the class distribution. Random undersampling is a commonly used approach that has been used in multiple areas (López et al., 2013). It randomly removes data samples from the major class until the size of both classes becomes the same (Chawla, 2005). The other balancing method we tried is called SMOTE (synthetic minority over-sampling technique); however, due to the computational complexity problems of SMOTE, the experiment with SMOTE was not successfully finished when this thesis was written. Details concerning this attempt are introduced in Chapter 7.

Data representation

As it is the main contribution of this research, event-frequency-based data representation was used as the main method for representing the dataset. As was mentioned in Section 4.5.2, this data-representation method is compared with the game-specific features used in another existed study by Sifa et al. (2015). However, as shown in Table 5.1, the availability of the features used in their work is quite limited for testing in our game datasets. In this table, *ND* stands for ‘not defined in the current game context, *NA* for ‘not available’ and *Yes* for ‘available’. The event-frequency-based data representation was compared to a random classifier to see if this data-representation method is able to provide reasonable performance.

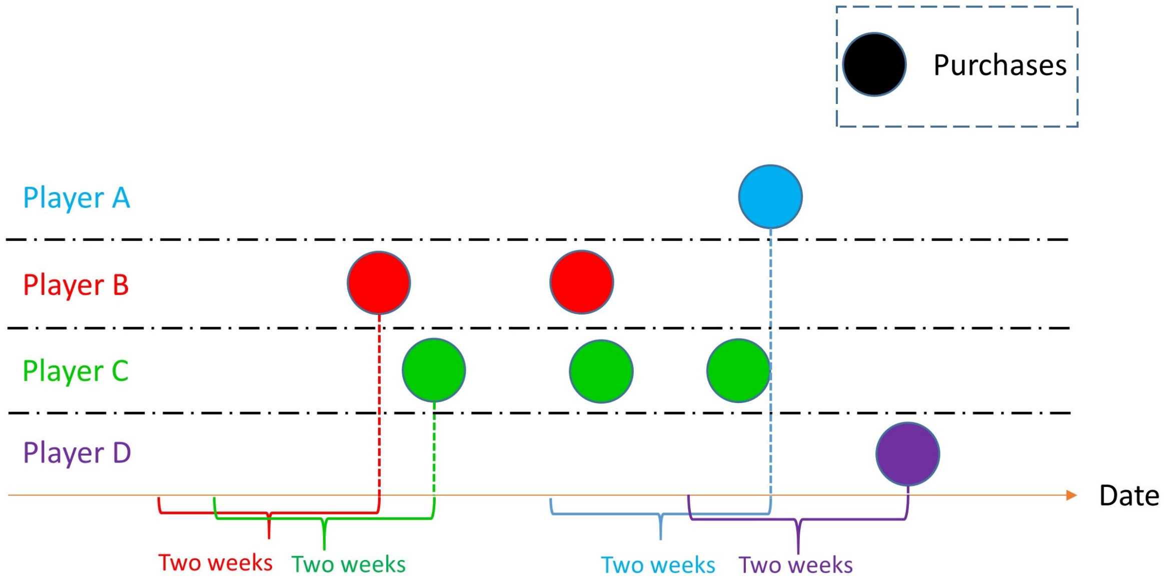

Figure 5.2: Examples of Players in Making First Purchase Class. The circles stand for purchases that players made. For example, Player C made three purchases. Thus, the training data to be considered is selected from two weeks before the date of first purchase.

As for the dataset used in this work, a time period of two weeks before players' first-purchase behaviour was used as the observation window. For players who made purchases, features captured by event-frequency-based data representation are events that happened during the two weeks before their first purchases (shown in Figure 5.2). But for players who have not made any purchase before, training features are events which happened during two weeks before a randomly selected date (shown in Figure 5.3).

One additional thing to note is that, in this work, players were assumed to make their first purchase on a specific date if no other purchases happened before this date recorded in the dataset. This assumption has to be made because the whole history of players is not available in the dataset, which makes it impossible to determine when the 'real' first payment was made after a player had signed up for the game or whether it ever took place.

Classification Algorithms

As discussed in Section 4.7, we applied decision tree, logistic regression and SVM in this experiment. The definitions of these methods were offered in Section 2.2.1. Hyper-parameters used in these algorithms were decided by a random-search method (introduced in 4.7) from the candidates included in Table 4.6.

Evaluation

In this experiment, the evaluation methods utilised are the areas under PR and ROC curves and Cohen's kappa. As discussed in Section 4.8, the area under the PR curve reflects the performance of classifiers for predicting positive examples, whereas the Cohen's kappa score and the area under the ROC curve help to show the comprehensive performance of the classifiers generated.

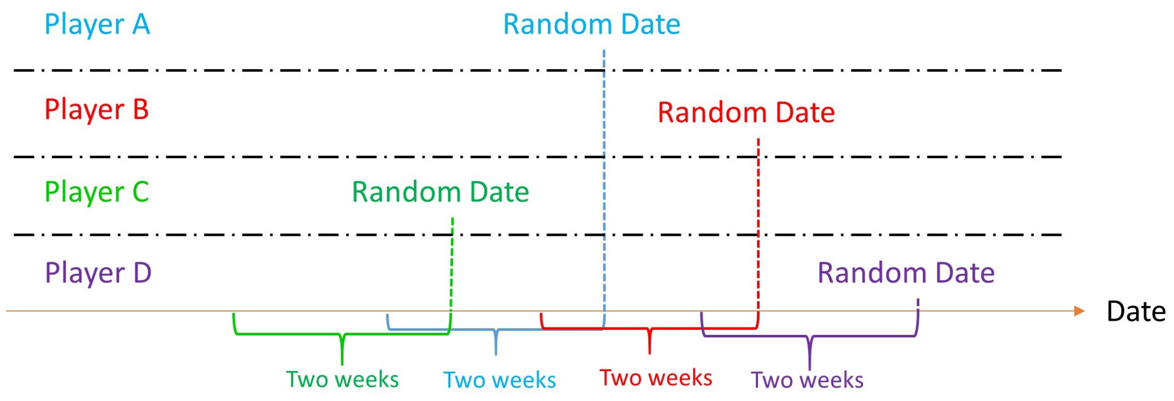

Figure 5.3: Examples of Players in the Non-first-purchase Class. Because no one in this class has made any purchase, a random date is chosen, and two weeks before that date is selected as training data.

5.2.2 Experiment Details and Results

This section shows the results of predicting first purchase with event frequency-based data representation. This experiment aims to determine whether event-frequency-based data representation can be applied across different games and can provide a good predictive performance.

I Am Playr

In the experiment conducted on *I Am Playr*, there are 489 players who at least purchased one item and 88,568 non-paying ones. As mentioned before, it is natural to observe a highly biased situation when predicting first purchases, and a biased situation may bring negative effects to the classifiers being trained. Therefore, to get rid of its effect on training the classifier, the random-undersampling method is included as a sub-process of the whole training process in this section. The clear structure of this training process was introduced in Algorithm 3. This balancing process did improve the performance of the classifiers, and its details can be found in Chapter 7. In this game, while event-frequency-based data representation acts as the data-representation approach, there are 2,460 features (game events) that have been experienced by these selected players. Some features may not have been experienced by a subset of players in some 10-fold split pieces, but to keep the features being used consistent across all cases, events were summarised before the 10-fold cross-validation.

To show if event-frequency-based data representation can offer reasonable performance, a random classifier was added as a baseline to show the difference in performance. As has been discussed in 4.8, the area under both PR curves can be approximated with the ratio of positive examples in the class, whilst the area under the ROC curve and the Cohen's kappa score will always be 0.50 and 0.0, respectively, for a random classifier under any class distributions.

The result for predicting first purchase with event-frequency-based data representation in *I Am Playr* is shown in Table 5.2. In the table, as discussed in Section 4.8.2, the evaluation measurements applied include the *area under the PR curve (with both classes act as positive examples)*, the *area under the ROC curve* and the *Cohen's kappa* score. All measurements (including the area under the PR curves of random

Table 5.2: Performance for predicting first-purchase behaviours with event-frequency-based data representation of the dataset of *I Am Playr* (balanced by random undersampling)

		Event Feature Based Data Representation	Random Classifier	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (first purchase)	LogisticRegression	**0.38631±0.01606**	0.00549	2.2496e+01	3.2106e-09	10.0606
	DecisionTree	**0.49320±0.00746**	0.00549	6.1987e+01	3.7325e-13	27.7215
	SVM	**0.07478±0.00680**	0.00549	9.6723e+00	4.7187e-06	4.3256
AUPRC (non first purchase)	LogisticRegression	**0.99954±0.00018**	0.99451	2.6209e+01	8.2724e-10	11.7212
	DecisionTree	**0.99983±0.00003**	0.99451	1.8867e+02	1.6792e-17	84.3754
	SVM	**0.99976±0.00001**	0.99451	4.0886e+02	1.5943e-20	182.8487
AUROC	LogisticRegression	**0.97515±0.00491**	0.50000	9.1848e+01	1.0898e-14	41.0756
	DecisionTree	**0.97141±0.00430**	0.50000	1.0407e+02	3.5446e-15	46.5403
	SVM	**0.95860±0.00207**	0.50000	2.0982e+02	6.4548e-18	93.8344
KAPPA	LogisticRegression	**0.15555±0.00372**	0.00000	3.9631e+01	2.0627e-11	17.7234
	DecisionTree	**0.11900±0.00506**	0.00000	2.2330e+01	3.4289e-09	9.9861
	SVM	**0.07448±0.00241**	0.00000	2.9346e+01	3.0235e-10	13.1238

classifiers) shown in the table are rounded up for easier display. As for the algorithms, classifiers such as logistic regression, decision tree and support vector machine (SVM) were applied. As for the columns of the table, the *event-frequency* and the *random classifier* columns show the prediction accuracy (measured by three different criteria) brought by the event-frequency-based data representation and a random classifier. To the right of them, the 't-value' and 'p-value' columns show the results of two-tailed t-tests conducted between the results of event-frequency-based data representation and a random classifier. Along with it, the effect sizes (represented by Cohen's D) were also computed and provided in the table to show the size of the differences between the event-frequency-based data representation and the random classifier. The numbers shown after the ± sign are SEMs (standard error of the mean) calculated from 10-fold cross-validation.

Because of the bias that has been discovered, the performance measured by AUPRC (with non-first-purchasers as positive examples) was expected to be high and the AUPRC (with first-purchasers as positive examples) was expected to be low. This bias situation can be firstly reflected by the performance of random classifier respectively in Table 5.2. While being compared with the event frequency based data representation, significant differences can be found across both cases of AUPRC. By combining it with t-values and effect-size, it is indicated that the classifier trained with the event-frequency-based data representation achieved significantly better performance than the random classifier for predicting both first-purchasers and non-first-purchasers. This is further verified while AUROC and KAPPA are used as the measurements. As can be seen, in the six cases of both experiments, the event-frequency-based data representation also brought significantly better performance than the random classifier. While being measured by AUPRC (first-purchasers as positive examples), SVM classifiers seemed behaving not as good as the other two algorithms. This is possibly related to its parameter selections. This found will be further observed in the following experiments. Another observation to notice is that even though the performance of event-frequency-based data representation measured Cohen's Kappa is significantly better than the random classifiers, the score of all three classifiers are still not high. As has been introduced in Section 4.8.2, even though the Cohen's Kappa stays at 0.0 for a random classifier, different from AUROC, its score is affected by bias in both sides (more positive or negative examples).

Table 5.3: Performance for predicting first purchase behaviours with event frequency based data representation on the dataset of *Lyroke* (Balanced by the random undersampling method)

		Event Feature Balanced by Undersampling	Random Classifier	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (first purchase)	LogisticRegression	**0.20950±0.01448**	0.00182	1.3607e+01	2.6213e-07	6.0854
	DecisionTree	**0.40875±0.00929**	0.00182	4.1537e+01	1.3541e-11	18.5761
	SVM	**0.04072±0.00245**	0.00182	1.5046e+01	1.0987e-07	6.7286
AUPRC (non first purchase)	LogisticRegression	**0.99976±0.00009**	0.99818	1.6735e+01	4.3453e-08	7.4842
	DecisionTree	**0.99995±0.00000**	0.99818	6.9915e+02	1.2756e-22	312.6676
	SVM	**0.99995±0.00000**	0.99818	1.2734e+03	5.7814e-25	569.5033
AUROC	LogisticRegression	**0.97369±0.00479**	0.50000	9.3905e+01	8.9300e-15	41.9957
	DecisionTree	**0.97466±0.00128**	0.50000	3.5226e+02	6.0956e-20	157.5337
	SVM	**0.97468±0.00073**	0.50000	6.1856e+02	3.8403e-22	276.6303
KAPPA	LogisticRegression	**0.06182±0.00130**	0.00000	4.5221e+01	6.3239e-12	20.2236
	DecisionTree	**0.04642±0.00358**	0.00000	1.2309e+01	6.1982e-07	5.5048
	SVM	**0.02815±0.00127**	0.00000	2.1088e+01	5.6873e-09	9.4309

Lyroke

Similar experiments were also conducted on the game *Lyroke* . In this game, the degree of imbalance is as serious as it is in the dataset of *I Am Playr* . After the players have been labelled, there are 509 players who have paid for in-game items and 279,829 players who have not made any purchases. Because of this, the random-undersampling method was also used to train the classifiers. The event-frequency-based method was used as the data-representation method, and 7,823 events form the feature space. To determine whether event-frequency-based data representation is able to provide reasonable prediction for this game, a random classifier was added as the baseline, just as it was for the experiment on *I Am Playr* . The performance of the random classifiers are the same as before. The area under the PR curve is associated with the ratio of positive examples in the dataset, while the area under the ROC curve is always 0.5, and the score of Cohen's kappa is always 0. The results of the experiment are shown in Table 5.3. The notations are the same as the result table from *I Am Playr*.

Since similar imbalance was also observed in this experiment, the bias of performance in both AUPRC measurements are expected to be similar to the situations in I Am Playr. The degree of the bias can be seen from the performance of random classifiers in both AUPRC metrics. As for the performance brought by the event-frequency-based data representation, by comparing it with the random classifier with both AUPRC measures, observations of the t-values and p-values could show that its performance is significantly better than random classifiers for predicting both classes. This conclusion can be also verified by the measurements of AUROC and Cohen's Kappa. While being measured by Cohen's Kappa, the performance achieved by three classifiers are not high. As has been explained in the experiment of I Am Playr, this metric is affected by the bias towards both sides. Similar to what was found in the experiment of I Am Playr, it is notable that while being measured by AUPRC (first-purchasers as positive examples), the SVM classifiers also seem to have not achieved as good performance as the other classifiers. This will be validated in the experiment in *Race Team Manager*.

Race Team Manager

Produced by a different game company, the racing game, *Race Team Manager* , was also used to test the performance of event-frequency-based data representation on first-purchase prediction. In this game, 511 players had purchased at least one item whereas

Table 5.4: Performance for predicting first purchase behaviours with event frequency based data representation on the dataset of *Race Team Manager* (Balanced by the random undersampling method)

		Event Feature Balanced by Undersampling	Random Classifier	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (first purchase)	LogisticRegression	**0.17558±0.01293**	0.00299	1.2659e+01	4.8761e-07	5.6614
	DecisionTree	**0.32289±0.01197**	0.00299	2.5344e+01	1.1150e-09	11.3341
	SVM	**0.08678±0.00896**	0.00299	8.8710e+00	9.6073e-06	3.9672
AUPRC (non first purchase)	LogisticRegression	0.99791±0.00043	0.99701	1.9971e+00	7.6916e-02	0.8931
	DecisionTree	**0.99984±0.00001**	0.99701	2.7513e+02	5.6339e-19	123.0422
	SVM	**0.99979±0.00008**	0.99701	3.4298e+01	7.5141e-11	15.3387
AUROC	LogisticRegression	**0.91618±0.00905**	0.50000	4.3614e+01	8.7465e-12	19.5049
	DecisionTree	**0.95073±0.00289**	0.50000	1.4807e+02	1.4855e-16	66.2200
	SVM	**0.95884±0.00333**	0.50000	1.3075e+02	4.5501e-16	58.4723
KAPPA	LogisticRegression	**0.07357±0.00095**	0.00000	7.3098e+01	8.4863e-14	32.6905
	DecisionTree	**0.05119±0.00142**	0.00000	3.4105e+01	7.9034e-11	15.2522
	SVM	**0.06010±0.00082**	0.00000	6.9227e+01	1.3837e-13	30.9595

170,333 players had not made any purchases.

As in previous experiments, the bias in this game is also serious. Therefore, the random-undersampling method was included during the training process. After applying event-frequency-based data representation, there are 1,531 features (game events) that have been experienced by these selected players. The performance of the random classifiers is decided in the same way as for other games. Table 5.4 displays the performance of first-purchase prediction done with event-frequency-based data representation. The notations are the same as in the previous tables.

In Race Team Manager, while being labelled by the first-purchasing categorical definition, the bias is similar to what was observed in the previous two experiments. Therefore, the performance measured by both AUPRC measurements are expected to be biased as well. Same as before, this can be found from the biased performance of random classifiers. In terms of the performance achieved by the event-frequency-based data representation, as can be seen, excepting only one case (algorithm: logistic regression, measurement: area under the PR curve with non-first-purchase as positive), the performance for predicting first purchase with this data representation is superior to that obtained using the random classifier. Given that the performance of the random classifier is already around 1.00 due to the bias, the performance of the event-frequency-based data representation in the only exceptional case is also high. While considering the AUROC measurements, all algorithms display stable performances for predicting first purchases. Even though the performance of the event-frequency-based data representation measured by Cohen's Kappa is also significantly better than the random classifiers, due to the bias, the means of the score are not high. Similar to the observations of SVMs found in the two previous experiments, while being measured by AUPRC (first-purchaser as positive examples), the performance of SVM is not as good as other classifiers.

5.2.3 Discussion

In all three experiments, the performance brought by event-frequency-based data representation is significantly better than the random classifier (except for the only case in which the random guess has already achieved almost 1.0). A summary of the experiments con-

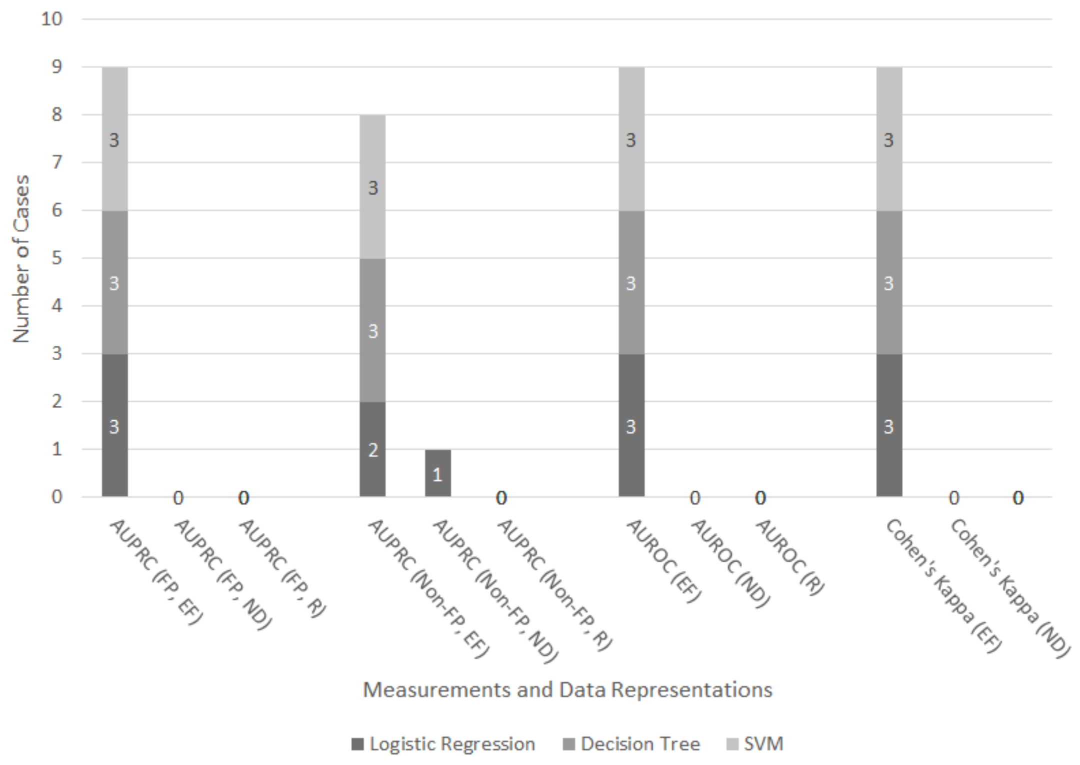

Figure 5.4: The number of cases where methods achieve significantly better performance and the number of cases where there is no difference found for predicting first purchasing behaviours

ducted across three games can be found in Figure 5.4. In this figure, for each measurement, three stacked bars are used to show the number of cases where the event-frequency-based data representation (labelled as EF) achieved significantly better performance, the number of cases where there are no difference found between both methods (labelled as ND), and the number of cases where the random classifier (labelled as R) achieved better performance. The evidences indicate that the event-frequency-based data representation is able to provide a good generality, as it can be easily applied to three different types of games without special pre-processing and achieves significantly better predictive performance than random classifiers in almost all cases across the three games.

However, as has been observed across the experiments in the three games, SVM classifiers did not work as good as the other algorithms for predicting the first purchasers while being measured by the AUPRC. In order to investigate if the performance of SVMs are affected by their selections of parameter in these cases, three extra experiments were conducted to investigate if the parameter selection is substantially affecting the SVM algorithm. Table 5.5 to 5.7 display the performance of SVM classifiers that were trained to predict the first purchasers with a range of parameter selections. Performance in these table are measured by the averaged AUPRC of 10-fold cross validation with first purchasing players as the positive examples, and the error bars show standard error of the mean.

As can be seen, in the experiments conducted in all three game, especially in I Am Playr and Lyroke, the parameter choices seem to have a substantial impact on the classifier's performance. First, in all three games, when the parameter $gamma = 100$, the mean of the

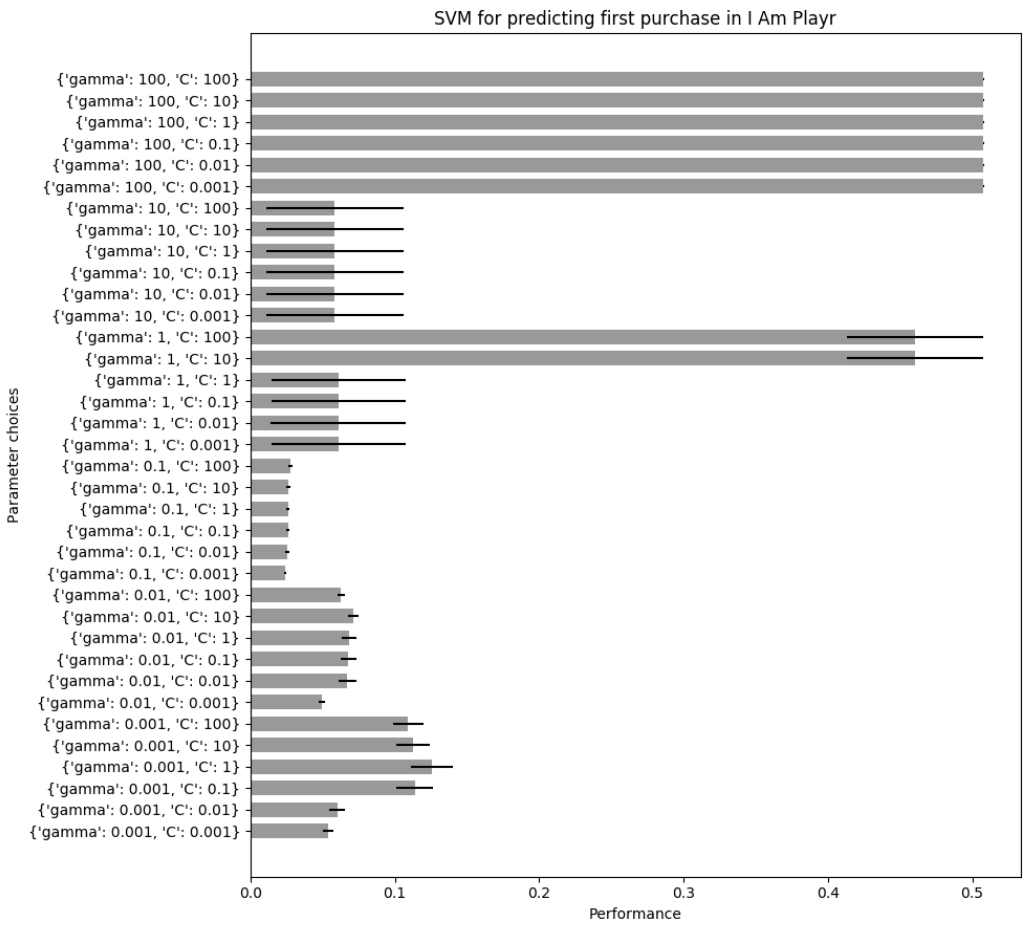

Figure 5.5: Investigation of SVM in I Am Playr for Predicting First Purchase

performance of classifier are higher than other cases. Because that the *gamma* parameter in SVM stands for the inverse influence radius of each data sample in the support vector, when larger *gamma* are more likely to achieve better performance, the model is suggested to reduce the influence of each data sample in the support vector. This may reflects that the data samples are noisy in these games, models are suggested not to trust in specific examples to make decisions. The difference among the three games is that, in I Am Playr, there are two other cases ($gamma = 1$, $C = 100$ or 10) which also can provide higher performance than the rest cases; in Lyroke, the parameter *gamma* also worked well with 10; and in Race Team Manager, the performance across different parameter selections worked similar than the other two games, but cases with $gamma = 100$ are still outstanding. that in this experiment, the mean of the performance of classifiers are also high when the $gamma = 10$. There results indicate that the random searching applied in this research did not find the optimal parameter sets of the SVM classifiers. Therefore, through a better fine-tune process, the performance brought by SVM classifier can be further improved in the future research.

As was mentioned in Section 4.6, a possible limitation of event frequency-based data representation might be that the models built by it are often highly dimensional. An overly complex model can contain redundant structures and can easily lead to overfitting when the quantity of the training data is insufficient. To reduce its dimension, a feature-selection

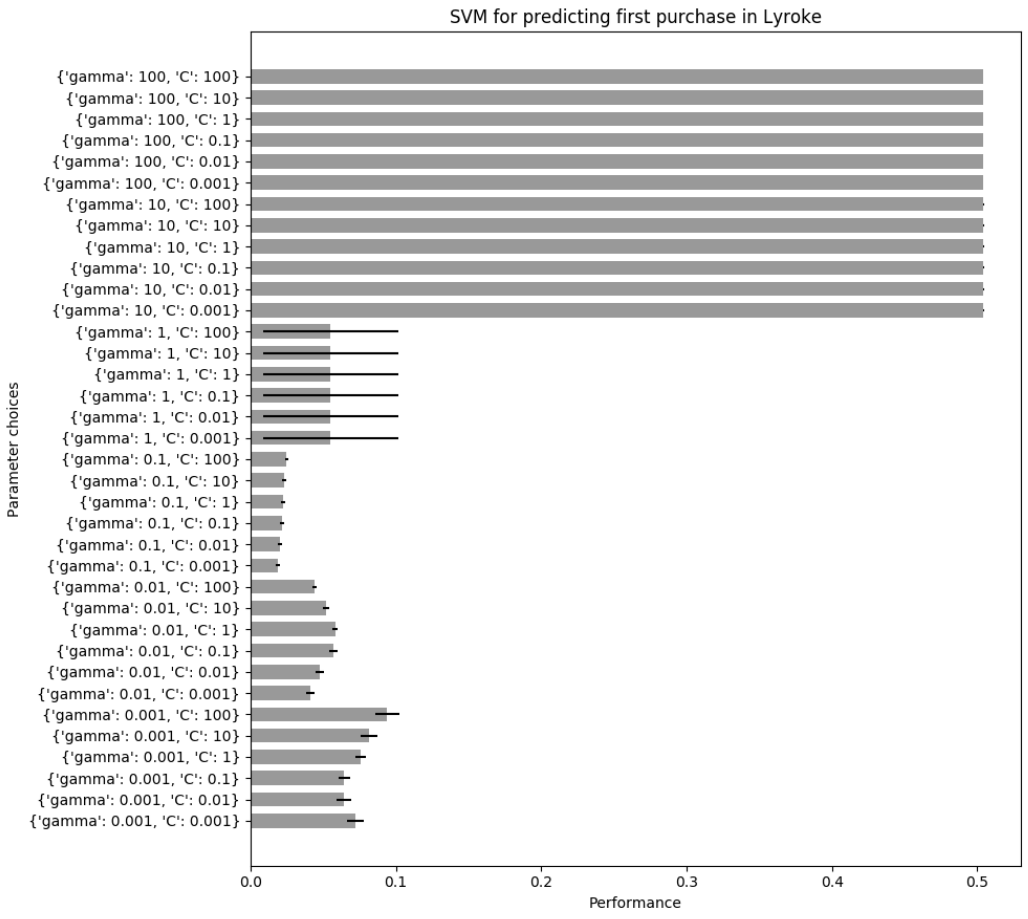

Figure 5.6: Investigation of SVM in Lyroke for Predicting First Purchase

process was often included for reducing the number of features to be used. To determine if a feature-selection process may have a negative impact on the performance of classifiers, the experiment for predicting first purchases by event-frequency-based data representation with feature selection enabled is shown in the next section.

5.3 First-purchase Prediction with Feature Selection

The previous section shows that event-frequency-based data representation is able to provide better performance than random guessing across different games; however, when event-frequency-based data representation was applied, depending on the complexity of the games, the model built might be highly dimensional. For example, in *I Am Playr* , 2,460 events were used as features. As mentioned above, a redundant model may bring problems such as over-fitting or may be hard to interpret. Even if a decision tree (the most interpretable classifier) was applied, the rules generated from a deep tree could hardly be understood. In this section, a feature-selection method called *random forest* is used to reduce the model's complexity. An experiment was conducted to investigate whether a feature-selection algorithm has any negative impact on performance when event-frequency-based data representation is used. This experiment follows the routes shown in Figure 5.8.

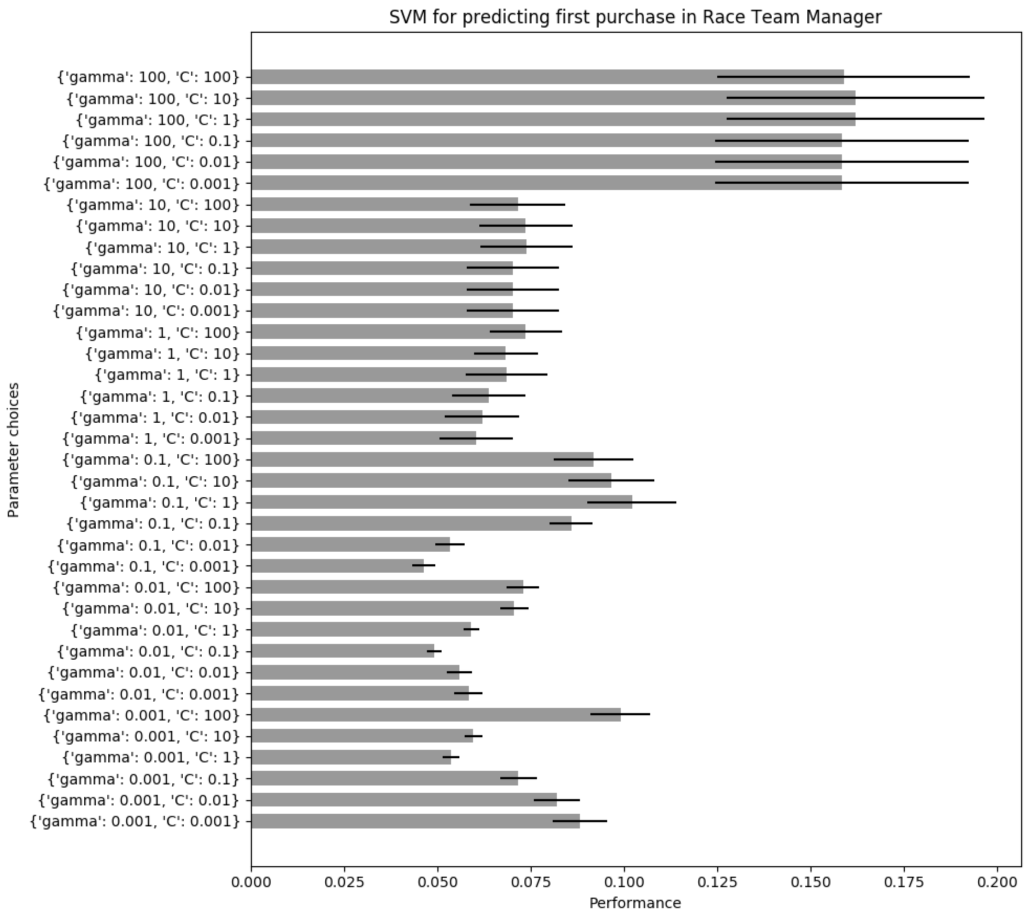

Figure 5.7: Investigation of SVM in Race Team Manager for Predicting First Purchase

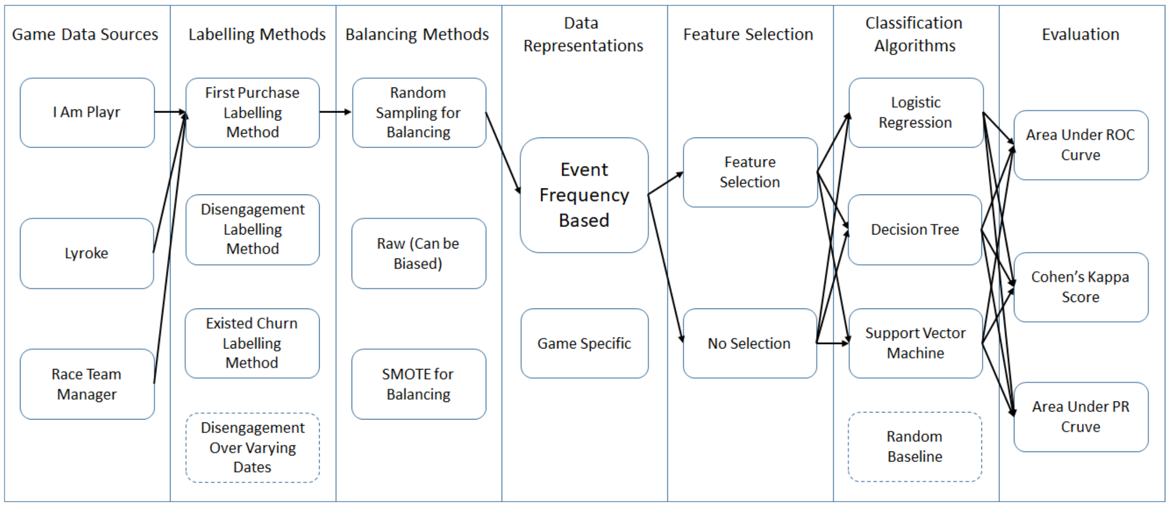

Figure 5.8: Experiment of First-purchase prediction

Table 5.5: Performance for predicting first-purchase behaviours with event-frequency-based data representation (with and without feature selection) on the dataset of *I Am Playr* (balanced by the random-undersampling method)

		Event Feature Without Feature Selection	Event Feature with Feature Selection	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (first purchase)	LogisticRegression	0.38631±0.01606	0.39145±0.01583	-2.1643e-01	8.3109e-01	-0.0968
	DecisionTree	0.49320±0.00746	0.49694±0.01072	-2.7162e-01	7.8901e-01	-0.1215
	SVM	0.07478±0.00680	0.08266±0.00797	-7.1388e-01	4.8446e-01	-0.3193
AUPRC (non first purchase)	LogisticRegression	0.99954±0.00018	0.99966±0.00011	-5.0751e-01	6.1796e-01	-0.2270
	DecisionTree	0.99983±0.00003	0.99980±0.00004	6.2394e-01	5.4050e-01	0.2790
	SVM	0.99976±0.00001	0.99966±0.00011	9.6848e-01	3.4564e-01	0.4331
AUROC	LogisticRegression	0.97515±0.00491	0.97882±0.00412	-5.4278e-01	5.9394e-01	-0.2427
	DecisionTree	0.97141±0.00430	0.96634±0.00668	6.0579e-01	5.5222e-01	0.2709
	SVM	0.95860±0.00207	0.95822±0.00213	1.2159e-01	9.0457e-01	0.0544
KAPPA	LogisticRegression	0.15555±0.00372	0.16965±0.00384	-2.4995e+00	2.2331e-02	-1.1178
	DecisionTree	0.11900±0.00506	0.12010±0.00546	-1.4040e-01	8.8990e-01	-0.0628
	SVM	0.07448±0.00241	0.08144±0.00290	-1.7507e+00	9.7021e-02	-0.7829

5.3.1 Experiment Information

As this experiment shares the same predictive purpose as that introduced in Section 5.2, the game-data sources, labelling methods, balancing methods, classification algorithms and evaluation methods are identical. The difference is that, in this experiment, rather than comparing to a random classifier, event-frequency-based data representation is compared to a feature-selected version of itself to show the performance changes.

5.3.2 Experiment Details and Results

I Am Playr

The experiment was first applied in *I Am Playr* to investigate if a feature-selection process can have any impact on the performance of event-frequency-based data representation. As mentioned in discussion of the previous experiment, the class distribution is biased with respect to the non-paying user class, while the first-purchase labelling method was applied to *I Am Playr*. As before, a random-undersampling method was included for training the classifiers.

When event frequency-based data representation was applied, there were 2,460 features that were experienced by these players. In this experiment, while the random-forest algorithm was applied for feature selection, only 512.8 (averaged from 10-fold cross-validation) important features were finally used. According to what was introduced in Section 4.6, these selected features have higher importance scores than the mean of the scores from all features.

The results for predicting first purchases with the two versions of event-frequency-based data representation in *I Am Playr* are shown in Table 5.5. This table uses notations similar to those used for the previous experiment. The difference is that the 't-value' and 'p-value' columns now show the results of t-tests conducted between the event-frequency-based data representation and its alternative version with the feature-selection approach. As before, the effect sizes (represented by Cohen's D) can display the size of the difference between these two data representations. SEM values were also provided after the ± sign of each measurement. According to the p-values shown in all three experiments, no significant difference can be found between the mean of

Table 5.6: Equivalence testing on the performance for predicting first-purchase behaviours with event-frequency-based data representation (with and without feature selection) on the dataset of *I Am Playr* (balanced by the random-undersampling method)

		Event Feature Without Feature Selection	Event Feature with Feature Selection	P-Value
AUPRC (first purchase)	LogisticRegression	0.38631±0.01606	0.39145±0.01583	3.7671e-02
	DecisionTree	0.49320±0.00746	0.49694±0.01072	**1.7459e-03**
	SVM	0.07478±0.00680	0.08266±0.00797	**6.3342e-04**
AUPRC (non first purchase)	LogisticRegression	0.99954±0.00018	0.99966±0.00011	**1.0682e-32**
	DecisionTree	0.99983±0.00003	0.99980±0.00004	**3.6807e-44**
	SVM	0.99976±0.00001	0.99966±0.00011	**4.0461e-38**
AUROC	LogisticRegression	0.97515±0.00491	0.97882±0.00412	**1.0208e-06**
	DecisionTree	0.97141±0.00430	0.96634±0.00668	**2.1127e-05**
	SVM	0.95860±0.00207	0.95822±0.00213	**2.5529e-12**
KAPPA	LogisticRegression	0.15555±0.00372	0.16965±0.00384	**2.6681e-06**
	DecisionTree	0.11900±0.00506	0.12010±0.00546	**3.5171e-06**
	SVM	0.07448±0.00241	0.08144±0.00290	**1.3045e-09**

the performance brought by either version (with or without feature selection) of the event frequency based data representation.

To investigate if both version are similar enough, this work further conducted a separate equivalence testing, namely two one-sided test (TOST), to work out if the impact of the performance brought by the feature selection is within the interval [-0.05, 0.05]. In a TOST, the null hypothesis is that the difference of two groups of samples is outside of a given interval. Therefore, a 'p-value' less than 0.01 indicates that a significant similarity (difference within the given interval) is presented between the two groups Walker and Nowacki (2011). The results of the TOST are shown in Table 5.6. As can be seen, in all other cases except for the first case (Logistic Regression is used and measured by AUPRC), the performance difference of both versions (with and without feature selection) of the event frequency based data representation are significantly within 0.05. Even in the only exceptional case, the *p-value* is small enough to be less than 0.05. In other words, the features selection can be applied to reduce the dimension of the models without significantly affect their performance in this experiment.

Lyroke

In this experiment, to further test the generality of the conclusion we reached from the game *I Am Playr* , similar research experiments are applied to another game: *Lyroke* . As observed for *I Am Playr* , the class distribution shown in *Lyroke* is biased to non-paying users. For this reason, the random-undersampling process was applied prior to the training of the classifiers, as before. The event-frequency-based data representation used 7,823 unique events to represent the player behaviours in the game. After being processed by the feature selection, the number of features involved was dramatically reduced to about 1,481.7 (averaged from 10-fold cross-validation), which is more than six times the original dimension.

The results of the experiment are shown in Table 5.7. The notations inside the table are the same as before. Similar to what was observed in I Am Playr, the p-values and effect sizes in the table indicate, based on any measurement of the three, no

Table 5.7: Performance for predicting first-purchase behaviours with event-frequency-based data representation (with and without feature selection) on the dataset of *Lyroke* (balanced by the random-undersampling method)

		Event Feature Based Data Representation	Event Feature Based Data Representation with Feature Selection	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (first purchase)	LogisticRegression	0.20950±0.01448	0.21968±0.01455	-4.7050e-01	6.4365e-01	-0.2104
	DecisionTree	0.40875±0.00929	0.40528±0.01312	2.0471e-01	8.4010e-01	0.0915
	SVM	0.04072±0.00245	0.04373±0.00366	-6.4878e-01	5.2467e-01	-0.2901
AUPRC (non first purchase)	LogisticRegression	0.99976±0.00009	0.99977±0.00008	-6.1736e-02	9.5145e-01	-0.0276
	DecisionTree	0.99995±0.00000	0.99996±0.00000	-7.2997e-01	4.7480e-01	-0.3265
	SVM	0.99995±0.00000	0.99995±0.00000	-5.2823e-01	6.0379e-01	-0.2362
AUROC	LogisticRegression	0.97369±0.00479	0.97490±0.00443	-1.7733e-01	8.6123e-01	-0.0793
	DecisionTree	0.97466±0.00128	0.97656±0.00171	-8.4798e-01	4.0758e-01	-0.3792
	SVM	0.97468±0.00073	0.97539±0.00110	-5.0683e-01	6.1842e-01	-0.2267
KAPPA	LogisticRegression	0.06182±0.00130	0.06589±0.00112	-2.2554e+00	3.6797e-02	-1.0086
	DecisionTree	0.04642±0.00358	0.04697±0.00454	-9.1079e-02	9.2844e-01	-0.0407
	SVM	0.02815±0.00127	0.02972±0.00115	-8.7188e-01	3.9476e-01	-0.3899

Table 5.8: Equivalence testing on the performance for predicting first-purchase behaviours with event-frequency-based data representation (with and without feature selection) on the dataset of *Lyroke* (balanced by the random-undersampling method)

		Event Frequency Based Data Representation	Event Frequency Based Data Representation with Feature Selection	P-Value
AUPRC (first purchase)	LogisticRegression	0.20950±0.01448	0.21968±0.01455	4.1136e-02
	DecisionTree	0.40875±0.00929	0.40528±0.01312	**6.6401e-03**
	SVM	0.04072±0.00245	0.04373±0.00366	**3.7328e-09**
AUPRC (non first purchase)	LogisticRegression	0.99976±0.00009	0.99977±0.00008	**2.6450e-37**
	DecisionTree	0.99995±0.00000	0.99996±0.00000	**1.4188e-63**
	SVM	0.99995±0.00000	0.99995±0.00000	**1.0842e-67**
AUROC	LogisticRegression	0.97369±0.00479	0.97490±0.00443	**6.5054e-07**
	DecisionTree	0.97466±0.00128	0.97656±0.00171	**1.5419e-14**
	SVM	0.97468±0.00073	0.97539±0.00110	**2.1215e-18**
KAPPA	LogisticRegression	0.06182±0.00130	0.06589±0.00112	**7.4181e-16**
	DecisionTree	0.04642±0.00358	0.04697±0.00454	**1.0035e-07**
	SVM	0.02815±0.00127	0.02972±0.00115	**2.8292e-16**

significant difference is found between the two versions of event-frequency-based data representation.

Therefore, same as before, a TOST (with the interval [-0.05, 0.05]) was also conducted in the dataset of Lyroke to work out the similarity between the two versions of the event frequency based data representations. As can be seen from Table 5.8, similar to what was observed in I Am Playr, except for the first case (logistic regression was applied and measured by AUPRC), all other cases indicate that the performance difference between both versions (with and without feature selection) of the data representation is significantly within the interval of [-0.05, 0.05]. Likewise, the *p-value* in the only exceptional case is also smaller than 0.05. The results match the findings in I Am Playr that the feature selection can be applied to reduce the dimension of the predictive models without affecting their performance significantly.

Race Team Manager

Table 5.9: Performance for predicting first-purchase behaviours with event-frequency-based data representation (with and without feature selection) on the dataset of *Race Team Manager* (balanced by the random-undersampling method)

		Event Feature Based Data Representation	Event Feature Based Data Representation with Feature Selection	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (first purchase)	LogisticRegression	0.17558±0.01293	0.17640±0.01672	-3.6651e-02	9.7117e-01	-0.0164
	DecisionTree	0.32289±0.01197	0.32245±0.01482	2.1683e-02	9.8294e-01	0.0097
	SVM	0.08678±0.00896	0.09216±0.00918	-3.9786e-01	6.9541e-01	-0.1779
AUPRC (non first purchase)	LogisticRegression	0.99791±0.00043	0.99807±0.00028	-2.8940e-01	7.7558e-01	-0.1294
	DecisionTree	0.99984±0.00001	0.99978±0.00007	9.7174e-01	3.4406e-01	0.4346
	SVM	0.99979±0.00008	0.99979±0.00008	4.9398e-02	9.6115e-01	0.0221
AUROC	LogisticRegression	0.91618±0.00905	0.91605±0.00941	9.3735e-03	9.9262e-01	0.0042
	DecisionTree	0.95073±0.00289	0.94896±0.00347	3.7094e-01	7.1501e-01	0.1659
	SVM	0.95884±0.00333	0.95931±0.00340	-9.3650e-02	9.2642e-01	-0.0419
KAPPA	LogisticRegression	0.07357±0.00095	0.07353±0.00122	2.5135e-02	9.8022e-01	0.0112
	DecisionTree	0.05119±0.00142	0.05118±0.00160	4.7723e-03	9.9624e-01	0.0021
	SVM	0.06010±0.00082	0.05908±0.00093	7.7649e-01	4.4754e-01	0.3473

The two previous experiments show that feature selection can act as an effective tool for reducing the dimensions of the event-frequency-based data representation model without significantly affecting its performance for predicting first-purchases across games. In this experiment, this solution is further tested on another game, *Race Team Manager* , which is produced by a different game studio. As in previous experiments, because the same bias situation in class distribution can be found in *Race Team Manager* , the random-undersampling balancing method is again included in the process of training classifiers. While the event-frequency-based data representation was used to represent player behaviours, there are 1,531 relevant features (game events) in total. As in the previous two experiments, the random-forest algorithm was applied for selecting the important features. It helped to reduce the dimension of the model to 357.4 (averaged from 10-fold cross-validation).

Table 5.9 displays the performance of first-purchase prediction done with event-frequency-based data representation. According to both p-values and effect sizes, similar results can be seen in the table: There is no significant difference between results from both version of event-frequency-based data representation. Identical to the previous experiments, a further TOST was applied to work out the similarity between the results brought by both versions of the data representation. As can be seen in Table 5.10, except for the first two cases (Logistic Regression and Decision Tree was applied, measured by AUPRC), all the rest cases indicate that the performance different between both data representations is significantly within the interval of [-0.05, 0.05]. Similar to the previous founds, the *p-values* of the exceptional two cases are also smaller than 0.05. Therefore, same as before, evidences indicate that the feature selection is beneficial to reduce the dimension of the created models without affecting the performance of them significantly.

5.3.3 Discussion

Across all three experiments, feature selection seems to act robustly for reducing the dimensions of the event-frequency-based data representation model without significantly affecting its performance for predicting first purchase in most cases. This suggests that feature selection is a beneficial method that can work together with event-frequency-based data repre-

Table 5.10: Equivalence testing on the performance for predicting first-purchase behaviours with event-frequency-based data representation (with and without feature selection) on the dataset of *Race Team Manager* (balanced by the random-undersampling method)

		Event Frequency Based Data Representation	Event Frequency Based Data Representation with Feature Selection	P-Value
AUPRC (first purchase)	LogisticRegression	0.17558±0.01293	0.17640±0.01672	2.0273e-02
	DecisionTree	0.32289±0.01197	0.32245±0.01482	1.1929e-02
	SVM	0.08678±0.00896	0.09216±0.00918	**1.9951e-03**
AUPRC (non first purchase)	LogisticRegression	0.99791±0.00043	0.99807±0.00028	**7.5110e-26**
	DecisionTree	0.99984±0.00001	0.99978±0.00007	**8.4885e-42**
	SVM	0.99979±0.00008	0.99979±0.00008	**1.2823e-37**
AUROC	LogisticRegression	0.91618±0.00905	0.91605±0.00941	**9.7030e-04**
	DecisionTree	0.95073±0.00289	0.94896±0.00347	**3.5890e-09**
	SVM	0.95884±0.00333	0.95931±0.00340	**5.4129e-09**
KAPPA	LogisticRegression	0.07357±0.00095	0.07353±0.00122	**2.7440e-17**
	DecisionTree	0.05119±0.00142	0.05118±0.00160	**8.1514e-15**
	SVM	0.06010±0.00082	0.05908±0.00093	**8.1584e-19**

sentation to generate lower-dimensional models, thereby reducing the chance of overfitting.

5.4 Conclusion

Overall, significantly better performance than random classifiers can be achieved by applying event-frequency-based data representation to the first-purchase prediction problem across all three games. During the experiments conducted in this chapter, event-frequency-based data representation showed its generality, as it shows robust performance across all three different genres of games without any special pre-processing. In addition, since applying feature selection does not significantly affect the performance brought by event-frequency-based data representation, to get a more interpretable and lower-dimensional model, feature selection is recommended for pre-processing of event-frequency-based data representation when predicting first purchase.

As event-frequency-based data representation was smoothly applied to three different genres of games and offered a robust performance for predicting players' first-purchase behaviours, its generality has been investigated. To further study its generality with respect to different predictive purposes, this data-representation method is applied in the next chapter to predict another important player-behaviour trend: player disengagement.

Chapter 6

Predicting Disengagement

The previous chapter verified the performance of the event-frequency-based data representation for predicting first purchases across games. To investigate if the data representation is not only applicable to predicting purchases but also for other player modelling purposes, in this chapter, event frequency-based data representation will be applied for predicting disengagement behaviours. In general, disengagement is a concept which describes players' decreasing activity in a game. This type of prediction is important to a game company, because a game without players is nothing. Once a predictive model of disengagement behaviour is built, companies can use it for forecasting disengaging players and trying to prevent this trend by taking pre-emptive measures.

This chapter investigates the generality and performance of event-frequency-based data representation. Event-frequency-based data representation is first compared with a random classifier to predict disengagement in all three commercial games mentioned in Chapter 4. Next, should the data-representation method achieve a significantly better performance than the random guess, it will be further compared with another state-of-the-art data representation method which relies on specific features selected from games (used in the work by (Runge et al., 2014)). In the two experiments mentioned above, the disengagement-labelling method (introduced in Section 6.1.1) was used as the predictive target. In the last experiment, event-frequency-based data representation is also applied to predict another similar concept which is called 'churn'. This labelling method has been used in several works which focus on predicting the decision of players to quit a game. These two definitions are slightly different in their purposes, but both describe players' disengagement from games. The details of the churn-labelling method are covered in Section 6.4.1.

Main points in this chapter:

- ◆ introduction to the problem of churn and disengagement,
- ◆ introduction of the commonly known churn-labelling method,
- ◆ introduction of the new disengagement-labelling method,
- ◆ description of a game-specific data representation used by Runge et al. (2014),
- ◆ case studies for predicting disengagement to verify data representation performance, and
- ◆ case studies for predicting churn to verify data representation performance.

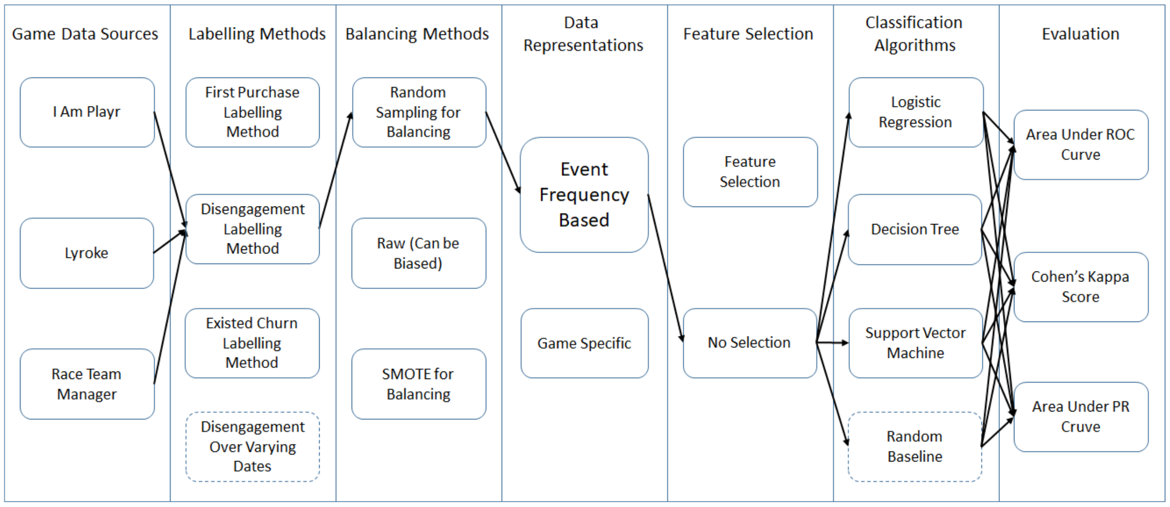

Figure 6.1: Experiment of First-purchase prediction

6.1 Disengagement Prediction with Event-frequency-based Data Representation

To show if event-frequency-based data representation can be used for other predictive purposes, player disengagement is used as a target. Like the experiment regarding first-purchase prediction, a random classifier is added as a baseline to help us identify whether event frequency-based data representation is able to provide better performance. This experiment follows the path shown in Figure 6.1:

6.1.1 Experiment Information

Game Data Sources

Chapter 5 demonstrated that event-frequency-based data representation can be applied to all three games for predicting first purchases and for providing performance that is significantly better than random classifiers. To further verify its generality while predicting disengagement, all three games (i.e., *I Am Playr* , *Lyroke* and *Race Team Manager*) are used again in this experiment.

Labelling Method

The first step of disengagement prediction is to classify players into binary groups (disengaging and non-disengaging) based on their behaviours. Disengagement, sometimes referred to as *churn*, is mentioned in some existing works (Hadiji et al., 2014; Runge et al., 2014). Churn-labelling methods are discussed in detail in their corresponding sections. Since most of these definitions focus on predicting the disengaging action whereby players entirely stop playing the game, they can help the developers identify potentially disengaging players so that pre-emptive measures can be taken to stop disengagement before it occurs. Unfortunately, as discussed by Runge et al. (2014), though players' disengagement can be forecast, it is hard to stop it by giving free currencies. For this reason, instead of investigating players' leaving actions, another labelling method that focuses on players' disengaging trend is introduced in this work. Under this labelling method, if a player's activity trend is predicted to be disengaging, developers have more time to retain the player by giving extra care, because the player has not yet decided to leave the game. This labelling method is called *disengagement* in this work. In the disengagement-labelling method, players are labelled with either category following the procedure below:

1. Divide the dataset into two periods by a middle date.
2. For each player, their total activities (the sum of all event-frequency features) in both Period 1 and Period 2 are calculated separately and sorted.
3. For each period, the sorted list of total activities is divided into four quartiles. Players are then ranked between them according to their overall activity.
4. For each player, if his/her rank in a Period 2 minus his/her rank in Period 2 is greater than two, then he/she would be allocated to the disengagement group. Otherwise, he/she would be allocated to the non-disengagement group.

Balancing Method

After labelling by the disengagement-labelling approach, players are distributed into two groups. Like the first-purchase labelling method, imbalanced classes can often be observed. For example, in *I Am Playr* , 1,341 and 13,396 players were labelled as disengaging and non-disengaging, respectively, after the disengagement-labelling method was applied. As was explained in Section 4.4, imbalanced classes can lead to unreliable models. To deal with this issue, a naive random-undersampling method was used to balance the sizes of the binary class in this experiment, similar to what was done for first-purchase prediction. More approaches have been tried to balance the dataset. Since this is not the focus of this experiment, the details of these methods are explained in Chapter 7.

Data Representation

To further verify the generality and performance of the main contribution of this work, event-frequency-based data representation continues to be applied for predicting disengagement behaviours.

As for the part of the dataset to be used, since the disengagement-labelling method has already split the entire dataset into two by dates and investigates activity changes before and after the middle date, behaviours that players exhibited before the middle date are used for players in either class.

Classification Algorithms

To maintain consistency with the previous experiments, the three algorithms, logistic regression, decision tree and support vector machine, will continue to be used as the main methods for building the classification model. As before, the parameters were optimised with the random search method introduced in Section 4.7.

Evaluation

Because the classification tasks in this experiment have all been balanced by random undersampling, all measurements are applied, including the area under the PR and ROC curves and Cohen's kappa. As discussed in Section 4.8, the area under the PR curve shows the performance of classifiers for predicting positive examples. Instead, the Cohen's kappa score and the area under the ROC curve are good at displaying the comprehensive performance of the classifiers generated for predicting both classes.

Table 6.1: Performance for predicting disengagement behaviours with event-frequency-based data representation on the dataset of *I Am Playr* (balanced by random undersampling)

		Event Frequency Based Data Representation	Random Classifier	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (disengaging)	LogisticRegression	**0.22±0.0082**	0.10	1.3823e+01	2.2896e-07	6.1817
	DecisionTree	**0.35±0.0097**	0.10	2.4022e+01	1.7943e-09	10.7429
	SVM	**0.22±0.0122**	0.10	9.0747e+00	7.9797e-06	4.0583
AUPRC (non disengaging)	LogisticRegression	**0.94±0.0024**	0.90	1.6792e+01	4.2185e-08	7.5095
	DecisionTree	**0.98±0.0010**	0.90	7.2658e+01	8.9593e-14	32.4938
	SVM	**0.97±0.0011**	0.90	6.2581e+01	3.4261e-13	27.9872
AUROC	LogisticRegression	**0.72±0.0084**	0.50	2.4918e+01	1.2961e-09	11.1438
	DecisionTree	**0.81±0.0066**	0.50	4.3719e+01	8.5597e-12	19.5519
	SVM	**0.78±0.0092**	0.50	2.9175e+01	3.1851e-10	13.0473
KAPPA	LogisticRegression	**0.20±0.0081**	0.00	2.3827e+01	1.9289e-09	10.6557
	DecisionTree	**0.23±0.0118**	0.00	1.8069e+01	2.2189e-08	8.0807
	SVM	**0.17±0.0107**	0.00	1.4804e+01	1.2644e-07	6.6207

6.1.2 Experiment Details and Results

I Am Player

At first, the experiment was conducted on the football game *I Am Playr* developed by WeR Interactive. This experiment uses the disengagement-labelling method to distribute players into binary classes. As discussed above, the process requires the dataset to be evenly split into two time periods so that players can be labelled according to their activity differences in between these periods. After labelling, there are 1,354 disengaging and 12,044 engaging players found in the dataset.

Like the situation observed in the first-purchase prediction, this is an imbalanced predictive problem. In case any negative effect is brought in by the bias, a random-undersampling method was used in the process of training to balance the class distribution. While event-frequency-based data representation was applied to represent the dataset, 4,740 events have been experienced by these players.

To determine whether event-frequency-based data representation can provide performance that is better than a random guesser, a random classifier was added in this experiment to test the significance of differences in performance. As introduced in Section 4.8, the area under the PR curve of a random classifier can be approximated by the ratio of positive examples in the dataset whereas the area under the ROC curve and the Cohen's kappa score are always 0.5 and 0.0, respectively.

The result for predicting first purchases in *I Am Playr* with event-frequency-based data representation is shown in Table 6.1. Like the notations used in the previous experiment, the 't-value', 'p-value' and effect-size columns were utilised to show the significance of differences between the performance brought by the event-frequency-based data representation and the random guess. In addition, as in the other experiments, SEMs that were calculated from 10-fold cross-validation can help to reveal any overlaps between performance. Identical to previous experiments, all measurements shown in the table are rounded up for easy display.

According to the table, performance scores across all cases indicate that event-frequency-based data representation is able to provide significantly better performance than a random classifier. In addition, both the p-value ($p < 0.01$) and the effect-size indicate

Table 6.2: Performance for predicting disengagement behaviours with event-frequency-based data representation on the dataset of *Lyroke* (balanced by random undersampling)

		Event Frequency Based Data Representation	Random Classifier	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (disengaging)	LogisticRegression	**0.26±0.0042**	0.12	3.0429e+01	2.1881e-10	13.6084
	DecisionTree	**0.30±0.0091**	0.12	1.8617e+01	1.7066e-08	8.3258
	SVM	**0.25±0.0050**	0.12	2.4542e+01	1.4836e-09	10.9755
AUPRC (non disengaging)	LogisticRegression	**0.95±0.0019**	0.88	3.8628e+01	2.5944e-11	17.2751
	DecisionTree	**0.97±0.0009**	0.88	8.8740e+01	1.4851e-14	39.6859
	SVM	**0.97±0.0006**	0.88	1.4641e+02	1.6441e-16	65.4777
AUROC	LogisticRegression	**0.77±0.0044**	0.50	5.6903e+01	8.0491e-13	25.4476
	DecisionTree	**0.78±0.0054**	0.50	4.9378e+01	2.8742e-12	22.0825
	SVM	**0.78±0.0035**	0.50	7.5626e+01	6.2521e-14	33.8211
KAPPA	LogisticRegression	**0.24±0.0053**	0.00	4.2178e+01	1.1808e-11	18.8624
	DecisionTree	**0.22±0.0077**	0.00	2.7377e+01	5.6122e-10	12.2435
	SVM	**0.21±0.0027**	0.00	7.2414e+01	9.2348e-14	32.3845

that the difference is significant. This further verifies the phenomenon found while predicting first purchases and extends the generality of the data representation. To ensure that this performance is robust with respect to different games, the observation was further verified with respect to other games.

Lyroke

To further investigate the generality of event-frequency-based data representation, the same experiment was also carried out on another game, *Lyroke* , which is a music game also produced by WeR Interactive. After the disengagement-labelling method is applied, 3,494 players are labelled as disengaging and 25,013 players as engaging. As in the previous experiments, a random-undersampling method was included to ensure that the resultant predictive models are not biased. After event-frequency-based data representation was applied, the total number of events that players experiences is 7,395.

The results of the experiment are shown in Table 6.2. The notations applied here are identical to previous ones. The calculations of the measurements of the random classifiers remain the same. As the table indicates, the event-frequency-based data representation is still able to provide significantly better performance than the random guess. The significance of the difference can be further verified by observing the significant test results ($p < 0.01$) and the effect size.

Results in this experiment further display the robust ability of event-frequency-based data representation. To further investigate its generality, the next experiment applies event-frequency-based data representation to another game which is not only in a different genre but is also produced by another company.

Race Team Manager

The third experimental candidate, the racing game, *Race Team Manager* , produced by Bigbit Ltd. was also used for further verification for the same predictive purposes. After the disengagement-labelling method was applied, 3,416 and 22,957 players are labelled as disengaging and engaging, respectively.

Because of this bias, the random-undersampling process was also included for training more balanced classifiers, as was done for the other two games. After the event-

Table 6.3: Performance for predicting disengagement behaviours with event-frequency-based data representation on the dataset of *Race Team Manager* (balanced by random undersampling)

		Event Frequency Based Data Representation	Random Classifier	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (disengaging)	LogisticRegression	**0.27±0.0026**	0.13	4.9651e+01	2.7355e-12	22.2046
	DecisionTree	**0.31±0.0050**	0.13	3.3773e+01	8.6250e-11	15.1038
	SVM	**0.30±0.0031**	0.13	5.0201e+01	2.4783e-12	22.4504
AUPRC (non disengaging)	LogisticRegression	**0.93±0.0018**	0.87	3.2041e+01	1.3805e-10	14.3291
	DecisionTree	**0.97±0.0006**	0.87	1.5558e+02	9.5211e-17	69.5764
	SVM	**0.97±0.0005**	0.87	1.8709e+02	1.8112e-17	83.6691
AUROC	LogisticRegression	**0.75±0.0027**	0.50	8.7060e+01	1.7637e-14	38.9342
	DecisionTree	**0.79±0.0028**	0.50	9.6364e+01	7.0778e-15	43.0955
	SVM	**0.80±0.0026**	0.50	1.1119e+02	1.9539e-15	49.7268
KAPPA	LogisticRegression	**0.25±0.0044**	0.00	5.4943e+01	1.1025e-12	24.5712
	DecisionTree	**0.25±0.0049**	0.00	4.8572e+01	3.3315e-12	21.7221
	SVM	**0.25±0.0038**	0.00	6.1738e+01	3.8702e-13	27.6099

frequency-based method was used as the data-representation method, a total of 464 events were related to these players.

The results of disengagement prediction by event-frequency-based data representation can be found in Table 6.3. Likewise, notations applied in the table are the same as before.

The results indicate that event-frequency-based data representation is able to significantly outperform a random classifier just as the previous experiments on disengagement prediction. P-values and effect sizes shown in the table show the statistical differences.

6.1.3 Summary

According to results from all three experiments conducted on different games, evidence shows that the event-frequency-based data representation is able to behave significantly better than the random guess for predicting disengagement. A summary of all results across the three experiments is shown in Figure 6.2. The notations in this graph are identical to the one that summarised experiment results in first-purchase prediction. According to this figure, the performance that the event-frequency-based data representation achieved is significantly better than random classifiers.

While being combined with the experiments for predicting first-purchase behaviours, as can be seen in 6.3, in almost all cases (except for an only case where no significant difference can be found between both methods), the event-frequency based data representation is able to provide significantly better performance. Thereby, it further indicates that event-frequency-based data representation exhibits good generality not only for different games but also for various predictive purposes.

As mentioned in Section 4, a potential problem event-frequency-based data representation may face is that a complex model can contain redundant features. Therefore, in the cases in which the model needs to be simplified or interpreted, a feature selection algorithm can act as a possible solution, as it did for first-purchase prediction. In the next section, the random forest is added as the feature selector for reducing the dimension of the models, just as it

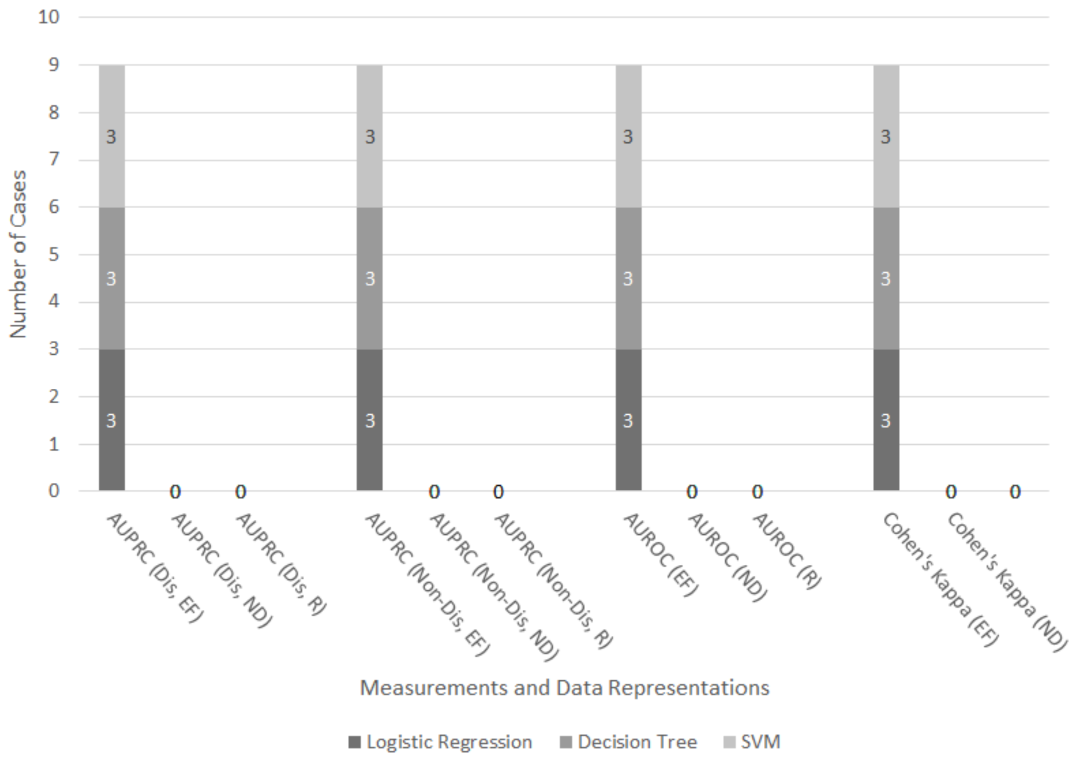

Figure 6.2: The number of cases where methods achieve significantly better performance and the number of cases where there is no difference found for predicting disengagement behaviours

was for first-purchase prediction.

6.2 Disengagement Prediction with Event-frequency-based Data Representation with Feature Selection

As discussed above, complex models are more likely to cause overfitting problems when the quantity of the training data is insufficient. To reduce the dimensionality of the resultant model, applying feature selection as a pre-processor is a common strategy. This section aims determine whether a commonly used feature-selection approach can affect prediction performance. The random-forest feature selector is tested in the disengagement-prediction experiment. The experiments in this section follow the path shown in Figure 6.4:

6.2.1 Experiment Information

Unchanged experiment settings

As in previous experiments, to ensure that the effects caused by the feature selection are not limited to any individual example, experiments were conducted on all three games. These games are *I Am Playr* , *Lyroke* and *Race Team Manager* , all of which are depicted in Figure 6.4. As for the experimental settings, the experiment in this section stays the same as in the previous section except that a feature-selection algorithm is added for comparison.

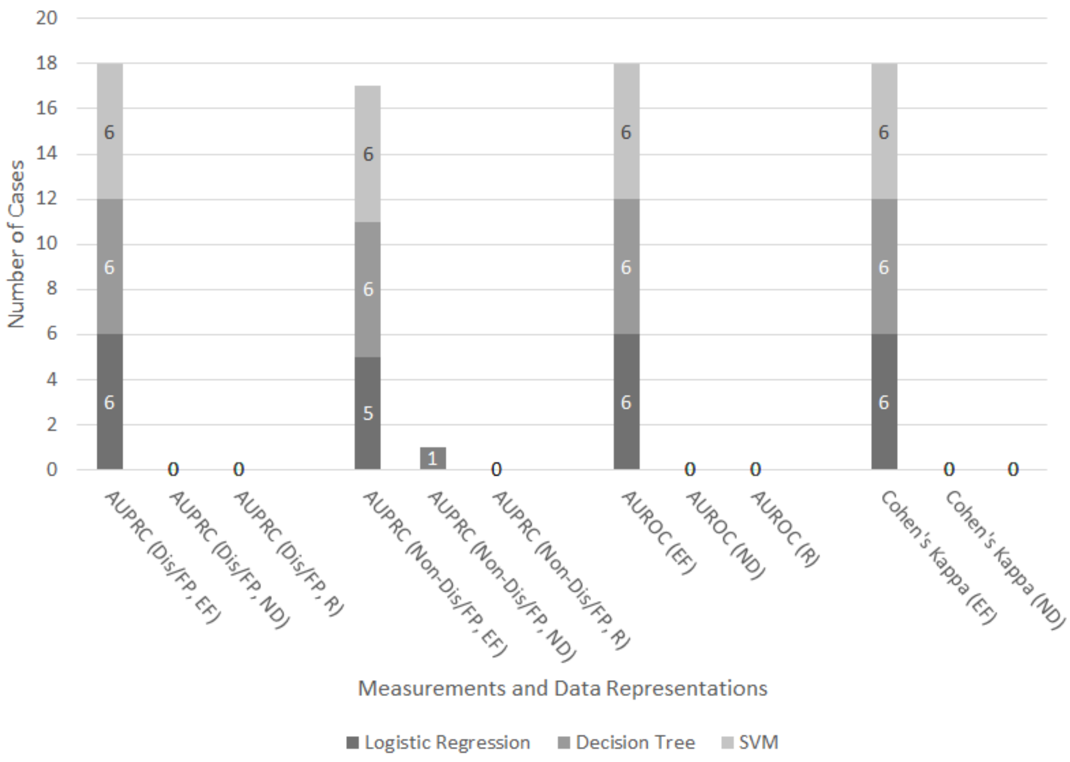

Figure 6.3: The number of cases where methods achieve significantly better performance and the number of cases where there is no difference found for predicting churn behaviours

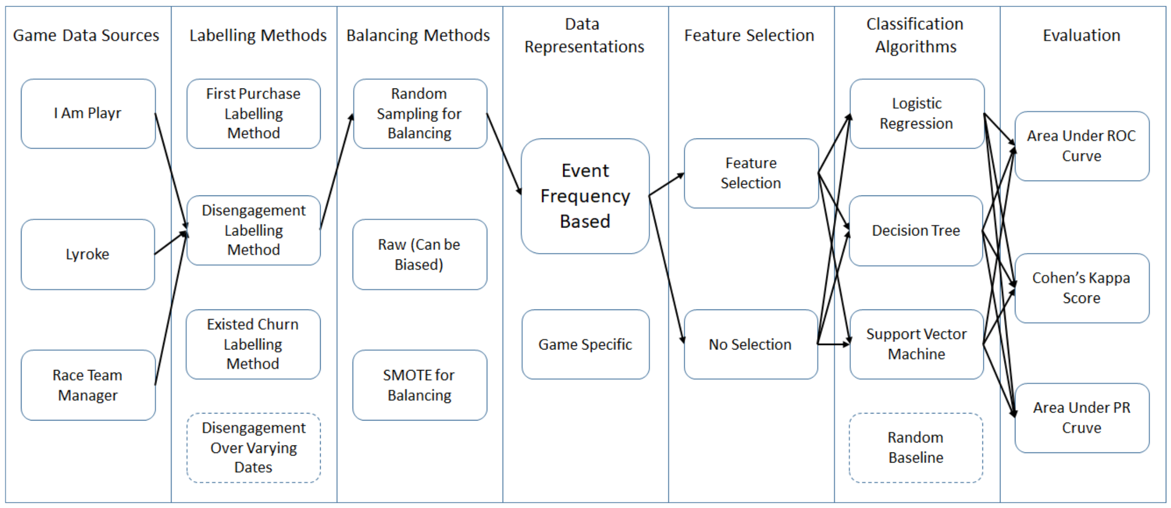

Figure 6.4: Experiment of First Purchase Prediction

Table 6.4: Performance for predicting disengagement behaviours with event-frequency-based data representation (with and without feature selection) on the dataset of *I Am Playr* (balanced by random undersampling)

		Event Frequency Based Data Representation	Event Frequency Based Data Representation with Feature Selection	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (disengaging)	LogisticRegression	0.22±0.0082	0.22±0.0080	-5.6009e-02	9.5595e-01	-0.0250
	DecisionTree	0.35±0.0097	0.35±0.0158	6.6394e-02	9.4780e-01	0.0297
	SVM	0.22±0.0122	0.22±0.0103	5.1473e-02	9.5952e-01	0.0230
AUPRC (non disengaging)	LogisticRegression	0.94±0.0024	0.95±0.0026	-1.2804e+00	2.1666e-01	-0.5726
	DecisionTree	0.98±0.0010	0.97±0.0010	8.9687e-01	3.8163e-01	0.4011
	SVM	0.97±0.0011	0.97±0.0011	2.1749e-01	8.3027e-01	0.0973
AUROC	LogisticRegression	0.72±0.0084	0.73±0.0086	-9.8101e-01	3.3959e-01	-0.4387
	DecisionTree	0.81±0.0066	0.80±0.0068	9.4222e-01	3.5855e-01	0.4214
	SVM	0.78±0.0092	0.78±0.0081	5.9181e-02	9.5346e-01	0.0265
KAPPA	LogisticRegression	0.20±0.0081	0.22±0.0098	-1.2392e+00	2.3118e-01	-0.5542
	DecisionTree	0.23±0.0118	0.21±0.0105	6.4914e-01	5.2444e-01	0.2903
	SVM	0.17±0.0107	0.18±0.0083	-8.9762e-01	3.8124e-01	-0.4014

Feature Selection

In the feature-selection stage, event-frequency-based data representation with feature selection is conducted, and the results will be compared with the experimental results from Section 6.1, in which the feature selection was not applied. Similar to what has been done in Chapter 5, the random-forest algorithm was selected to reduce the dimensionality of the models built by event-frequency-based data representation. For example, in *I Am Playr* , 4,740 events were experienced by all the players used in the experiment. After processing by the random-forest feature selector, the dimension was reduced to 1,111.9 (Averaged from 10-fold cross-validation). As discussed in Section 4.6, these selected features are those whose importance scores (calculated by random forest) are larger than the mean of the scores of all features.

6.2.2 Experiment Details and Results

I Am Playr

This experiment was first tested on *I Am Playr* . As mentioned just above, the number of events in this game that were taken as features by the event-frequency-based data representation was reduced from 4,740 to 1,111.9. This experiment aims to determine if this process might affect the performance of event-frequency-based data representations when predicting disengagement.

As Table 6.4 indicates, after all measurements across three different classifiers are considered, no significant difference is found between the two versions of the event-frequency-based data representations. Therefore, similar to what has been investigated in first purchasing prediction experiments, TOST was also conducted here for figuring out if the performance of classifiers were significantly affected by the feature selection. Same as before, the interval [-0.5, 0.5] was used in TOST. Results can be found in Table 6.5. As the results indicate, the difference of performance between the two versions of event frequency based data representation is significantly within the predefined interval. This supports the solution found in first purchasing prediction that

Table 6.5: Equivalence testing on the performance for predicting disengagement behaviours with event-frequency-based data representation (with and without feature selection) on the dataset of *I Am Playr* (balanced by the random-undersampling method)

		Event Frequency Based Data Representation	Event Frequency Based Data Representation with Feature Selection	P-Value
AUPRC (disengaging)	LogisticRegression	0.22±0.0082	0.22±0.0080	**3.4618e-04**
	DecisionTree	0.35±0.0097	0.35±0.0158	**1.1494e-02**
	SVM	0.22±0.0122	0.22±0.0103	**4.5653e-03**
AUPRC (non disengaging)	LogisticRegression	0.94±0.0024	0.95±0.0026	**2.3400e-10**
	DecisionTree	0.98±0.0010	0.97±0.0010	**7.1961e-18**
	SVM	0.97±0.0011	0.97±0.0011	**4.0982e-17**
AUROC	LogisticRegression	0.72±0.0084	0.73±0.0086	**4.2915e-03**
	DecisionTree	0.81±0.0066	0.80±0.0068	**3.8189e-04**
	SVM	0.78±0.0092	0.78±0.0081	**6.3591e-04**
KAPPA	LogisticRegression	0.20±0.0081	0.22±0.0098	**1.1519e-02**
	DecisionTree	0.23±0.0118	0.21±0.0105	**1.5217e-02**
	SVM	0.17±0.0107	0.18±0.0083	**9.2460e-03**

feature selection can be used for reducing the dimension of classifiers without affect their performance significantly.

Lyroke

Identical to *Lyroke* , the random-forest feature selector also decreased the number of features used in this game. After processing, the dimensions of the model were dropped to 1,728.9 (averaged from 10-fold cross-validation) from 7,395.

As can be seen from Table 6.6, performance comparisons are similar to what was observed for *I Am Playr*. No significant difference can be seen in the two versions of event-frequency-based data representations. Because of this, the TOST was also conducted in this game for working out the similarity between the two versions of the event frequency based data representation. As can be seen from Table 6.7, the performance difference between the two versions are within the interval [-0.5, 0.5] with a significant probability. This further supports the observations that feature selection was able to reduce the dimension of feature space without affecting the performance significantly,

Race Team Manager

The last game investigated is *Race Team Manager* . The number of events in this game is 464, which is slightly less than the other two products. However, the reduction ratio is similar. After processing by the random-forest algorithm, this number was reduced to 107.7 (averaged from 10-fold cross-validation).

According to Table 6.8, the results are similar to what was observed for the other two games; there is no significant difference in any case. Based on the TOST that was

Table 6.6: Performance for predicting disengagement behaviours with event-frequency-based data representation (with and without feature selection) on the dataset of *Lyroke* (balanced by random undersampling)

		Event Frequency Based Data Representation	Event Frequency Based Data Representation with Feature Selection	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (disengaging)	LogisticRegression	0.26±0.0042	0.26±0.0037	-3.3164e-02	9.7391e-01	-0.0148
	DecisionTree	0.30±0.0091	0.31±0.0069	-5.2692e-01	6.0468e-01	-0.2356
	SVM	0.25±0.0050	0.26±0.0044	-5.4864e-01	5.8999e-01	-0.2454
AUPRC (non disengaging)	LogisticRegression	0.95±0.0019	0.95±0.0017	2.2544e-01	8.2417e-01	0.1008
	DecisionTree	0.97±0.0009	0.97±0.0008	-1.1868e-01	9.0685e-01	-0.0531
	SVM	0.97±0.0006	0.97±0.0003	-5.0246e-01	6.2144e-01	-0.2247
AUROC	LogisticRegression	0.77±0.0044	0.76±0.0035	5.7329e-01	5.7354e-01	0.2564
	DecisionTree	0.78±0.0054	0.78±0.0047	-2.4150e-01	8.1189e-01	-0.1080
	SVM	0.78±0.0035	0.79±0.0021	-6.6926e-01	5.1182e-01	-0.2993
KAPPA	LogisticRegression	0.24±0.0053	0.23±0.0057	4.7872e-01	6.3790e-01	0.2141
	DecisionTree	0.22±0.0077	0.22±0.0059	2.1542e-01	8.3187e-01	0.0963
	SVM	0.21±0.0027	0.21±0.0025	7.6445e-01	4.5451e-01	0.3419

Table 6.7: Equivalence testing on the performance for predicting disengagement behaviours with event-frequency-based data representation (with and without feature selection) on the dataset of *Lyroke* (balanced by the random-undersampling method)

		Event Frequency Based Data Representation	Event Frequency Based Data Representation with Feature Selection	P-Value
AUPRC (disengaging)	LogisticRegression	0.26±0.0042	0.26±0.0037	**6.4968e-08**
	DecisionTree	0.30±0.0091	0.31±0.0069	**9.6367e-04**
	SVM	0.25±0.0050	0.26±0.0044	**1.8668e-06**
AUPRC (non disengaging)	LogisticRegression	0.95±0.0019	0.95±0.0017	**1.5121e-13**
	DecisionTree	0.97±0.0009	0.97±0.0008	**8.1708e-19**
	SVM	0.97±0.0006	0.97±0.0003	**1.0121e-23**
AUROC	LogisticRegression	0.77±0.0044	0.76±0.0035	**1.8213e-07**
	DecisionTree	0.78±0.0054	0.78±0.0047	**2.3695e-06**
	SVM	0.78±0.0035	0.79±0.0021	**1.2731e-09**
KAPPA	LogisticRegression	0.24±0.0053	0.23±0.0057	**1.2319e-05**
	DecisionTree	0.22±0.0077	0.22±0.0059	**9.6698e-05**
	SVM	0.21±0.0027	0.21±0.0025	**2.6617e-10**

Table 6.8: Performance for predicting disengagement behaviours with event-frequency-based data representation (with and without feature selection) on the dataset of *Race Team Manager* (balanced by random undersampling)

		Event Frequency Based Data Representation	Event Frequency Based Data Representation with Feature Selection	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (disengaging)	LogisticRegression	0.27±0.0026	0.26±0.0033	3.8950e-01	7.0147e-01	0.1742
	DecisionTree	0.31±0.0050	0.30±0.0035	1.3676e+00	1.8826e-01	0.6116
	SVM	0.30±0.0031	0.30±0.0031	5.7277e-02	9.5496e-01	0.0256
AUPRC (non disengaging)	LogisticRegression	0.93±0.0018	0.94±0.0018	-1.7382e+00	9.9261e-02	-0.7773
	DecisionTree	0.97±0.0006	0.96±0.0007	3.1955e-01	7.5299e-01	0.1429
	SVM	0.97±0.0005	0.97±0.0005	1.3808e-01	8.9171e-01	0.0618
AUROC	LogisticRegression	0.75±0.0027	0.75±0.0039	-7.5910e-01	4.5762e-01	-0.3395
	DecisionTree	0.79±0.0028	0.79±0.0033	1.0702e-01	9.1595e-01	0.0479
	SVM	0.80±0.0026	0.80±0.0027	8.2876e-02	9.3487e-01	0.0371
KAPPA	LogisticRegression	0.25±0.0044	0.25±0.0049	4.3235e-01	6.7063e-01	0.1934
	DecisionTree	0.25±0.0049	0.25±0.0053	3.0895e-01	7.6091e-01	0.1382
	SVM	0.25±0.0038	0.25±0.0050	2.0910e-01	8.3672e-01	0.0935

also conducted in this game, the results help to further verify that the feature-selection algorithm would not significantly affect the performance of the event-frequency-based data representation while reducing the dimension of the models.

6.2.3 Summary

Experiments in this section show that the performance impacts brought by feature selection is significantly within the pre-defined interval [-0.5, 0.5]. Similar to what was observed in the experiments for predicting first purchases, the experiments conducted for predicting disengagement also indicate that feature selection can help generate more interpretable, less-complex models for further investigations when used together with event-frequency-based data representation.

Compared with the random guess, all previous experiments introduced in this chapter show that event-frequency-based data representation can provide reasonable performance. In addition, it will be more challenging to compare this data-representation method with another state-of-the-art approach for investigating its competency. Given that several works have accomplished similar things in this area, a game-specific data representation is added in the next section for comparison.

6.3 Disengagement Prediction with Event-frequency-based and Game-specific Data Representation

Experiments considered in the previous sections show that the event frequency-based data representation is able to offer a predictive performance that is significantly better than a random classifier with or without feature selection. Since many of the state-of-the-art works which applied game-specific data representations might also be able to achieve promising performance for predicting similar purposes, the event-frequency-based data representation could become more valuable only if it is able to provide a better generality while still maintaining a competitive performance. As has been introduced in Section 4.5.1, the event-

Table 6.9: Equivalence testing on the performance for predicting disengagement behaviours with event-frequency-based data representation (with and without feature selection) on the dataset of *Race Team Manager* (balanced by the random-undersampling method)

		Event Frequency Based Data Representation	Event Frequency Based Data Representation with Feature Selection	P-Value
AUPRC (disengaging)	LogisticRegression	0.27±0.0026	0.26±0.0033	**1.3552e-09**
	DecisionTree	0.31±0.0050	0.30±0.0035	**2.6106e-06**
	SVM	0.30±0.0031	0.30±0.0031	**1.5096e-09**
AUPRC (non disengaging)	LogisticRegression	0.93±0.0018	0.94±0.0018	**1.3643e-12**
	DecisionTree	0.97±0.0006	0.96±0.0007	**1.9073e-21**
	SVM	0.97±0.0005	0.97±0.0005	**3.9924e-23**
AUROC	LogisticRegression	0.75±0.0027	0.75±0.0039	**1.3615e-08**
	DecisionTree	0.79±0.0028	0.79±0.0033	**1.1708e-09**
	SVM	0.80±0.0026	0.80±0.0027	**1.1789e-10**
KAPPA	LogisticRegression	0.25±0.0044	0.25±0.0049	**1.2511e-06**
	DecisionTree	0.25±0.0049	0.25±0.0053	**2.9688e-06**
	SVM	0.25±0.0038	0.25±0.0050	**4.1425e-07**

frequency-based data representation is said to be competitive when it could provide significantly better or at least no significance can be found in most cases while being compared with another game-specific data representation. In this section, one of the classical game-specific data representations introduced in the work by Runge et al. (2014) is added for comparison. Experiments introduced in this section follow the paths shown in Figure 6.5:

6.3.1 Experiment Information

Unchanged Experiment Settings

To keep the experiments in this sections focused on their investigation targets, most of the experiment settings will remain the same. At first, all three games will be tested to show that the conclusion from this experiment is applicable not just to some specific situations. The experiments in this section aim to predict disengagement behaviour, and the balancing method (random undersampling) that was applied to balance the resultant class distribution remains unchanged. To avoid possible effects, feature selection was not used in the experiments of this section. Finally, all the applied classification algorithms and evaluation approaches were kept the same.

Data Representation

The research target of the experiments in this section are in the stage of data representation. As Figure 6.5 shows, in addition to event-frequency-based data representation, another data-representation method was added which is game specific. This game-specific data-representation method originally appeared in the study by Runge et al. (2014), which aims to predict another players' behaviour called churn. The churn behaviour is another labelling method that is slightly different from disengagement.

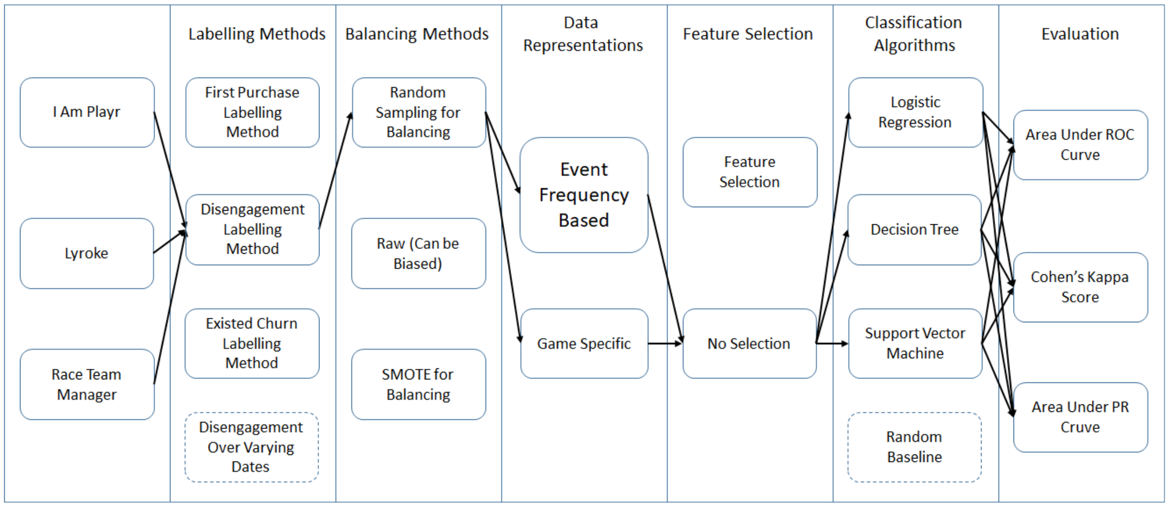

Figure 6.5: Experiment of Disengagement Prediction

Table 6.10: The availability of the game-specific features used in the study by Runge et al. (2014)

	I Am Playr	*Lyroke*	*Race Team Manager*
Rounds played	Yes	Yes	Yes
Accuracy	ND	Yes	Yes
Invites sent	Yes	Yes	Yes
Days in game	Yes	NA	NA
Last Purchase	Yes	Yes	Yes
Days since last purchase	Yes	Yes	Yes

Details of it are introduced in Section 6.4. The reason for choosing this game-specific data-representation method for comparison is that most of the features utilised in the work by Runge et al. (2014) can be found in the game data used in this study. The details of the situation can be found in Table 6.10. The notation in this table is the same as in Table 5.1: *ND* stands for 'not defined in the current game context', *NA* for 'not available' and *Yes* for 'available'. As can be seen from Table 6.10, though several features applied in this data-representation method can generally be found to some extent, some features are missing from the three games to be investigated. That is, 'Accuracy' is missing from *I Am Playr* and 'Days in the game' is missing from both *Lyroke* and *Race Team Manager* .

6.3.2 Experiment Details and Results

I Am Playr

As described in Section 6.1, the dataset labelled by the disengagement is shown to be biased toward the negative class (non-disengaging players). Because of this, the random-undersampling method is included for training classifiers to avoid bias. To determine if event-frequency-based data representation is able to achieve competitive performance, the raw game dataset was represented by both event-frequency-based and game-specific data representation. With event-frequency-based data representation

Table 6.11: Performance for predicting disengagement behaviours with both event-frequency-based and a game-specific data representation on the dataset of *I Am Playr* (balanced by random undersampling)

		Event Frequency Based Data Representation	Game-specific Data Representation	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (disengaging)	LogisticRegression	**0.22±0.0082**	0.18±0.0036	4.4538e+00	3.0672e-04	1.9918
	DecisionTree	**0.35±0.0097**	0.21±0.0059	1.1842e+01	6.2474e-10	5.2958
	SVM	0.22±0.0122	0.22±0.0086	2.6412e-02	9.7922e-01	0.0118
AUPRC (non disengaging)	LogisticRegression	0.94±0.0024	**0.96±0.0021**	-4.7246e+00	1.6911e-04	-2.1129
	DecisionTree	**0.98±0.0010**	0.97±0.0010	4.3684e+00	3.7047e-04	1.9536
	SVM	0.97±0.0011	0.97±0.0009	2.4979e+00	2.2405e-02	1.1171
AUROC	LogisticRegression	0.72±0.0084	0.73±0.0060	-9.1878e-01	3.7036e-01	-0.4109
	DecisionTree	**0.81±0.0066**	0.76±0.0062	4.6070e+00	2.1886e-04	2.0603
	SVM	0.78±0.0092	0.77±0.0060	1.0227e+00	3.2000e-01	0.4574
KAPPA	LogisticRegression	**0.20±0.0081**	0.17±0.0071	3.1280e+00	5.8114e-03	1.3989
	DecisionTree	**0.23±0.0118**	0.18±0.0063	3.1165e+00	5.9606e-03	1.3937
	SVM	0.17±0.0107	0.19±0.0087	-1.1920e+00	2.4875e-01	-0.5331

applied, there were 4,740 features (events) to be used as the feature space, whereas there were five features to be used by the game-specific data representation.

Table 6.11 shows the results of comparing the differences in performance of the two different data-representation methods. The notations in the table are the same as in the previous experiments. As can be seen, there are six of 12 cases in which event-frequency-based data representation achieved significantly better results than game-specific data representation and five of 12 cases in which there was no significant difference can be found between both data representations. This suggests that event-frequency-based data representation is able to provide competitive performance. The only exceptional case happens when the logistic regression is applied and the area under the PR curve (non-disengaging as positive) is used as the evaluation metric. This matches the findings from the work by Runge et al. (2014), which suggests that logistic regression works best with their selection of features.

As a sub-summary, the results found for *I Am Playr* indicate that the event-frequency-based data representation could bring competitive accuracy to the prediction of disengagement trends. To ensure that this solution can still hold for other cases, the same experiment was also tried on *Lyroke* .

Lyroke

As mentioned above, to further verify the experimental results observed for *I Am Playr*, the same experiment conducted for *Lyroke* is introduced in this section. As was mentioned concerning previous experiments, the dataset labelled with the disengagement-labelling method has also been shown to be biased. Based on the behaviours of players, there were 7,395 events utilised as the feature space when event-frequency-based data representation was applied. As with *I Am Playr* , five features were used when the game-specific data representation was applied.

The results for predicting disengagement with both data-representation methods can be found in Table 6.12, where the notations are the same as for the others. As can be seen from the table, the performance comparison is rather one-sided, as the event-frequency-based data representation did well for the prediction in *Lyroke*. For all cases in this

Table 6.12: Performance for predicting disengagement behaviours with event-frequency-based and game-specific data-representation methods on the dataset of *Lyroke* (balanced by random undersampling)

		Event Frequency Based Data Representation	Game-specific Data Representation	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (disengaging)	LogisticRegression	**0.26±0.0042**	0.16±0.0028	1.9282e+01	1.8095e-13	8.6230
	DecisionTree	**0.30±0.0091**	0.21±0.0049	8.7198e+00	7.0348e-08	3.8996
	SVM	**0.25±0.0050**	0.22±0.0026	5.6468e+00	2.3418e-05	2.5253
AUPRC (non disengaging)	LogisticRegression	**0.95±0.0019**	0.93±0.0015	8.4931e+00	1.0349e-07	3.7982
	DecisionTree	**0.97±0.0009**	0.95±0.0005	1.4680e+01	1.8471e-11	6.5653
	SVM	**0.97±0.0006**	0.95±0.0005	1.6291e+01	3.2141e-12	7.2855
AUROC	LogisticRegression	**0.77±0.0044**	0.63±0.0052	1.9087e+01	2.1540e-13	8.5361
	DecisionTree	**0.78±0.0054**	0.70±0.0025	1.2769e+01	1.8439e-10	5.7107
	SVM	**0.78±0.0035**	0.73±0.0025	1.2445e+01	2.8024e-10	5.5655
KAPPA	LogisticRegression	**0.24±0.0053**	0.08±0.0040	2.2412e+01	1.3374e-14	10.0230
	DecisionTree	**0.22±0.0077**	0.13±0.0026	1.0654e+01	3.3412e-09	4.7645
	SVM	**0.21±0.0027**	0.14±0.0019	1.8872e+01	2.6175e-13	8.4398

table, event-frequency-based data representation was able to achieve significantly better predictive performance. The only exception happens when SVM is applied as the classier and the area under the PR curve (non-disengaging as positive) is used for measuring the performance. As results indicate, the observation is similar to what was seen for *I Am Playr*: the event-frequency-based data representation is able to provide competitive results compared with those achieved by game-specific data representation. Finally, in the next section, another game, *Race Team Manager* , which was developed by Bigbit Ltd is also used for investigation.

Race Team Manager

Finally, the experiment was also conducted on a racing game: *Race Team Manager* . In this game, the ratio between disengaging players and non-disengaging players is also biased. Therefore, the random-undersampling method is also included in the training process. After the event-frequency-based data representation was applied, there were 464 events selected to represent the dataset. As in the previous experiments, five features were utilised by the game-specific data representation.

Table 6.13 indicates that there is no significant difference found between the performance brought by both data representations in most cases. First, as can be seen from the table, in three cases of 12, event-frequency-based data representation still achieves significantly ($p<0.01$) better performance than the game-specific data representation and eight of 12 cases in which both data-representation methods work more closely with each other. This gives more support to the previous conclusion: that event-frequency-based data representation is able to achieve competitive performance in comparison with other state-of-the-art approaches. Their similarity can be confirmed by the p-values and the effect sizes. The only exception occurred when logistic regression was used as the classifier and the performance was measured by the area under the PR curve (non-disengaging players as positive). This matches the findings made for *Lyroke*: that the game-specific data representation works best with the logistic-regression algorithms. Therefore, though the results achieved by event-frequency-based data representation are not significantly better than the game-specific results in most cases (as was observed in the other two games), event-frequency-based data representation still

Table 6.13: Performance for predicting disengagement behaviours with event-frequency-based and game-specific data-representation methods on the dataset of *Race Team Manager* (balanced by random undersampling)

		Event Frequency Based Data Representation	Game-specific Data Representation	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (disengaging)	LogisticRegression	**0.27±0.0026**	0.24±0.0051	5.1009e+00	7.4671e-05	2.2812
	DecisionTree	0.31±0.0050	0.29±0.0042	2.5134e+00	2.1694e-02	1.1240
	SVM	0.30±0.0031	0.30±0.0054	-6.6658e-01	5.1349e-01	-0.2981
AUPRC (non disengaging)	LogisticRegression	0.93±0.0018	**0.95±0.0008**	-1.0152e+01	7.0793e-09	-4.5400
	DecisionTree	0.97±0.0006	0.96±0.0039	1.0774e+00	2.9551e-01	0.4818
	SVM	0.97±0.0005	0.97±0.0007	1.4780e+00	1.5670e-01	0.6610
AUROC	LogisticRegression	**0.75±0.0027**	0.73±0.0038	3.0247e+00	7.2831e-03	1.3527
	DecisionTree	0.79±0.0028	0.79±0.0037	-5.4226e-01	5.9429e-01	-0.2425
	SVM	0.80±0.0026	0.80±0.0037	4.7206e-01	6.4256e-01	0.2111
KAPPA	LogisticRegression	**0.25±0.0044**	0.17±0.0037	1.4500e+01	2.2693e-11	6.4847
	DecisionTree	0.25±0.0049	0.24±0.0052	6.6700e-01	5.1323e-01	0.2983
	SVM	0.25±0.0038	0.25±0.0060	-8.7974e-01	3.9060e-01	-0.3934

achieves competitive results for predicting disengagement behaviour.

6.3.3 Summary

A summarised chart can be found in Figure 6.6. To recap, this chart is a summary of performance across the previous experiments in the three commercial games. For each measurement, there are three stacked bars, and they are the number of cases where the event-frequency-based data representation achieved significantly better performance (labelled as EF), the number of cases no significant difference are found between both data representations (labelled as ND), and the number of case where the game-specific data representation achieved significantly better performance (labelled as GF). Each stacked bar is formed by the performance brought by the three applied algorithms. As can be seen from the figure, in all cases across the three games, event-frequency-based data representation was able to perform competitively for predicting disengagement while being compared to a state of the art game-specific data representation. This is because in most cases, it can provide significantly better performance or there is no significant performance difference found between the event-frequency-based data representation and the game-specific data representation.

Combining with its performance for predicting the first purchase behaviour, this new-data representation method can be considered a competitive alternative for player-behaviour modelling where there is a requirement of generality for various predictive purposes in multiple games.

Though event-frequency-based data representation has achieved competitive performance in most cases so far, the complexity of the model might sometimes be an issue in some extreme situations. During all our experiments, there was a challenging predictive purpose which was affected by this shortcoming. Details of it are shown in the next section.

6.4 Churn Prediction

As event frequency-based data representation has been successfully used to predict both first purchases and disengagement across three different games, a challenging experiment predicting churn-labelling method is used in this section to show some limitations of this

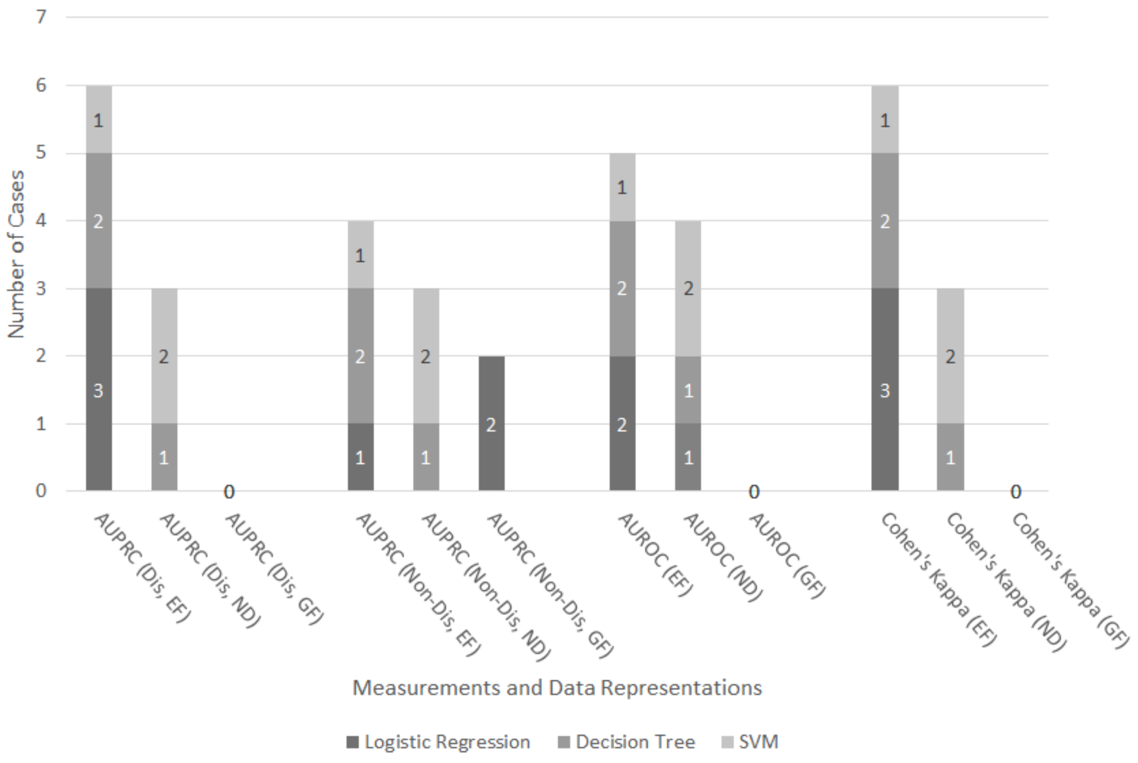

Figure 6.6: The number of cases where methods achieve significantly better performance and the number of cases where there is no difference found for predicting disengagement behaviours

data-representation method. As has been mentioned above, event-frequency-based data representation is a highly dimensional method which sometimes may lead to complex predictive models. Therefore, as discussed in Section 4.9, when the number of training samples is too small compared with the number of events selected for building models, an unreliable classifier might be built–either because the data cannot cover all the important patterns (Raudys et al., 1991) or because of overfitting. In other words, when this happens, the model may not be able to provide good performance for predicting unseen situations (Rokach and Maimon, 2014).

During the course of my study, the churn-labelling method proposed by Runge et al. (2014) was offered as a challenging experiment that provided smaller datasets for training classifiers with the higher-dimensional event-frequency-based data representation. This is because, according to the definition introduced in Section 6.4.1, the labelling method focuses only on highly valued (paying and highly active) players. This largely reduces the number of training samples. Because of this, experiments in this section are expected to be challenging and may affect the event-frequency-based data representation (higher-dimensional) more than the game-specific data representation (lower-dimensional). Experiments that were tested in this section follow the path shown in Figure 6.7.

6.4.1 Experiment Information

Unchanged experiments

Experiments conducted in this section aim to investigate the performance of event-

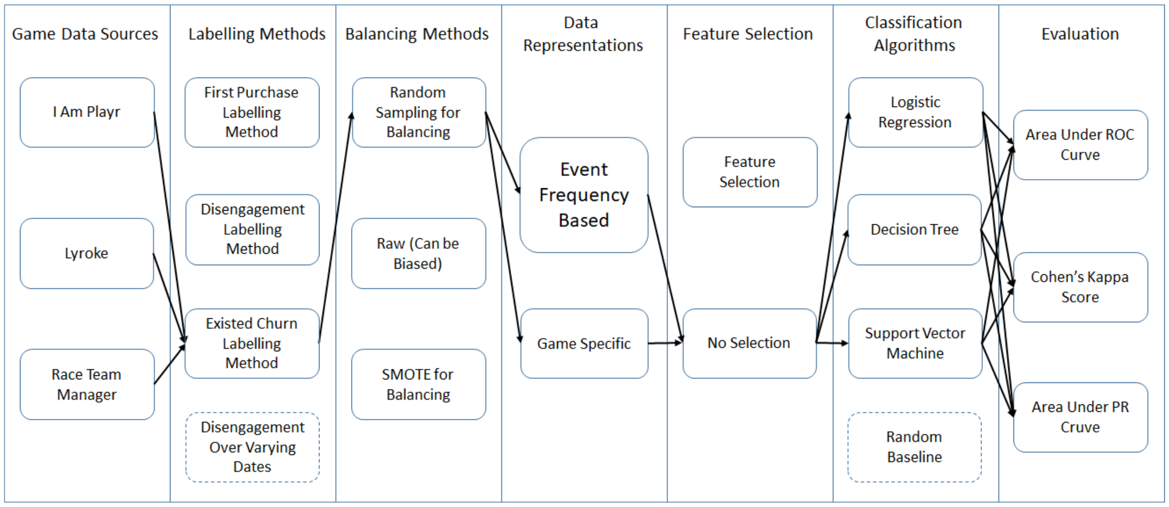

Figure 6.7: Experiment of First-purchase prediction

frequency-based data representation for predicting churn. Therefore, except for the labelling methods, experimental settings such as game-data sources, balancing methods, data representation, feature selection, classification algorithms and evaluation approaches stay the same.

Churn Labelling Method

As mentioned in Section 3.5, another disengagement concept, named *churn*, has been investigated by several other researchers. Although there are different ways to define *churn* (Borbora and Srivastava, 2012; Hadiji et al., 2014; Kawale et al., 2009; Runge et al., 2014), they all refer to a player leaving a game. Building predictive models for this behaviour is valuable because they can be used to forecast when some player is going to make a decision to leave. Based on these predictions, game companies are able to take extra care on these churning players and try to prevent their leaving by pre-emptive measures.

As was mentioned concerning the experiments for comparing data representations, the work by Runge et al. (2014) is among the previous papers which aim to predict churn behaviour. In their definition of churn, only high-valued players are considered to be valid predictive objectives. More precisely, a player who is active and highly valued is said to be churning on a specific day (day 0) if he/she starts 14 consecutive days of inactivity from any days between day 0 and day 6. This definition contains three conditions. First, to be considered, a player has to be a high-value player. A player is said to be highly valued if he/she is in the top 10% of the players who are sorted by the revenue generated. In addition, for a player to be considered, he/she must also be an active player. This is defined by observing whether a player has played the game at least once between days-14 and day-1. Finally, the last condition, which focuses on players who start 14 consecutive days of inactivity from any days between day 0 and day 6, is depicted in Figure 6.8.

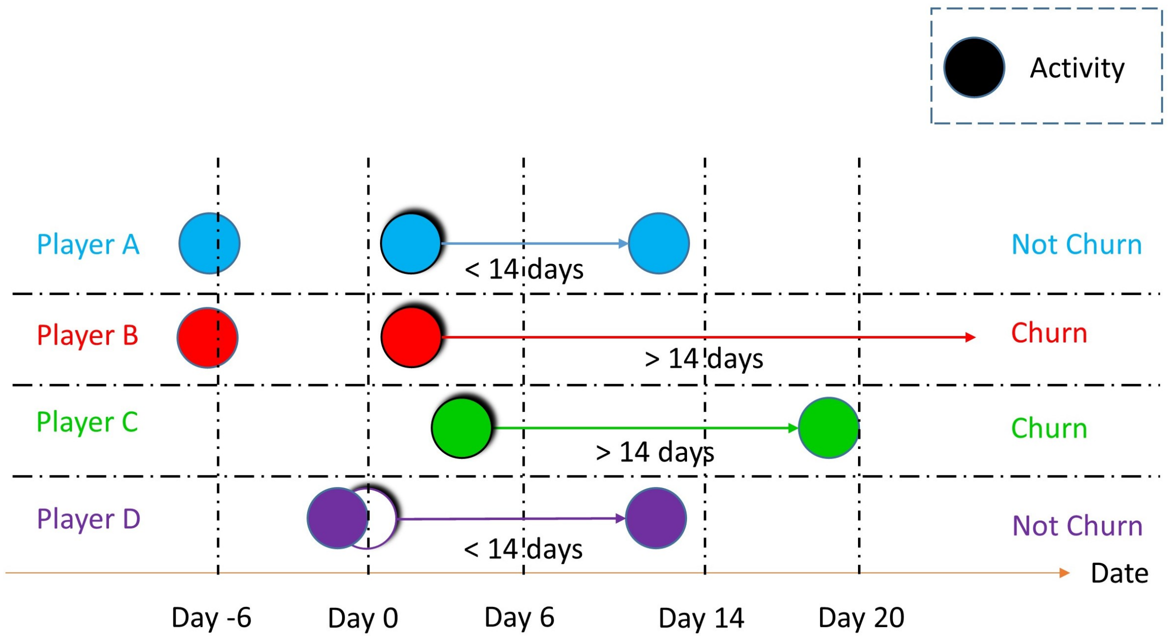

Figure 6.8: Churn Definition by Runge et al. (2014). Player A is not churning because he/she has an activity point at around Day 2 and his/her next activity occurs only at around Day 12, and 12 - 2 = 10 < 14. Players B and C are both churning, because more than 14 days have passed since their last activity in period, '$Day0 - Day6'$, and next activity after Day 6. Player D is a special case: He/she has an activity at Day 1 but no activity in the period '$Day0 - Day6'$. In this case, an activity will be assumed at Day 0. Depending on that, there are fewer than 14 days between Day 0 and Player D's next activity, so Player D is not churning.

6.4.2 Experiment Details and Results

I Am Playr

In *I Am Playr* , after being labelled by the churn-labelling method, there are 132 churning users and 124 non-churning ones. Compared to previous experiments, the balancing ratio is better but the number of data samples is relatively small. In this section, since the random-undersampling method is used for balancing, the number is lower during the training of classifiers. More than the number of samples, there are 2,394 events that have been experienced by the event-frequency-based data representation for representing the behaviours of these players. This number gives this experiment a high chance of encountering problems caused by smaller datasets. Regarding game-specific data representation, five features were extracted from the game data, as in the previous experiment.

Results for predicting churn (balanced) in *I Am Playr* are shown in Table 6.14. According to the table, it is clear that event-frequency-based data representation does not work as well as in previous experiments. Compared to previous experiments in which training examples were sufficient, the performance brought by event-frequency-based data representation has been affected. In addition, lower-dimensional game-specific method works significantly better ($p<0.01$) than event-frequency-based data representation in three cases of 12. All of these cases occurred when logistic regression was applied as the

Table 6.14: Performance for predicting churn behaviours with both event-frequency-based and a game-specific data-representation methods for the dataset of *I Am Playr* (balanced by random undersampling)

		Event Frequency Based Data Representation	Game-Specific Data Representation	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (churning)	LogisticRegression	0.50±0.0146	**0.73±0.0330**	-6.2042e+00	7.4388e-06	-2.7746
	DecisionTree	0.69±0.0198	0.71±0.0294	-6.1259e-01	5.4781e-01	-0.2740
	SVM	0.66±0.0339	0.63±0.0359	4.9108e-01	6.2931e-01	0.2196
AUPRC (non churning)	LogisticRegression	0.54±0.0203	**0.71±0.0387**	-3.7367e+00	1.5101e-03	-1.6711
	DecisionTree	0.72±0.0302	0.68±0.0315	7.4615e-01	4.6521e-01	0.3337
	SVM	0.47±0.0495	0.53±0.0366	-9.8403e-01	3.3814e-01	-0.4401
AUROC	LogisticRegression	0.50±0.0254	**0.72±0.0334**	-4.8367e+00	1.3236e-04	-2.1630
	DecisionTree	0.66±0.0324	0.71±0.0245	-1.0217e+00	3.2046e-01	-0.4569
	SVM	0.51±0.0283	0.57±0.0365	-1.2560e+00	2.2517e-01	-0.5617
KAPPA	LogisticRegression	0.09±0.0597	0.34±0.0607	-2.8311e+00	1.1073e-02	-1.2661
	DecisionTree	0.29±0.0558	0.36±0.0517	-8.8857e-01	3.8595e-01	-0.3974
	SVM	0.06±0.0297	0.07±0.0515	-2.3972e-01	8.1326e-01	-0.1072

Table 6.15: Performance for predicting churn behaviours using both event-frequency-based data representation (balanced by random undersampling) and a random guess on the dataset of *I Am Playr*

		Event Frequency Based Data Representation	Random Classifier	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (churning)	LogisticRegression	0.50±0.0146	0.52	-1.2287e+00	2.5033e-01	-0.5495
	DecisionTree	**0.69±0.0198**	0.52	8.2906e+00	1.6626e-05	3.7077
	SVM	**0.66±0.0339**	0.52	4.0192e+00	3.0218e-03	1.7974
AUPRC (non churning)	LogisticRegression	0.54±0.0203	0.48	2.5345e+00	3.1999e-02	1.1335
	DecisionTree	**0.72±0.0302**	0.48	7.2755e+00	4.6859e-05	3.2537
	SVM	0.47±0.0495	0.48	-2.9400e-01	7.7543e-01	-0.1315
AUROC	LogisticRegression	0.50±0.0254	0.50	1.0078e-01	9.2194e-01	0.0451
	DecisionTree	**0.66±0.0324**	0.50	4.7352e+00	1.0661e-03	2.1176
	SVM	0.51±0.0283	0.50	4.2816e-01	6.7860e-01	0.1915
KAPPA	LogisticRegression	0.09±0.0597	0.00	1.3535e+00	2.0891e-01	0.6053
	DecisionTree	**0.29±0.0558**	0.00	4.9493e+00	7.9203e-04	2.2134
	SVM	0.06±0.0297	0.00	1.8477e+00	9.7713e-02	0.8263

classifier. Except for these, in all other cases, there has been no significant difference in the performance achieved by both event-frequency-based data representation and the game-specific data representation. Because neither data-representation method produced results as good as those obtained in previous experiments that utilized more data samples for training, the results from Table 6.14 are also compared to a random classifier.

The results are shown in Table 6.15 and Table 6.16. As can be seen, there are five cases where the classifiers trained with the event-frequency-based data representation behaved significantly better than random classifiers. Although the classifiers trained with the game-specific data representation did better with eight cases in which classifiers behave significantly better than random classifiers, there ae still four cases when there is no significant different found between it and random classifiers. Compared with the results of previous experiments, this result suggests that the classification problem might be more challenging with limited data samples for training and that it

Table 6.16: Performance for predicting churn behaviours using both game-specific data representation (Balanced by random undersampling) and a random guess on the dataset of *I Am Playr*

		Game-Specific Data Representation	Random Classifier	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (churning)	LogisticRegression	**0.73±0.0330**	0.52	6.2418e+00	1.5112e-04	2.7914
	DecisionTree	**0.71±0.0294**	0.52	6.3089e+00	1.3950e-04	2.8214
	SVM	0.63±0.0359	0.52	3.1172e+00	1.2374e-02	1.3941
AUPRC (non churning)	LogisticRegression	**0.71±0.0387**	0.48	5.5420e+00	3.6012e-04	2.4785
	DecisionTree	**0.68±0.0315**	0.48	5.9494e+00	2.1550e-04	2.6606
	SVM	0.53±0.0366	0.48	1.2576e+00	2.4019e-01	0.5624
AUROC	LogisticRegression	**0.72±0.0334**	0.50	6.1520e+00	1.6831e-04	2.7513
	DecisionTree	**0.71±0.0245**	0.50	7.9494e+00	2.3285e-05	3.5551
	SVM	0.57±0.0365	0.50	1.9230e+00	8.6633e-02	0.8600
KAPPA	LogisticRegression	**0.34±0.0607**	0.00	5.3021e+00	4.9247e-04	2.3712
	DecisionTree	**0.36±0.0517**	0.00	6.6496e+00	9.3779e-05	2.9738
	SVM	0.07±0.0515	0.00	1.3416e+00	2.1258e-01	0.6000

is beneficial to apply data representation with fewer features in this case.

In the next part, to determine whether the observations from this experiment represent a general case when the churn-labelling method was applied, the results from the same experiment tested on a different game (*Lyorke*) are shown for comparison.

Lyroke

As in *I Am Playr* , in *Lyroke* , after being labelled by the churn-labelling method, there are 127 churning users and 103 non-churning ones. Like what was observed with respect to *I Am Playr* , the balancing ratio is better than the disengagement prediction problems, but the number of samples is even smaller than in the churn-prediction experiment in *I Am Playr* and much smaller than in the previous examples, such as in first-purchasing and disengagement prediction.

Since the random-undersampling method is used for balancing, the number will be even smaller during the training process. As for the dimensions (number of features) to be used for event-frequency-based data representations, 4,462 events have been experienced by these players. As before, the five features were used by the game-specific based data representation. Therefore, as in the experiment with *I Am Playr*, the chance for hitting issues brought by smaller datasets is high in this experiment as well.

As can be seen from Table 6.17, event-frequency-based data representation still did not work as well as for other predictive purposes. However, according to both Figure 6.18 and Figure 6.19, the performance achieved by the event-frequency-based data representation and the game-specific data representation used in this game seem to be more stable than that in the *I Am Playr* experiment. In both tables, except for a single case (SVM, measured by AUPRC with non-churning players as positive) while using the event-frequency-based data representation, in other cases, the classifiers trained with both data representation achieved significantly better performance than random classifiers. One possible reason for this result, is that the represented feature space of this game is less variable than that of *I Am Playr* when labelled by the churn-labelling method and that both the inference pattern and its related noises are consistent across cross-validations. Second, according to Table 6.17, the game-specific data representation worked best with SVM in this case; it achieved a significantly better ($p<0.01$)

Table 6.17: Performance for predicting churn behaviours with event-frequency-based and a game-specific data representation on the dataset of *Lyroke* (balanced by random undersampling)

		Event Frequency Based Data Representation	Game-Specific Data Representation	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (churning)	LogisticRegression	0.77±0.0361	0.86±0.0286	-1.8664e+00	7.8362e-02	-0.8347
	DecisionTree	0.82±0.0283	0.82±0.0279	-1.7960e-02	9.8587e-01	-0.0080
	SVM	0.78±0.0198	0.82±0.0171	-1.5492e+00	1.3873e-01	-0.6928
AUPRC (non churning)	LogisticRegression	0.79±0.0317	0.83±0.0335	-6.2301e-01	5.4109e-01	-0.2786
	DecisionTree	0.81±0.0248	0.80±0.0300	4.6290e-01	6.4899e-01	0.2070
	SVM	0.56±0.0358	**0.71±0.0343**	-3.0446e+00	6.9744e-03	-1.3616
AUROC	LogisticRegression	0.78±0.0310	0.84±0.0274	-1.2895e+00	2.1353e-01	-0.5767
	DecisionTree	0.80±0.0272	0.78±0.0310	3.5705e-01	7.2520e-01	0.1597
	SVM	0.71±0.0163	0.77±0.0218	-2.2851e+00	3.4654e-02	-1.0219
KAPPA	LogisticRegression	0.48±0.0631	0.55±0.0556	-7.3721e-01	4.7050e-01	-0.3297
	DecisionTree	0.42±0.0530	0.48±0.0402	-8.2804e-01	4.1849e-01	-0.3703
	SVM	0.24±0.0270	0.38±0.0422	-2.6102e+00	1.7716e-02	-1.1673

Table 6.18: Performance for predicting churn behaviours with both event-frequency-based data representation (balanced by random undersampling) and a random guess at the dataset of *Lyroke*

		Event Frequency Based Data Representation	Random Classifier	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (churning)	LogisticRegression	**0.77±0.0361**	0.55	5.6221e+00	3.2494e-04	2.5143
	DecisionTree	**0.82±0.0283**	0.55	8.8974e+00	9.3766e-06	3.9790
	SVM	**0.78±0.0198**	0.55	1.0878e+01	1.7681e-06	4.8649
AUPRC (non churning)	LogisticRegression	**0.79±0.0317**	0.45	1.0389e+01	2.6019e-06	4.6462
	DecisionTree	**0.81±0.0248**	0.45	1.4024e+01	2.0208e-07	6.2717
	SVM	0.56±0.0358	0.45	2.8396e+00	1.9415e-02	1.2699
AUROC	LogisticRegression	**0.78±0.0310**	0.50	8.6952e+00	1.1307e-05	3.8886
	DecisionTree	**0.80±0.0272**	0.50	1.0384e+01	2.6118e-06	4.6441
	SVM	**0.71±0.0163**	0.50	1.2044e+01	7.4651e-07	5.3861
KAPPA	LogisticRegression	**0.48±0.0631**	0.00	7.2344e+00	4.8978e-05	3.2353
	DecisionTree	**0.42±0.0530**	0.00	7.4712e+00	3.8063e-05	3.3412
	SVM	**0.24±0.0270**	0.00	8.5048e+00	1.3532e-05	3.8035

performance than event-frequency-based data representation did when measured by the area under the PR curve (non-churning as positive). However, as has been examined in the first-purchase prediction experiments, broader parameter-tuning process may bring better results for SVM classifiers. Except for these cases, all the rest of the 12 cases show that there have been no significant difference found between the performance of the two data-representation methods.

The observations made in this experiment were better than those made in the *I Am Playr* experiment. In this experiment, in most cases, there have been no significant difference found between the event-frequency-based data representation and a lower-dimensional game-specific data representation. In the last experiment, the same experimental settings are tested on *Race Team Manager* .

Race Team Manager

In *Race Team Manager* , situations are the same as in the other two games. When

Table 6.19: Performance for predicting churn behaviours with both the game-specific data representation (balanced by random undersampling) and a random guess on the dataset of *Lyroke*

		Game-Specific Data Representation	Random Classifier	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (churning)	LogisticRegression	**0.86±0.0286**	0.55	1.0108e+01	3.2706e-06	4.5207
	DecisionTree	**0.82±0.0279**	0.55	9.0621e+00	8.0709e-06	4.0527
	SVM	**0.82±0.0171**	0.55	1.4972e+01	1.1467e-07	6.6956
AUPRC (non churning)	LogisticRegression	**0.83±0.0335**	0.45	1.0677e+01	2.0682e-06	4.7751
	DecisionTree	**0.80±0.0300**	0.45	1.1007e+01	1.6011e-06	4.9226
	SVM	**0.71±0.0343**	0.45	7.3671e+00	4.2492e-05	3.2947
AUROC	LogisticRegression	**0.84±0.0274**	0.50	1.1794e+01	8.9231e-07	5.2744
	DecisionTree	**0.78±0.0310**	0.50	8.6302e+00	1.2019e-05	3.8595
	SVM	**0.77±0.0218**	0.50	1.1841e+01	8.6235e-07	5.2956
KAPPA	LogisticRegression	**0.55±0.0556**	0.00	9.3204e+00	6.4073e-06	4.1682
	DecisionTree	**0.48±0.0402**	0.00	1.1213e+01	1.3693e-06	5.0146
	SVM	**0.38±0.0422**	0.00	8.5401e+00	1.3086e-05	3.8193

Table 6.20: Performance for predicting churn behaviours using event-frequency-based and game-specific data representations on the dataset of *Race Team Manager* (balanced by random undersampling)

		Event Frequency Based Data Representation	Game-Specific Data Representation	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (churning)	LogisticRegression	0.78±0.0253	**0.91±0.0178**	-3.7992e+00	1.3136e-03	-1.6991
	DecisionTree	0.81±0.0175	0.85±0.0175	-1.2058e+00	2.4349e-01	-0.5393
	SVM	0.83±0.0285	0.89±0.0231	-1.6192e+00	1.2278e-01	-0.7241
AUPRC (non churning)	LogisticRegression	0.36±0.0490	0.51±0.0636	-1.8266e+00	8.4389e-02	-0.8169
	DecisionTree	0.27±0.0329	0.40±0.0533	-1.9251e+00	7.0173e-02	-0.8609
	SVM	0.30±0.0460	0.41±0.0721	-1.2284e+00	2.3512e-01	-0.5494
AUROC	LogisticRegression	0.52±0.0432	**0.73±0.0411**	-3.2945e+00	4.0301e-03	-1.4733
	DecisionTree	0.50±0.0376	0.59±0.0382	-1.6821e+00	1.0982e-01	-0.7522
	SVM	0.55±0.0577	0.69±0.0531	-1.6176e+00	1.2314e-01	-0.7234
KAPPA	LogisticRegression	0.03±0.0417	**0.25±0.0497**	-3.2274e+00	4.6726e-03	-1.4433
	DecisionTree	0.03±0.0375	0.13±0.0243	-2.1333e+00	4.6920e-02	-0.9540
	SVM	-0.04±0.0451	0.17±0.0621	-2.6436e+00	1.6511e-02	-1.1822

the churn-labelling method was applied, 206 churning and 54 non-churning players were labelled. The dataset of this game is even more challenging for highly dimensional event-frequency-based data representation. This is because the dataset is not only biased to one side but also includes few training-data samples. As before, the number of samples will be even less after processing by the random-undersampling balance method. In terms of data representations, 373 events are selected by event-frequency-based data representation while five features are used by the game-specific data representation.

The results for the churn-prediction experiment in *Race Team Manager* are shown in Table 6.20. As can be seen, in most cases (nine of 12), there is no significant difference shown between the two data representations. Similar to what was observed with respect to *I Am Playr* , the game-specific-based data-representation method works best with a logistical regression classifier and achieves significantly better performance when measured by three different measurements. Although classifiers trained with both data-representation methods seem to behave well, because the class distribution in this game is biased, the real performance needs to be verified by comparison with the random classifier.

Table 6.21: Performance for predicting churn behaviours with both the event data representation (balanced by random undersampling) and a random guess on the dataset of *Race Team Manager*

		Event Frequency Based Data Representation	Random Classifier	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (churning)	LogisticRegression	0.78±0.0253	0.79	-3.4326e-01	7.3928e-01	-0.1535
	DecisionTree	0.81±0.0175	0.79	1.1503e+00	2.7966e-01	0.5144
	SVM	0.83±0.0285	0.79	1.1108e+00	2.9544e-01	0.4968
AUPRC (non churning)	LogisticRegression	0.36±0.0490	0.21	2.8909e+00	1.7858e-02	1.2929
	DecisionTree	0.27±0.0329	0.21	1.7964e+00	1.0599e-01	0.8034
	SVM	0.30±0.0460	0.21	1.8457e+00	9.8019e-02	0.8254
AUROC	LogisticRegression	0.52±0.0432	0.50	4.6467e-01	6.5321e-01	0.2078
	DecisionTree	0.50±0.0376	0.50	-1.1862e-01	9.0818e-01	-0.0530
	SVM	0.55±0.0577	0.50	8.5367e-01	4.1543e-01	0.3818
KAPPA	LogisticRegression	0.03±0.0417	0.00	5.9618e-01	5.6575e-01	0.2666
	DecisionTree	0.03±0.0375	0.00	6.5654e-01	5.2791e-01	0.2936
	SVM	-0.04±0.0451	0.00	-8.2127e-01	4.3271e-01	-0.3673

Table 6.22: Performance for predicting churn behaviours using both event-data representation (balanced by random undersampling) and a random guess on the dataset of *Race Team Manager*

		Game-Specific Data Representation	Random Classifier	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (churning)	LogisticRegression	**0.91±0.0178**	0.79	6.1186e+00	1.7524e-04	2.7363
	DecisionTree	0.85±0.0175	0.79	2.8589e+00	1.8813e-02	1.2786
	SVM	**0.89±0.0231**	0.79	3.9517e+00	3.3461e-03	1.7672
AUPRC (non churning)	LogisticRegression	**0.51±0.0636**	0.21	4.5344e+00	1.4172e-03	2.0278
	DecisionTree	**0.40±0.0533**	0.21	3.3728e+00	8.2201e-03	1.5084
	SVM	0.41±0.0721	0.21	2.6342e+00	2.7173e-02	1.1780
AUROC	LogisticRegression	**0.73±0.0411**	0.50	5.2694e+00	5.1428e-04	2.3565
	DecisionTree	0.59±0.0382	0.50	2.2450e+00	5.1423e-02	1.0040
	SVM	**0.69±0.0531**	0.50	3.3146e+00	9.0183e-03	1.4823
KAPPA	LogisticRegression	**0.25±0.0497**	0.00	4.7118e+00	1.1017e-03	2.1072
	DecisionTree	**0.13±0.0243**	0.00	4.9336e+00	8.0927e-04	2.2064
	SVM	0.17±0.0621	0.00	2.6703e+00	2.5608e-02	1.1942

As can be seen from Table 6.21 and Table 6.22, the classifiers trained using event-frequency-based data representation is not able to provide performance that is significantly better than the random classifiers in any case. This indicates that the classifiers are not well trained by an insufficient quantity of training examples. Although the classifiers trained with the game-specific data representation achieve better performance, similar to what was observed for *I Am Playr*, there are still four of 12 cases in which there has been no significant difference found between its performance and that of a random classifier.

6.4.3 Summary

Results from this experiment in three games raise two important points. First, as summarised in Figure 6.9, in most of the cases (29 case out of 36 cases), there have been no

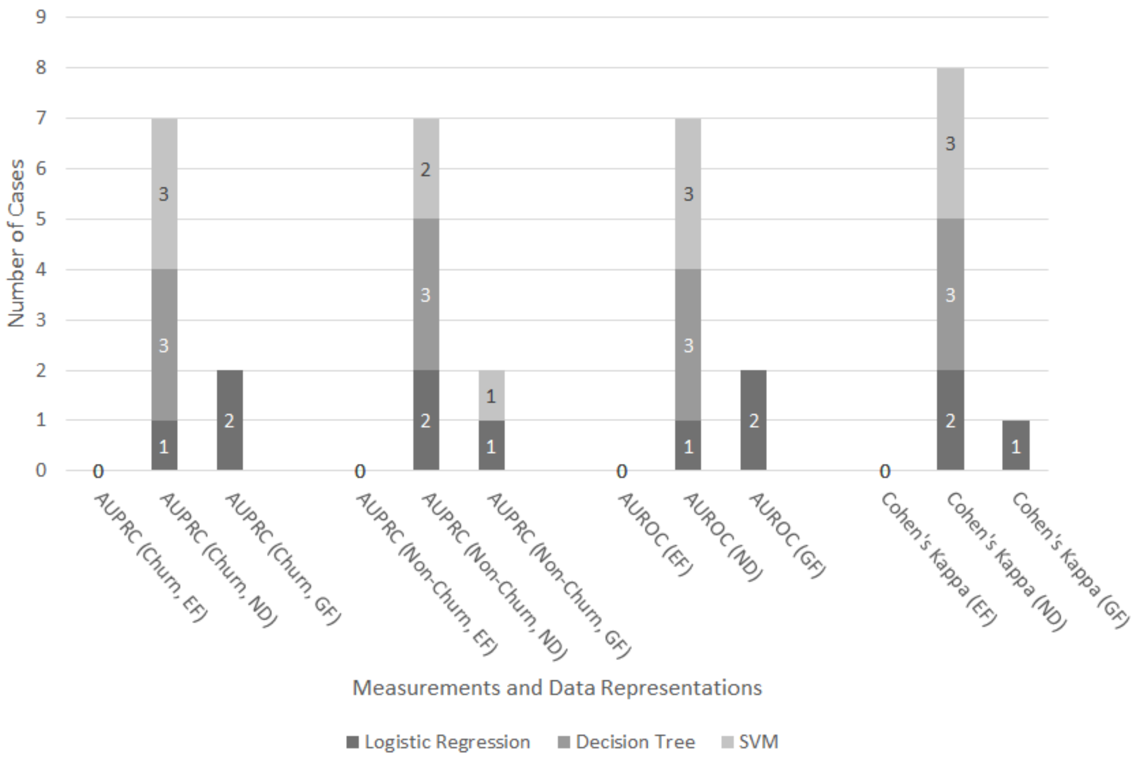

Figure 6.9: The number of cases where methods achieve significantly better performance and the number of cases where there is no difference found for predicting churn behaviours

significant differences can be found between both data representations. The lower dimensional game-specific data representations achieved slightly better performance as there are 7 out of 36 cases where it displayed significantly better performance. Comparing to the previous cases where the event-frequency-based data representation achieved more promising performance, when the dataset is quantitatively insufficient, the models generated can be unreliable, thereby causing their performance to be less effective and to behave like a random classifier. The game-specific data-representation method relies on fewer features to achieve better performance in several cases; but it still suffers from the problem of the limited number of data samples. This suggests that, for training classifiers – especially those represented by higher-dimensional data representations, such as event-frequency-based data representations – quantitatively sufficient training samples are needed to maintain reliability.

6.5 Discussion

This chapter has applied event-frequency-based data representation to predict player disengagement behaviours. The method is able to achieve competitive performance across all three games. Combining these findings with the results obtained from the study of predicting first purchases, event-frequency-based data representation exhibits good generality and can achieve competitive performance for various predictive purposes. These experiments well support the hypothesis made in this study: ***Event-frequency-based data representation can be used to predict player behaviour with supervised learning to provide a sig-***

nificantly better performance than random guess and competitive performance while being compared to other state-of-the-art methods, where applicable.. However, based on the discoveries in the churn-prediction experiments, while the quantity of the dataset is insufficient, the performance that can be brought by event-frequency-based data representation may be affected. As discussed, this is because the number of data samples is too small for training the higher-dimensional classifiers formed by event-frequency-based data representation. Since the limited number of samples is caused by the churn-labelling method in this case, a possible solution is to work out an alternative labelling method which is close to churn but is able to use most of the information from the raw data. This work also conducted some preliminary research into a new labelling method called *disengagement over varying dates* that uses this strategy. Details of this method can be found in Section A.

In addition, taking all three predictive experiments into consideration, a notable phenomenon is that the class distribution might be highly biased (i.e., the majority of players will be categorised into one class) when the players are labelled by any of the following labelling methods: first purchase, disengagement and churn. As discussed in Section 4.4, performance of classifiers trained under biased situations might be affected. Apart from that, as can be seen from the experiments reported in the last two chapters, while the test dataset is biased, the measurements by the area under the PR curve will become less informative, as a random classifier can already achieve a very high (or very low, depending on the bias direction) performance and therefore can hardly be used as a reference to reflect the real performance of the classifiers. Besides, as discussed in Section 4.8.2, the performance measured by Cohen's kappa will be affected as well. A common approach to this type of problem is to apply a balancing method for balancing the classes before modelling to create better classifiers. Although, the random-undersampling method was applied as the default-balancing approach in previous experiments, it is not the only solution for balancing class distributions. In this work, another commonly used method, SMOTE, was also applied. Different from the random-undersampling, SMOTE applies the opposite strategy (oversampling) to achieve the balance. Investigations into whether these balancing methods would help improve the performance of classifiers are shown in the next chapter.

Chapter 7

Biased Player Behaviour Modelling

Previous chapters have used event-frequency-based data-representation for different predictive purposes and has achieved competitive results under the situations where the number of training examples is sufficient. As one may have noticed, a common fact that appeared during these examples is that the class distribution (the ratio between the number of samples in positive and negative classes) resulting from these labelling methods is highly biased before it is processed by the balancing method.

7.1 Bias in Data Mining

The biased-classification problem is common. It occurs in data-mining applications in many areas (Chawla, 2005; Chawla et al., 2002; Grzymala-Busse et al., 2005; Gu et al., 2008; Longadge and Dongre, 2013) and may have negative effects on the reliability of the resultant classifiers–especially for their ability to predict the class with minor examples. In the game context, this often happens because most labelling methods distribute players into groups by way of some pre-defined criteria. For example, in the disengagement-labelling method, the statement, 'if the activity level of a players decreases by at least two levels, the player will be labelled as disengaging; otherwise, he or she is to be labelled as engaging', is an example of categorising players into classes based on the criterion *activity level*. This is similar to first-purchasing prediction in which the splitting condition is whether a player has purchased any item. Likewise, churn definitions by Runge et al. (2014) exhibit the same problem: Players who were active on any date between days 0 and 6 are labelled churning players if no more activity is seen in the next two weeks. A common problem with these labelling methods is that a specific splitting condition has to be defined in the dataset for categorizing samples. However, depending on the facts in different datasets, this line for splitting is not necessarily drawn in the middle, and when it is biased to one side, the resultant classification problem will also be highly biased.

Sometimes, this type of problem can be handled simply. In previous chapters, we introduced a commonly used method, random undersampling, which can be used to balance the class distribution. With this approach, all the experiments that were shown in previous chapters were pre-balanced before they were used to train classifiers. However, a drawback of this random-undersampling method is that, if the dataset is highly biased to one side, or if the original dataset is small, too many data samples will have to be given up to achieve the ideal balance, which, in return, may lead to weak classifiers. As discussed in 4.9 and Section 6.4, a quantitatively insufficient dataset is more likely to lead to weak classifiers when the dimension of the classifiers is high. To solve this problem, a widely used alternative balancing approaches named SMOTE will be introduced and compared to the random-undersampling method. Instead of removing data samples from the major class, because this method applies oversampling strategy to create synthetic data for the minor class, all available information can be kept. Based on the discussions of their comparison, this work also presents a new labelling method named *disengagement over varying dates* for disengagement prediction in Appendix A, preliminary results show that this approach is able to maintain an approximately balanced distribution of resultant classes without losing any samples. At the same time, the classifiers trained under this condition are performing significantly better than random guesses.

Main points in this chapter:

- ◆ investigate the impacts that can be brought by the bias of dataset in disengagement, first purchase and churn predictions. Based on these, the influence brought by the random undersampling can also be analysed,
- ◆ introduction to a classification balancing approach called *SMOTE*,
- ◆ description of the experiment for predicting disengagement and churn while being balanced by SMOTE

7.2 Disengagement prediction – Biased

Chapter 6 of this work shows that event-frequency-based data representations are able to achieve competitive predictive performance for predicting players' disengagement trends. However, the successful results achieved are based on a dataset that has been well balanced with help from the random-undersampling method. In this section, to show the possible negative impacts an imbalanced dataset may bring, the same experiment is conducted on the raw, imbalanced dataset that was labelled by the disengagement-labelling method without any balancing. Based on this experiment, comparisons will enable us to determine whether the random-undersampling method has helped to improve the performance of classifiers. All three games, including *I Am Playr* , Lyorke and *Race Team Manager* , are used for the investigation. The experiments covered can be found in Figure 7.1.

7.2.1 Experiment Information

Most settings in these experiments remain the same as they were in the balanced situations introduced in Chapter 6. As can be seen in Figure 7.1, the only change happens when the balancing approach is selected. Different from before, the experiments in this section compare the situations under both imbalanced (raw) and balanced datasets (with random undersampling). Because the classification problems in this chapter are highly biased, potential performance improvements are expected from applying the random undersampling balancing method. Due to that AUROC is not sensitive to the balance changes, when choosing the evaluation methods, AUPRC and Cohen's kappa score can be more effective in the result tables for showing the difference. A detailed explanation of the reason for this is given in Section 4.8.2. Additionally, Figure 7.1 shows that feature selection has been disabled for both version to get rid of any possible uncontrollable influences.

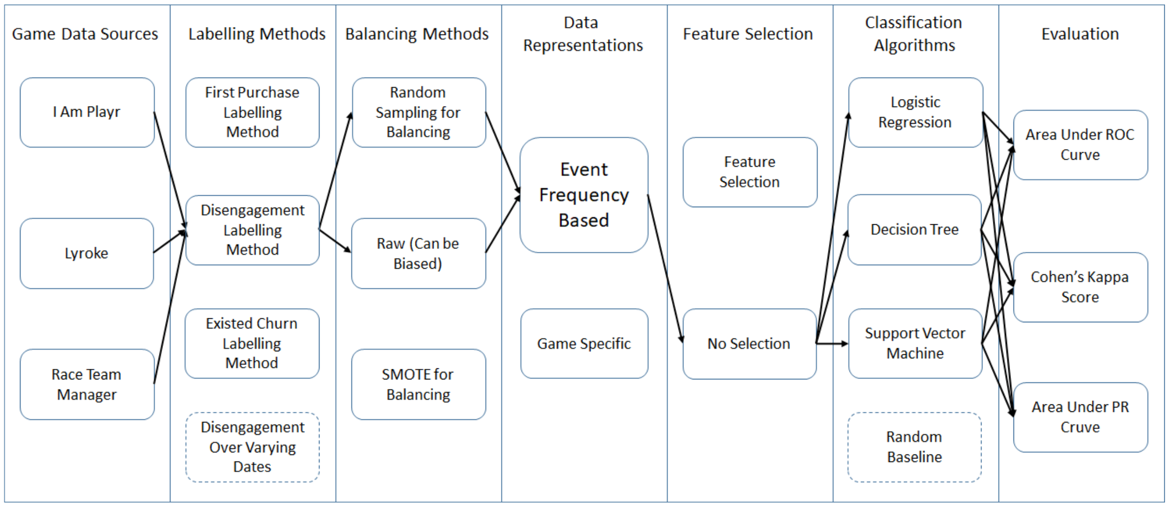

Figure 7.1: Balancing Investigation Experiment of Disengagement Prediction

Table 7.1: Performance for predicting disengagement behaviours with event-frequency-based data representation on the dataset of *I Am Playr* (with and without balancing)

		Event Frequency without Balancing	Event Feature Balanced by Undersampling	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (disengaging)	LogisticRegression	0.19±0.0056	0.22±0.0082	-2.6596e+00	1.5962e-02	-1.1894
	DecisionTree	0.27±0.0138	**0.35±0.0097**	-4.1478e+00	6.0442e-04	-1.8550
	SVM	0.28±0.0522	0.22±0.0122	1.1722e+00	2.5642e-01	0.5242
AUPRC (non disengaging)	LogisticRegression	0.91±0.0020	**0.94±0.0024**	-8.8866e+00	5.3170e-08	-3.9742
	DecisionTree	0.96±0.0018	**0.98±0.0010**	-5.1248e+00	7.0918e-05	-2.2919
	SVM	0.97±0.0011	0.97±0.0011	-7.7043e-01	4.5104e-01	-0.3445
AUROC	LogisticRegression	0.59±0.0077	**0.72±0.0084**	-1.1040e+01	1.9089e-09	-4.9371
	DecisionTree	0.72±0.0142	**0.81±0.0066**	-4.9508e+00	1.0328e-04	-2.2141
	SVM	0.75±0.0097	0.78±0.0092	-2.5738e+00	1.9124e-02	-1.1510
KAPPA	LogisticRegression	0.11±0.0068	**0.20±0.0081**	-7.9530e+00	2.6651e-07	-3.5567
	DecisionTree	0.21±0.0213	0.23±0.0118	-5.8819e-01	5.6371e-01	-0.2630
	SVM	0.00±0.0000	**0.17±0.0107**	-1.4804e+01	1.6056e-11	-6.6207

7.2.2 Experiment Details and Results

I Am Playr

At first, the results of the experiment conducted in *I Am Playr* are shown. As mentioned in Section 6.1.1, 1,354 and 12,044 players are labelled as disengaging and non-disengaging, respectively, in the raw situation. To simplify, the ratio between the two classes is around 1:8.895, which indicates that this classification problem is highly imbalanced. In addition, under this situation, the event-frequency feature space was formed by 4,740 events.

Results of the experiment can be found in Table 7.1. The notations shown are the same as in the experiments of previous chapters. Before diving into the results, to recap, event-frequency-based data representation is able to provide a competitive performance for predicting disengagement under a balanced situation. The results are brought to this table for easier comparison. As can be seen from the result table, in seven of 12 cases, the undersampling method helped to improve the performance of the classifiers

Table 7.2: Performance for predicting disengagement behaviours with event-frequency-based data representation on the dataset of *Lyroke* (with and without balancing)

		Event Frequency without Balancing	Event Feature Balanced by Undersampling	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (disengaging)	LogisticRegression	**0.29±0.0042**	0.26±0.0042	4.1890e+00	5.5148e-04	1.8734
	DecisionTree	0.26±0.0033	**0.30±0.0091**	-4.4558e+00	3.0541e-04	-1.9927
	SVM	0.24±0.0072	0.25±0.0050	-1.1948e+00	2.4767e-01	-0.5343
AUPRC (non disengaging)	LogisticRegression	0.95±0.0011	0.95±0.0019	-5.4212e-01	5.9439e-01	-0.2424
	DecisionTree	0.96±0.0007	**0.97±0.0009**	-7.9837e+00	2.5230e-07	-3.5704
	SVM	0.97±0.0006	0.97±0.0006	-1.3102e+00	2.0659e-01	-0.5860
AUROC	LogisticRegression	0.77±0.0033	0.77±0.0044	8.7409e-01	3.9358e-01	0.3909
	DecisionTree	0.73±0.0042	**0.78±0.0054**	-6.9993e+00	1.5557e-06	-3.1302
	SVM	0.77±0.0035	0.78±0.0035	-1.6372e+00	1.1895e-01	-0.7322
KAPPA	LogisticRegression	0.07±0.0042	**0.24±0.0053**	-2.2786e+01	1.0025e-14	-10.1903
	DecisionTree	0.14±0.0070	**0.22±0.0077**	-7.5927e+00	5.1158e-07	-3.3956
	SVM	0.00±0.0000	**0.21±0.0027**	-7.2414e+01	1.1918e-23	-32.3845

significantly. The results indicate that the classifiers were significantly improved in most cases after the dataset was balanced by random undersampling. While being trained with the raw dataset, the most serious case can be found when SVM was applied, as the classifier and its performance was measured by Cohen's kappa. In this case, its performance is similar to that of a random classifier (which gives 0.0 in Cohen's kappa), which cannot be accepted in practical problems. However, this situation improved significantly when random undersampling was used to process the raw dataset. It is notable that, because the test dataset is biased, the measurements with the area under the ROC curve are more reliable than either the area under the PR curve or Cohen's kappa, as both can be affected by the bias (Jeni et al., 2013). As can be seen from the area under the ROC curve, in two of three cases, the classifiers behave significantly better when balanced by the random-undersampling method. To verify that the phenomenon found in this experiment is unique, the same experiment has been further tested in another two commercial games.

Lyroke

As was seen with respect to *I Am Playr* , the impact of a biased classification is serious. To verify the generality of this issue, the same experiments conducted on *Lyroke* are also introduced in this section. Unlike the balanced version seen in Section 6.1.1, though no balancing method was applied, this classification problem is highly biased to one side. There are only 3,495 disengaging players and 25,012 non-disengaging users (the ratio is around 1:7.16). While the event-frequencies were used to provide the data representation, a total of 7,395 events were used as features for training classifiers.

Table 7.2 shows what happens when event-frequency-based data representation is used to predict disengagement in the raw situation. As with *I Am Playr* , studied in Chapter 6, event-frequency-based data representation was able to provide competitive prediction in *Lyroke* under a balanced situation. However, when the classification problem is highly biased to one side, the performance is much affected. As the table indicates, in six of the 12 cases, the classifiers' performance can be improved by the random-undersampling balancing method. Except for one case (classifier: logistic regression, measurement: area under the PR curve when the disengaging players are positive examples), the random-undersampling method is able either to bring up the performance

Table 7.3: Performance for predicting disengagement behaviours with event-frequency-based data representation on the dataset of *Race Team Manager* (with and without balancing)

		Event Frequency without Balancing	Event Feature Balanced by Undersampling	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (disengaging)	LogisticRegression	0.27±0.0024	0.27±0.0026	4.9298e-01	6.2799e-01	0.2205
	DecisionTree	0.28±0.0029	**0.31±0.0050**	-5.2250e+00	5.7197e-05	-2.3367
	SVM	0.30±0.0050	0.30±0.0031	7.2357e-01	4.7863e-01	0.3236
AUPRC (non disengaging)	LogisticRegression	0.92±0.0025	0.93±0.0018	-2.7320e+00	1.3687e-02	-1.2218
	DecisionTree	0.96±0.0006	**0.97±0.0006**	-8.9669e+00	4.6528e-08	-4.0101
	SVM	0.97±0.0006	0.97±0.0005	1.6936e-01	8.6740e-01	0.0757
AUROC	LogisticRegression	0.73±0.0049	0.75±0.0027	-2.8740e+00	1.0097e-02	-1.2853
	DecisionTree	0.75±0.0031	**0.79±0.0028**	-7.4621e+00	6.5084e-07	-3.3371
	SVM	0.81±0.0030	0.80±0.0026	1.9591e-01	8.4688e-01	0.0876
KAPPA	LogisticRegression	0.05±0.0041	**0.25±0.0044**	-3.2153e+01	2.3534e-17	-14.3791
	DecisionTree	0.13±0.0051	**0.25±0.0049**	-1.5904e+01	4.8238e-12	-7.1124
	SVM	0.01±0.0034	**0.25±0.0038**	-4.4606e+01	6.9706e-20	-19.9482

of classifiers or at least performs as well as the bias cases. In addition, similar to what was observed with respect to *I Am Playr* , the SVM acts like a random classifier when measured by Cohen's kappa in this game. In addition, when the area under the ROC curve is used in the measurements, the random-undersampling method brings significant improvements to the performance of a decision-tree classifier, whereas no significant difference can be noted when the other classifiers are used. This further verifies the observation made in the *I Am Playr* experiment: that when the target is to get a classifier with the ability to predict both classes, classifiers trained on the balanced dataset are more likely to have better or at least similar performance than classifiers trained with the biased dataset. Finally, the last game developed by a different company is also tested in the next section.

Race Team Manager

The same experiment was also conducted in *Race Team Manager* , where 3,416 and 22,957 players showed a trend of disengaging and engaging, respectively. Like the other two experiments, the bias is serious: The ratio is about 1:6.72. After the event-frequency-based data representation was applied, 464 game events could be used as features.

Table 7.3 displays the results of the experiment. Results for predicting first purchases with event-frequency-based data representation are displayed in Table 7.3. Similar to what was found in the study of *Lyroke* , in six of the 12 cases, the performance of classifiers is significantly improved when the dataset is processed by the random-undersampling method. In other words, the classifiers trained under a biased class distribution have fewer chances to provide a reliable performance. As in the other two experiments, SVM still performs like a random classifier when measured by Cohen's kappa in the original, biased dataset. As before, when the target is to achieve a classifier that can be used to predict both classes (with the area under the ROC curve being the prior consideration), the performance of a decision-tree classifier is significantly improved, whereas the other two classifiers behave like they did when they were trained with the raw, imbalanced dataset. This further helps to support the observation that the classifiers trained with a balanced dataset exhibit significantly better performances in most cases. While considering their performance for predicting both

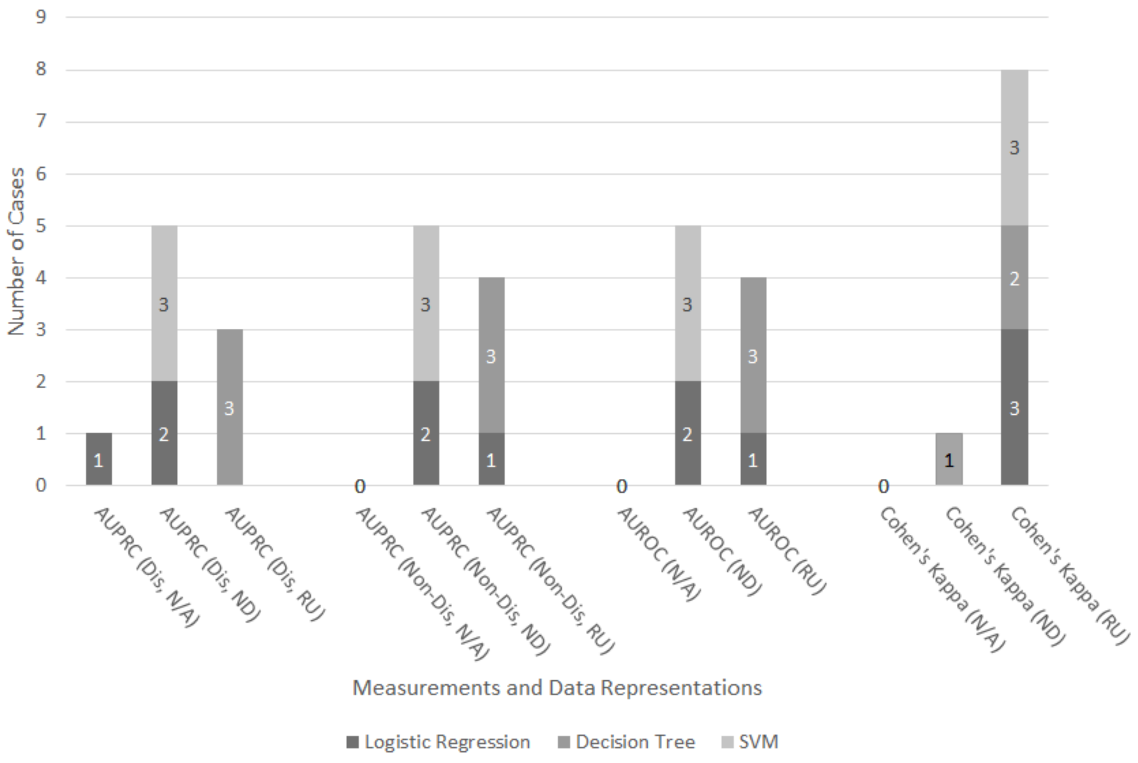

Figure 7.2: The number of cases where methods achieve significantly better performance and the number of cases where there is no difference found for predicting disengagement behaviours with undersampling

classes, classifiers tend to behave like those trained with the imbalanced dataset and behave significantly better in some cases.

7.2.3 Summary

Three experiments were conducted on different commercial games for understanding how a biased classification problem can affect the reliability of the trained classifier. As summarised in Figure 7.2, in most cases of these three experiments, a common fact is that classifiers trained under the raw biased datasets can be significantly improved by a random-undersampling balancing method. This indicates that the bias of class distribution may lead to weak classifiers in several cases. Serious problems can be found when SVM is applied as the classifier and measured by Cohen's kappa. The performance the classifiers achieved, in this case, are similar to those of a random classifier. In addition, while the target is to get a classifier that has the ability to predict both classes (i.e., the area under the ROC curve was considered), classifiers trained with a balanced dataset behave significantly better than those trained with the biased dataset in several cases and behave like them in the rest of the cases. According to the results of the experiment, the classifiers are more likely to achieve better predictive results, and the bias can be effectively improved for predicting disengagement by using a balancing method like random-undersampling.

In the next section, to verify the generality of the observations made in this section, similar experiments will be conducted with respect to another highly biased problem: First-

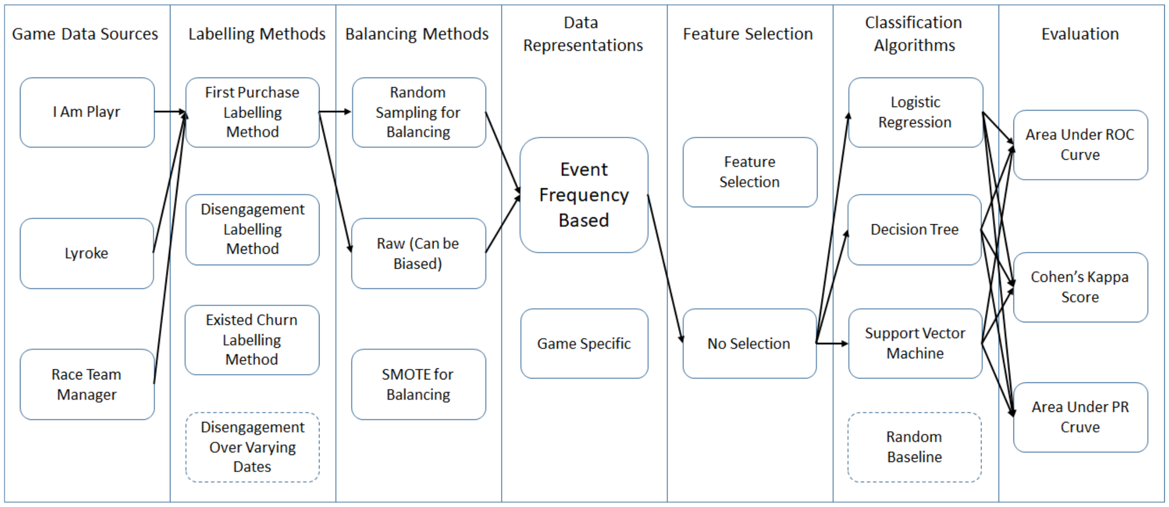

Figure 7.3: Balancing Investigation Experiment of First Purchase Prediction

purchasing prediction.

7.3 First-purchase prediction – Biased

Experiments in the last section show that the random-undersampling method is able to make significant improvements to the quality of models used for predicting the disengagement behaviours in most cases. As another successful predictive task in which event-frequency-based data representation behaved competitively, first-purchasing tasks have been shown to be highly biased in all three of the games we worked on. In this section, the performance of classifiers trained with the raw dataset (highly biased) is compared with the performance of those processed by undersampling to determine whether random undersampling always helps in this highly biased task. Because that the bias in these experiments are serious as well, improvements are also expected from the random undersampling balancing method. The experiments covered are depicted in Figure 7.3.

7.3.1 Experiment Information

While most of the experimental settings used in this section were kept the same as those discussed in Section 5.2.2, as can be seen in the Figure 7.3, a change is made at the stage in which the balancing approach is selected. This is done to determine whether the random-undersampling method can help with the first-purchasing prediction problem.

7.3.2 Experiment Details and Results

I Am Playr

As was explained in Section 5.2.2, the first-purchasing problem is also highly biased. When labelled by the first-purchasing labelling method, as introduced in Section 5.2.2, there were 489 players who at least purchased one item and 88,568 non-paying ones. This ratio is about 1:181.12, which is more serious than that encountered in the datasets considered in the disengagement-prediction problems. After the event-frequency-based

Table 7.4: Performance for predicting first-purchasing behaviours with event-frequency-based data representation on the dataset of *I Am Playr* (with and without balancing)

		Event Frequency without Balancing	Event Feature Balanced by Undersampling	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (first purchase)	LogisticRegression	**0.71±0.0189**	0.39±0.0161	1.2335e+01	3.2349e-10	5.5164
	DecisionTree	**0.76±0.0107**	0.49±0.0075	1.9210e+01	1.9296e-13	8.5909
	SVM	0.24±0.0636	0.07±0.0068	2.5111e+00	2.1800e-02	1.1230
AUPRC (non first purchase)	LogisticRegression	1.00±0.0002	1.00±0.0002	-1.7919e+00	8.9978e-02	-0.8014
	DecisionTree	1.00±0.0001	**1.00±0.0000**	-5.6973e+00	2.1072e-05	-2.5479
	SVM	1.00±0.0001	1.00±0.0000	-1.6818e+00	1.0987e-01	-0.7521
AUROC	LogisticRegression	0.95±0.0101	0.98±0.0049	-2.1638e+00	4.4174e-02	-0.9677
	DecisionTree	0.90±0.0101	**0.97±0.0043**	-5.9087e+00	1.3594e-05	-2.6425
	SVM	0.95±0.0036	0.96±0.0021	-1.5067e+00	1.4924e-01	-0.6738
KAPPA	LogisticRegression	**0.67±0.0119**	0.16±0.0037	3.9600e+01	5.8186e-19	17.7097
	DecisionTree	**0.74±0.0104**	0.12±0.0051	5.0791e+01	6.8506e-21	22.7145
	SVM	0.02±0.0159	**0.07±0.0024**	-3.3961e+00	3.2191e-03	-1.5188

data representation was applied, 2460 events can be included for representing the players' behaviours.

Results of the experiment can be found in Table 7.4, where the notations shown are the same as for the experiments in the previous chapters. The results are surprisingly quite different from what was observed in the disengagement-prediction cases. As can be seen from the table, random undersampling was not able to provide significant differences in most cases. On the contrary, in four of 12 cases, the classifiers trained on the raw dataset behave significantly better, whereas random undersampling improved the performance in only three of 12 cases. And in the cases in which undersampling helped, one of them is less informative because it happened when the area under the PR curve was used, and the non-first purchasers were considered as positive examples, due to the bias, the random guess can even reach a similar score in this case. Although the experiments in Section 5.2.2 show that classifiers trained with the undersampling method are able to achieve significantly better performance than a random classifier, the observation in this experiment issues a warning about using a random-undersampling method. This method involves removing data points from the dataset for balancing, and in a very highly biased dataset like this, too much data needs to be removed. When some important informant is removed, the classifiers might also be negatively impacted. However, notice that although random undersampling did not help as much as it did in the disengagement-prediction tasks, if the target is to get a classifier which is good at predicting both classes (while measured by the area under the ROC curve), the random-undersampling method can still help decision tree achieve significantly better results with the other two cases, as did the classifiers trained on the raw dataset. This observation still matches what has been seen for predicting disengagement.

Lyroke

To see if the problem found in As can seen in *I Am Playr* , the same experiments were also run in *Lyroke* , too. As introduced in Section 5.2.2, when labelled by the first purchasing labelling method, the raw dataset is more biased (with a ratio of 1:594.76) than that of *I Am Playr* . In this game, there are 509 first paying users and 279,829 players who have not purchase any game item. When event-frequency-based

Table 7.5: Performance for predicting first-purchasing behaviours with event frequency-based data representation on the dataset of *Lyroke* (with and without balancing)

		Event Frequency without Balancing	Event Feature Balanced by Undersampling	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (first purchase)	LogisticRegression	**0.49±0.0254**	0.21±0.0145	9.1937e+00	3.2039e-08	4.1116
	DecisionTree	**0.95±0.0080**	0.41±0.0093	4.1697e+01	2.3208e-19	18.6474
	SVM	0.06±0.0125	0.04±0.0025	1.2495e+00	2.2750e-01	0.5588
AUPRC (non first purchase)	LogisticRegression	1.00±0.0000	1.00±0.0001	1.0716e+00	2.9807e-01	0.4792
	DecisionTree	1.00±0.0000	1.00±0.0000	-2.3137e+00	3.2710e-02	-1.0347
	SVM	1.00±0.0000	1.00±0.0000	-1.8741e+00	7.7248e-02	-0.8381
AUROC	LogisticRegression	0.98±0.0030	0.97±0.0048	1.4202e+00	1.7264e-01	0.6351
	DecisionTree	0.96±0.0063	0.97±0.0013	-2.3751e+00	2.8864e-02	-1.0622
	SVM	0.97±0.0058	0.97±0.0007	-1.3897e+00	1.8158e-01	-0.6215
KAPPA	LogisticRegression	**0.48±0.0261**	0.06±0.0013	1.5043e+01	1.2294e-11	6.7272
	DecisionTree	**0.94±0.0075**	0.05±0.0036	1.0292e+02	2.1609e-26	46.0260
	SVM	0.00±0.0000	**0.03±0.0013**	-2.1088e+01	3.8518e-14	-9.4309

data representation was used, a total of 7,832 events were used as features for training classifiers.

Table 7.5 shows how event-frequency-based data representation performs in predicting the first-purchase behaviours in the raw situation. Like what has been observed in *I Am Playr* , random undersampling is not able to make significant improvements to the performance of classifiers. As was the case with *I Am Playr* , there are four of 12 cases in which the classifiers trained on the raw dataset behaved significantly better whereas there is only one case in which random-undersampling significantly helped. This verifies that the random-undersampling method is sometimes risky if too much information needs to be removed. However, like what has been found for *I Am Playr* , if the target was to train classifiers that can predict both classes (measured by the area under ROC), the, though random-undersampling did not help improve the performance of the classifiers significantly, it did not have any negative effects either.

Race Team Manager

Finally, the same experiments have also been done in *Race Team Manager* where there are 511 first purchasers and 170,333 players who did not make any purchases. As in the previous two games, the bias ratio in this game is about 1:333.33, which is also highly biased. When event-frequency-based data representation is applied, 1,531 events are included.

Table 7.6 displays the performance classifiers reached when trained under both the raw dataset and the dataset balanced by random undersampling. Unlike the other experiments, in this experiment, random undersampling helped to improve the performance significantly in four of 12 cases, whereas the classifiers trained on the raw dataset were able to achieve better performances in two of 12 cases. By comparing the results in this game to those of the other two, we can see that whether random-undersampling helps is dependent on the dataset. However, as can be seen, when measured by the area under the ROC curve, in two of the three cases the random-undersampling method was able to bring significant benefits to the performance of the classifiers. This matches the observations made for the other two games.

Table 7.6: Performance for predicting first-purchasing behaviours with event-frequency-based data representation on the dataset of *Race Team Manager* (with and without balancing)

		Event Frequency without Balancing	Event Feature Balanced by Undersampling	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (first purchase)	LogisticRegression	0.23±0.0139	0.18±0.0129	2.4944e+00	2.2570e-02	1.1155
	DecisionTree	0.29±0.0217	0.32±0.0120	-1.2500e+00	2.2732e-01	-0.5590
	SVM	0.08±0.0111	0.09±0.0090	-4.4975e-01	6.5826e-01	-0.2011
AUPRC (non first purchase)	LogisticRegression	1.00±0.0003	1.00±0.0004	-2.0845e+00	5.1630e-02	-0.9322
	DecisionTree	0.99±0.0002	**1.00±0.0000**	-3.0347e+01	6.5414e-17	-13.5718
	SVM	1.00±0.0003	1.00±0.0001	-2.8128e+00	1.1515e-02	-1.2579
AUROC	LogisticRegression	0.86±0.0147	**0.92±0.0091**	-3.3067e+00	3.9231e-03	-1.4788
	DecisionTree	0.62±0.0115	**0.95±0.0029**	-2.6364e+01	7.8077e-16	-11.7904
	SVM	0.94±0.0096	0.96±0.0033	-2.1863e+00	4.2242e-02	-0.9778
KAPPA	LogisticRegression	**0.25±0.0194**	0.07±0.0010	8.5592e+00	9.2409e-08	3.8278
	DecisionTree	**0.31±0.0274**	0.05±0.0014	8.8089e+00	6.0548e-08	3.9395
	SVM	0.02±0.0139	0.06±0.0008	-2.7002e+00	1.4645e-02	-1.2076

7.3.3 Summary

As has been summarised in Figure 7.4, different from the expectations of the research, the experiments described in this section issue a warning concerning the use of the random-undersampling method for balancing classification problems. When the bias is serious, removing too much information is risky in most cases. In addition, whether the random-undersampling method can still help in this case is dependent on individual datasets. However, similar to what was discovered while predicting disengagement behaviours, a common observation made in all three games is that, if the target is to get a classifier that can predict both classes well (measurement by the area under ROC), random undersampling is able to help in most cases and at least not bring significantly negative impacts. Therefore, depending on the predictive target and the datasets, one may consider whether random undersampling is the right technology.

Until now, experiments have shown that random-undersampling is able to help in most cases for predicting disengagement and some cases for predicting first-purchase with risks. However, even for the first-purchase prediction, we have shown in Section 5.2.2 that the performance of classifiers is significantly better than the random classifiers. As discussed in Section 6.4, the real challenging task happens when the total number of samples is not large enough. When this happens, the performance of classifiers created with higher-dimensional data representations might be affected. The experiments of Section 6.4 shown so far are about performance of classifiers trained with dataset that has been balanced with the random undersampling, in the next section, it is meaningful to investigate classifiers that were trained with its raw imbalanced dataset and work out if a random-undersampling method was able to help in the cases when the dataset is relatively small.

7.4 Churn prediction – Biased

As a challenging predictive purpose introduced in Section 6.4, when the event-frequency-based data representation is used for predicting churn, the number of samples seems to become insufficient for this highly dimensional method. This situation becomes even more challenging when the dataset is highly biased (for example, when the churn-labelling method was applied to *Race Team Manager*) and the random-undersampling method was applied.

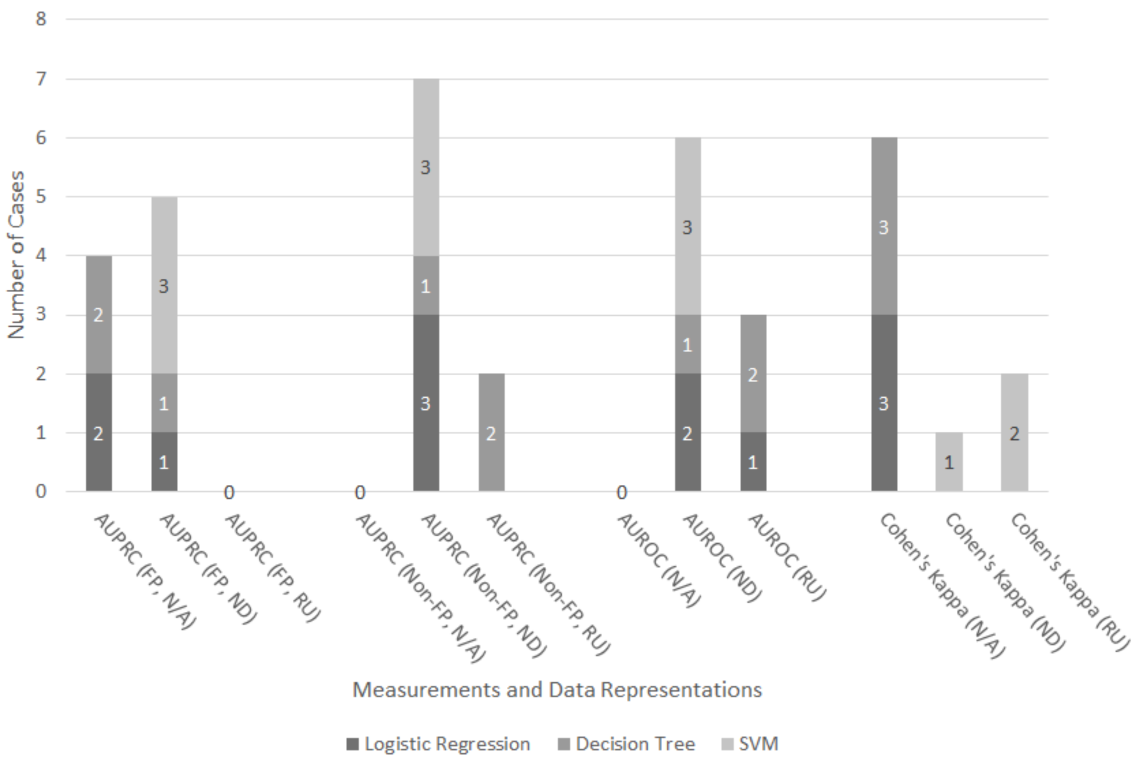

Figure 7.4: The number of cases where methods achieve significantly better performance and the number of cases where there is no difference found for predicting first purchase behaviours with undersampling

In this section, classifiers trained with the raw, imbalanced dataset are compared to the undersampling balanced situations to determine whether the random-undersampling method still helps in this challenging case. As in the previous section, all three games–*I Am Playr* , Lyorke and *Race Team Manager* –are used for the comparison. The experiments covered are depicted in Figure 7.5.

7.4.1 Experiment Information

While most of the experimental settings used in this section are the same as those shown in Section 6.4 (as can be seen in Figure 7.5), the only change happens at the stage in which the balancing approach is selected. To determine whether random-undersampling is still able to help in this challenging situation, classifiers trained without any balancing method were included for the comparisons.

7.4.2 Experiment Details and Results

I Am Playr

As discussed in 6.4, when the churn labelling is applied to the game, the bias situation is not very serious (132 churners and 124 non-churners). Therefore, though random undersampling is applied, it may lead to little difference. As explained above, the number of data samples is relatively small compared to the dimensionality (2,394) of

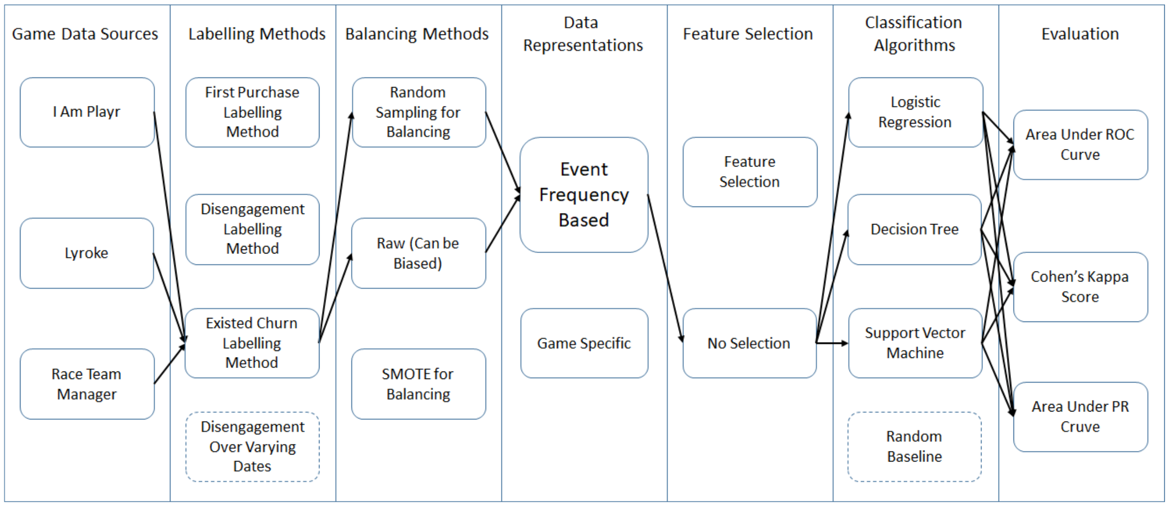

Figure 7.5: Balancing Investigation Experiment of Churn Prediction

Table 7.7: Performance for predicting churn behaviours with event-frequency-based data representation on the dataset of *I Am Playr* (with and without balancing)

		Event Frequency without Balancing	Event Feature Balanced by Undersampling	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (churning)	LogisticRegression	0.49±0.0144	0.50±0.0146	-3.2460e-01	7.4923e-01	-0.1452
	DecisionTree	0.66±0.0219	0.69±0.0198	-1.0184e+00	3.2200e-01	-0.4554
	SVM	0.71±0.0279	0.66±0.0339	9.9225e-01	3.3423e-01	0.4437
AUPRC (non churning)	LogisticRegression	0.54±0.0229	0.54±0.0203	1.3884e-01	8.9111e-01	0.0621
	DecisionTree	0.61±0.0344	0.72±0.0302	-2.2975e+00	3.3801e-02	-1.0275
	SVM	**0.67±0.0443**	0.47±0.0495	2.9349e+00	8.8506e-03	1.3125
AUROC	LogisticRegression	0.49±0.0261	0.50±0.0254	-4.4546e-01	6.6130e-01	-0.1992
	DecisionTree	0.59±0.0294	0.66±0.0324	-1.4967e+00	1.5181e-01	-0.6693
	SVM	0.51±0.0208	0.51±0.0283	-1.8168e-01	8.5786e-01	-0.0813
KAPPA	LogisticRegression	0.02±0.0534	0.09±0.0597	-7.6034e-01	4.5689e-01	-0.3400
	DecisionTree	0.09±0.0557	0.29±0.0558	-2.4738e+00	2.3553e-02	-1.1063
	SVM	0.00±0.0000	0.06±0.0297	-1.8477e+00	8.1152e-02	-0.8263

the event-frequency-based data representation model. For this reason, the performance of classifiers shown in Table 7.7 might be close to random classifiers in several cases.

Results of the experiment can be found in Table 7.7, where the notations shown are the same as in the experiments of the previous chapters. As can be seen, little significant difference is found between classifiers trained on the imbalanced and on the undersampled dataset, as expected. The only exceptional case (measurement: area under the PR curve with the non-churners as positive, classifier: SVM) is when the random-undersampling method brought significant negative impacts to the performance of the classifiers. This might because, some important information was removed when the random-undersampling method was applied. This experiment shows that there might be limited effects that a random-undersampling method can bring to the performance of classifiers when the number of data samples is relatively small. In addition, it may sometimes affect the quality of the classifiers if important information was removed by chance. To verify the observations from this experiment, similar experiments from the other two games are also shown.

Table 7.8: Performance for predicting churn behaviours with event-frequency-based data representation on the dataset of *Lyroke* (with and without balancing)

		Event Frequency without Balancing	Event Feature Balanced by Undersampling	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (churning)	LogisticRegression	0.75±0.0289	0.77±0.0361	-2.9401e-01	7.7211e-01	-0.1315
	DecisionTree	0.85±0.0305	0.82±0.0283	6.4880e-01	5.2466e-01	0.2902
	SVM	0.79±0.0291	0.78±0.0198	4.1100e-01	6.8592e-01	0.1838
AUPRC (non churning)	LogisticRegression	0.80±0.0301	0.79±0.0317	7.7560e-02	9.3903e-01	0.0347
	DecisionTree	0.83±0.0274	0.81±0.0248	3.1918e-01	7.5327e-01	0.1427
	SVM	0.63±0.0417	0.56±0.0358	1.3367e+00	1.9797e-01	0.5978
AUROC	LogisticRegression	0.78±0.0267	0.78±0.0310	-4.9334e-03	9.9612e-01	-0.0022
	DecisionTree	0.81±0.0344	0.80±0.0272	3.5259e-01	7.2849e-01	0.1577
	SVM	0.72±0.0310	0.71±0.0163	2.4523e-01	8.0906e-01	0.1097
KAPPA	LogisticRegression	0.46±0.0518	0.48±0.0631	-2.2090e-01	8.2766e-01	-0.0988
	DecisionTree	0.49±0.0699	0.42±0.0530	7.6619e-01	4.5350e-01	0.3426
	SVM	0.03±0.0481	**0.24±0.0270**	-3.5784e+00	2.1482e-03	-1.6003

Lyroke

As with *I Am Playr* , as discussed in Section 6.4, the bias is not serious (127 churners and 103 non-churner) in this case. For this reason, little difference might be found between the classifiers trained on the raw, imbalanced dataset and the undersampled dataset in Table 7.8.

Table 7.8 shows how the random-undersampling method has influenced the performance of the classifiers. As can be seen, the results shown in this table are similar to what was noticed in *I Am Playr* where no difference is found between the two situations in most cases. This might be because the bias in both games was not serious and random undersampling did not remove much information to reach the balancing point. The only exceptional case (measurement: Cohen's kappa, classifier: SVM) shows that random undersampling helped to improve the performance of the classifiers in this case only.

Race Team Manager

Unlike the datasets of other games, after processing by the churn-labelling method, this dataset shows a high bias. This fact makes the game more challenging for the random-undersampling method, as the original number of samples is only 260 (206 churners and 54 non-churners), which will be further reduced to 108 by the random-undersampling method. Therefore, even though this dataset is highly biased, improvements from random-undersampling are not expected to be significant.

Table 7.3 displays the results of the experiment. Similar to expected, though this dataset is highly biased and the number of samples has been even reduced by the random-undersampling method, no significant difference can be found in the comparison. This suggests that, if the original dataset is not quantitatively sufficient relative to the dimensionality of the models, random undersampling can hardly improve the situation.

7.4.3 Summary

This section reports on three experiments that were conducted to predict churn, which is a challenging predictive task because of the quantitative limitation of the labelled dataset. As

Table 7.9: Performance for predicting churn behaviours with event-frequency-based data representation on the dataset of *Race Team Manager* (with and without balancing)

		Event Frequency without Balancing	Event Feature Balanced by Undersampling	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (churning)	LogisticRegression	0.78±0.0209	0.78±0.0253	-2.0248e-02	9.8407e-01	-0.0091
	DecisionTree	0.81±0.0097	0.81±0.0175	-2.9703e-02	9.7663e-01	-0.0133
	SVM	0.79±0.0306	0.83±0.0285	-7.1713e-01	4.8250e-01	-0.3207
AUPRC (non churning)	LogisticRegression	0.27±0.0370	0.36±0.0490	-1.2846e+00	2.1520e-01	-0.5745
	DecisionTree	0.31±0.0491	0.27±0.0329	6.7847e-01	5.0610e-01	0.3034
	SVM	0.23±0.0292	0.30±0.0460	-1.0974e+00	2.8691e-01	-0.4908
AUROC	LogisticRegression	0.46±0.0408	0.52±0.0432	-9.7574e-01	3.4213e-01	-0.4364
	DecisionTree	0.49±0.0320	0.50±0.0376	-1.1562e-01	9.0924e-01	-0.0517
	SVM	0.46±0.0554	0.55±0.0577	-1.0665e+00	3.0029e-01	-0.4770
KAPPA	LogisticRegression	-0.06±0.0518	0.03±0.0417	-1.1912e+00	2.4906e-01	-0.5327
	DecisionTree	0.07±0.0583	0.03±0.0375	6.5183e-01	5.2275e-01	0.2915
	SVM	0.00±0.0000	-0.04±0.0451	8.2127e-01	4.2224e-01	0.3673

can be seen from the summarised Figure 7.6, the random-undersampling method behaves randomly in these experiments and was not able to make signification improvements in most cases. In addition, as can be found in the study of *I Am Playr* , this balancing method may sometimes have a negative impact on the performance of classifiers if some important information is removed by chance. This matches some cases that we noticed when predicting first purchases. Combining all investigations conducted for understanding the random-undersampling method, we might conclude that a random-undersampling method is more likely to be beneficial to the training of classifiers if the number of data samples is sufficient compared to the dimensionality of the data representations. In addition, it might sometimes have a negative impact if important information is removed. To help with the drawbacks of the random-undersampling method, an alternative, balanced approach, named *SMOTE*, may help to improve the performance of the classifiers without removing further important information.

7.5 SMOTE Balancing

SMOTE, short for synthetic-minority over-sampling technique, was introduced by Chawla et al. (2002). Instead of removing samples by random undersampling, this method generates synthetic data samples for the minor class by oversampling based on their k nearest neighbours. The original algorithm needs a parameter to control how many samples in minor class needs to be oversampled. To accurately balance the classification problem, in this work, this algorithm modified it to keep creating new synthetics based on samples in the minor class until the difference between two classes disappears. The modified SMOTE can be found in Algorithm 4.

This algorithm can balance class distributions for training more reliable classifiers based on limited resources. However, it also suffers from two aspects of problems. First, according to Wang et al. (2006), SMOTE cannot be used to generate synthetic data for balancing while the target classes are not linearly separable. The reason can be easily discovered in Figure 7.7. In addition, in some of the experiments to be shown in this section, if the data representation is highly dimensional (e.g., event frequency-based data representation) and the amount of synthetic data to be generated is large (while the data is highly biased to one side), SMOTE will require much time for generating these synthetic data samples. For

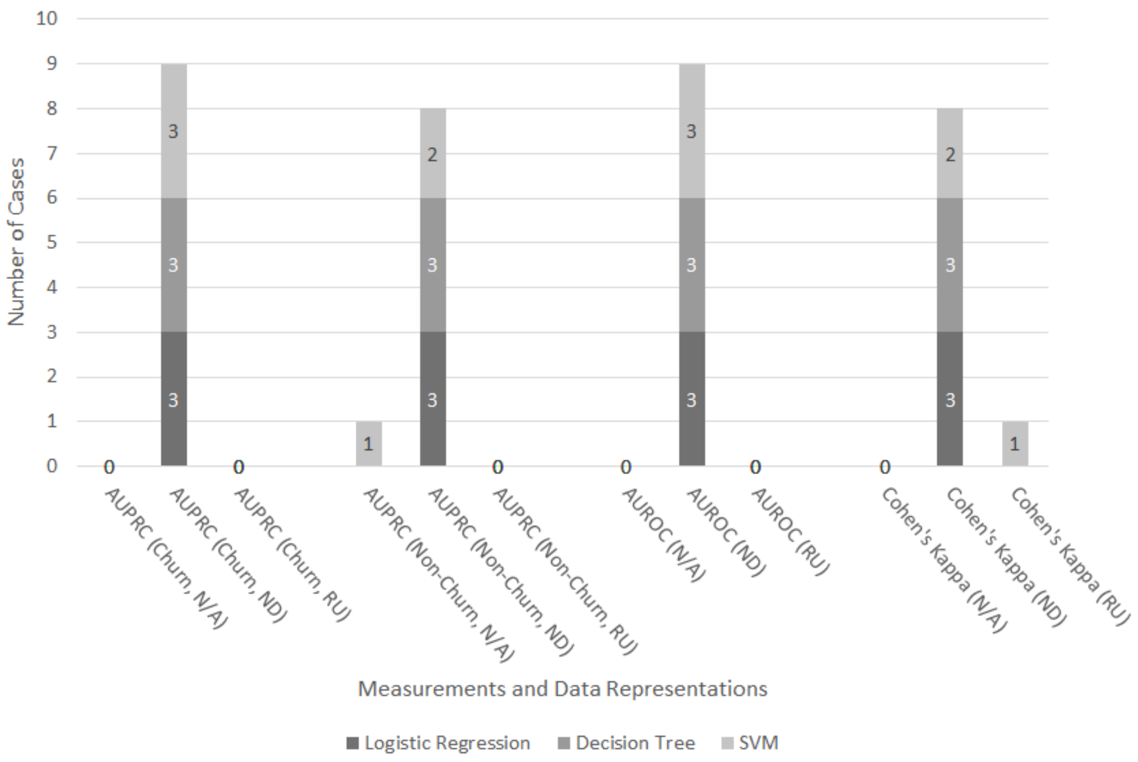

Figure 7.6: The number of cases where methods achieve significantly better performance and the number of cases where there is no difference found for predicting churn behaviours with undersampling

Algorithm 4 SMOTE For Balancing

```
1: procedure SMOTE For Balancing
2:     $diff$ = $numberOfSamplesInMajorClass$ - $numberOfSamplesInMinorClass$
3:     $generatedNumber = 0$
4:     $minorClassCopy$ = Copy($minorClass$)
5:     while $generatedNumber < diff$ do
6:         Pick random sample $i \in minorClassCopy$
7:         Find its $k$ nearest neighbours $k\_neighbours$
8:         Get a random neighbour $n$ from $k\_neighbours$
9:         Create an empty sample $x$
10:        for each feature $f$ used in data representation do
11:            random discount $c \in [0, 1]$
12:            $x[f] = i[f] + (n[f] - i[f]) * c$
13:        end for
14:        $minorClass = minorClass \cup x$
15:        $generatedNumber$ += 1
16:    end while
17: end procedure
```

Table 7.10: The specifications of the server for running the experiments

Specifications	**Details**
CPU	8 cores
Memory	56 GB
Storage	400 GB
Operating System	Windows Server 2012

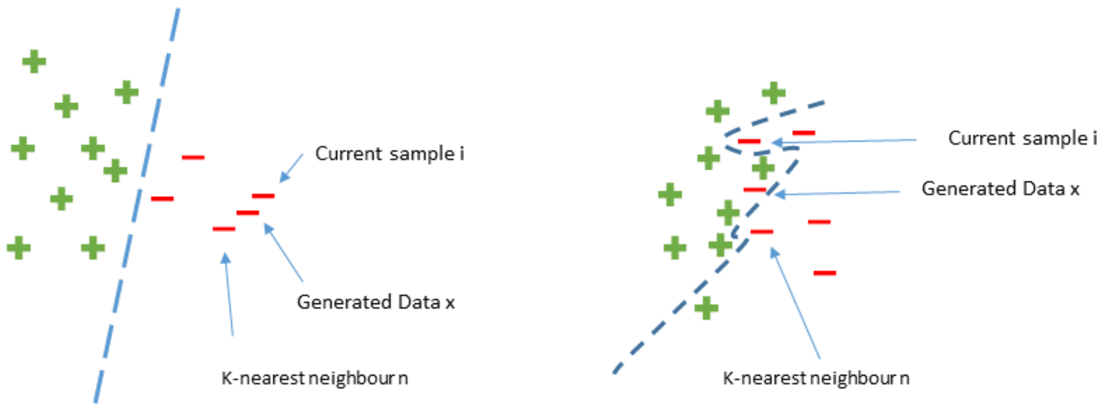

Figure 7.7: On the left, when the classes can be linearly separated, SMOTE is able to generate new synthetic data in the right place. However, when the classes can only be split by a non-linear classifier, SMOTE may create new synthetic data on the wrong side.

examples, in this work, while the SMOTE method was applied for balancing the predicting first purchase, because the difference between the number of paying users and non-paying users is large (489:885,681 in *I Am Playr* , 509:279,829 in *Lyroke* and 511:170,333 in *Race Team Manager*) and the dimensionality of the event frequency-based data representation is high (2,460 events in *I Am Playr* , 7,832 events in *Lyroke* and 1,531 events in *Race Team Manager*), although the experiment has kept running for seven days on the server described in Table 7.10, the results still cannot be gained. Because of this, while investigating the improvements that might be brought by SMOTE in sections 7.6 and 7.7, the experiments concerning first-purchase prediction are not included.

As Section 4.9 explains, this work runs every experiment with 10-fold cross-validation to avoid overfitting problems. SMOTE, as an oversampling method, can easily lead to overfitting problem (as it generates many synthetic data points similar to the original ones) if the classifiers trained are not tested in the right way. It would be natural to perform SMOTE upon the whole dataset first and then run a normal 10-fold cross-validation on the oversampled dataset. However, though 10-fold cross-validation was applied to avoid overfitting, the split training and testing set are no longer independent, as synthetic samples distributed in the testing set might be generated based on some samples that were left out of the training set. To solve this issue, the way to combine k-fold cross-validation and SMOTE is to generate synthetic samples only inside each experiment so that the test set has already been kept away and is therefore guaranteed to be blind for the classifiers. The modified k-fold cross-validation with SMOTE can be found in Algorithm 5.

Algorithm 5 K–Fold Cross-validation with SMOTE

```
1: procedure K–Fold Cross-validation with SMOTE
2:     Split Dataset into $k$ pieces
3:     $resultSet = \{ \}$
4:     for each piece $i \in k$ do
5:         $testingSet = i$, $trainingSet = k \setminus i$
6:         $OverSampledTrainingSet$ = SMOTEForBalancing($trainingSet$)
7:         Train classifier with $OverSampledTrainingSet$
8:         Test classifier with $testingSet$ and store in $currentResult$
9:         $resultSet = resultSet \cup currentResult$
10:    end for
11:    Calculate averaged result from elements in $resultSet$
12: end procedure
```

This section introduces another alternative balancing method SMOTE which is better for dealing with small datasets because it performs oversampling instead of undersampling on data samples. However, because no method is perfect for all types of problems, the detailed procedure and its potential drawbacks have also been covered. In the next section, experiments will be tested in all three games and compared with the random-undersampling method to evaluate the performance of SMOTE.

7.6 Disengagement Prediction – SMOTE

To determine whether the SMOTE method is able to replace the random undersampling approach, this section tests the disengagement prediction. As was observed in the experiments shown in Section 7.2, random-undersampling method is already able to make a positive impact in most cases. Therefore, the experiments in this section focus on investigating whether the SMOTE method is able to make a difference. In each experiment, the performance of classifiers trained with a SMOTE-balanced dataset are not only be compared with those trained with the raw dataset but are also be compared with those trained with the random-undersampling method. As has been investigated by Dittman et al. (2014), random undersampling are expected to achieve better performance across the three games. Experiments in this section follow the procedure in Figure 7.8.

I Am Playr

The disengagement in *I Am Playr* , as introduced in the experiment with the random-undersampling method, is highly biased to the negative (non-disengaging) side where the ratio is around 1:8.895 (1,354 and 12,044 players on either side, respectively).

Tables 7.11 and 7.12 display the comparison between the performances of classifiers trained under the balanced (by SMOTE) dataset and those trained under the raw (imbalanced) dataset and the other balanced (by random undersampling) dataset. As can be seen upon considering the comparison of SMOTE and the raw dataset, in five of 12 cases, the SMOTE balancing method did significantly improve the performance of the classifiers. However, according to Table 7.12, this is not as good as the classifiers trained under the random-undersampled dataset. As can be seen, there are three cases in which random undersampling did significantly better but only one case in which SMOTE significantly helped the classifiers. This suggests that SMOTE is able to help

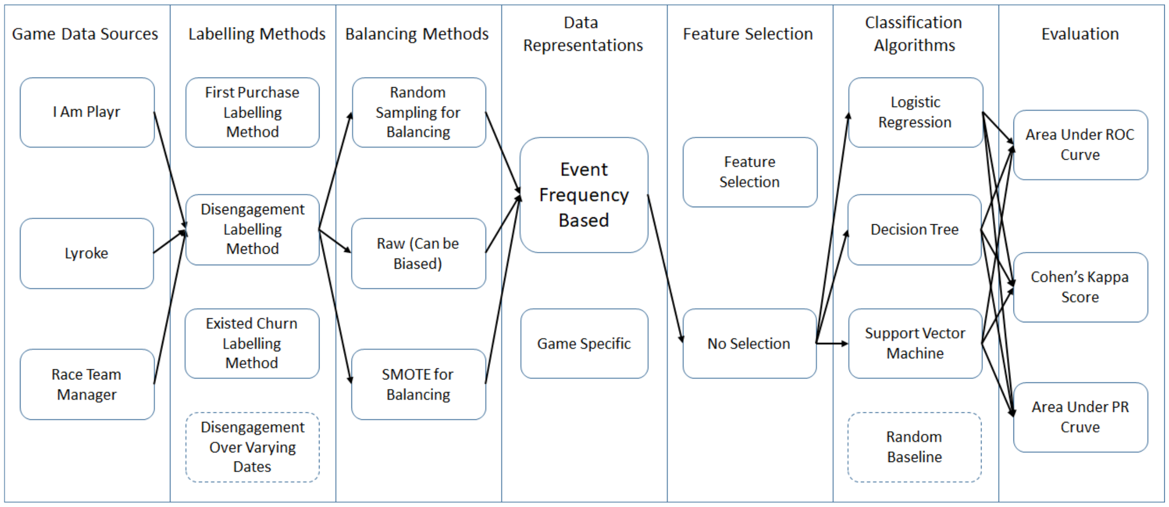

Figure 7.8: SMOTE Investigation Experiment of Disengagement Prediction

Table 7.11: Performance for prediction disengagement with SMOTE on the dataset of *I Am Playr*

		Event Feature without Balancing	Event Feature Balanced by SMOTE	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (disengaging)	LogisticRegression	0.19±0.0056	**0.25±0.0089**	-5.2301e+00	5.6573e-05	-2.3390
	DecisionTree	0.27±0.0138	0.28±0.0052	-4.5857e-01	6.5203e-01	-0.2051
	SVM	0.28±0.0522	0.21±0.0109	1.3983e+00	1.7902e-01	0.6253
AUPRC (non disengaging)	LogisticRegression	0.91±0.0020	**0.94±0.0033**	-5.8370e+00	1.5765e-05	-2.6104
	DecisionTree	0.96±0.0018	0.96±0.0008	2.4267e+00	2.5962e-02	1.0852
	SVM	0.97±0.0011	0.97±0.0012	-1.3240e-01	8.9614e-01	-0.0592
AUROC	LogisticRegression	0.59±0.0077	**0.71±0.0097**	-9.2225e+00	3.0573e-08	-4.1244
	DecisionTree	**0.72±0.0142**	0.66±0.0100	3.6181e+00	1.9664e-03	1.6181
	SVM	0.75±0.0097	0.78±0.0085	-2.0486e+00	5.5373e-02	-0.9162
KAPPA	LogisticRegression	0.11±0.0068	**0.25±0.0119**	-9.2718e+00	2.8217e-08	-4.1465
	DecisionTree	0.21±0.0213	0.19±0.0065	8.5339e-01	4.0466e-01	0.3816
	SVM	0.00±0.0000	**0.18±0.0084**	-2.0509e+01	6.2377e-14	-9.1720

Table 7.12: Performance for prediction disengagement with SMOTE or undersampling on the dataset of *I Am Playr*

		Event Feature Balanced by Undersampling	Event Feature Balanced by SMOTE	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (disengaging)	LogisticRegression	0.22±0.0082	0.25±0.0089	-2.3803e+00	2.8556e-02	-1.0645
	DecisionTree	**0.35±0.0097**	0.28±0.0052	5.7235e+00	1.9951e-05	2.5596
	SVM	0.22±0.0122	0.21±0.0109	7.1722e-01	4.8245e-01	0.3207
AUPRC (non disengaging)	LogisticRegression	0.94±0.0024	0.94±0.0033	1.2191e+00	2.3852e-01	0.5452
	DecisionTree	**0.98±0.0010**	0.96±0.0008	1.1996e+01	5.0753e-10	5.3647
	SVM	0.97±0.0011	0.97±0.0012	6.3140e-01	5.3572e-01	0.2824
AUROC	LogisticRegression	0.72±0.0084	0.71±0.0097	9.3849e-01	3.6042e-01	0.4197
	DecisionTree	**0.81±0.0066**	0.66±0.0100	1.1654e+01	8.0695e-10	5.2119
	SVM	0.78±0.0092	0.78±0.0085	6.4243e-01	5.2869e-01	0.2873
KAPPA	LogisticRegression	0.20±0.0081	**0.25±0.0119**	-2.9943e+00	7.7820e-03	-1.3391
	DecisionTree	0.23±0.0118	0.19±0.0065	2.4687e+00	2.3805e-02	1.1040
	SVM	0.17±0.0107	0.18±0.0084	-1.0109e+00	3.2546e-01	-0.4521

Table 7.13: Performance for prediction disengagement with SMOTE on the dataset of *Lyroke*

		Event Feature without Balancing	Event Feature Balanced by SMOTE	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (disengaging)	LogisticRegression	0.29±0.0042	0.27±0.0033	2.1944e+00	4.1571e-02	0.9814
	DecisionTree	0.26±0.0033	0.27±0.0073	-1.4955e+00	1.5212e-01	-0.6688
	SVM	0.24±0.0072	**0.28±0.0075**	-3.2814e+00	4.1477e-03	-1.4675
AUPRC (non disengaging)	LogisticRegression	0.95±0.0011	0.95±0.0014	5.4585e-01	5.9187e-01	0.2441
	DecisionTree	**0.96±0.0007**	0.95±0.0008	8.4455e+00	1.1231e-07	3.7770
	SVM	0.97±0.0006	**0.97±0.0006**	-3.8566e+00	1.1561e-03	-1.7247
AUROC	LogisticRegression	0.77±0.0033	0.76±0.0031	1.4538e+00	1.6321e-01	0.6502
	DecisionTree	**0.73±0.0042**	0.63±0.0086	9.6870e+00	1.4536e-08	4.3322
	SVM	0.77±0.0035	**0.80±0.0039**	-4.1658e+00	5.8075e-04	-1.8630
KAPPA	LogisticRegression	0.07±0.0042	**0.24±0.0062**	-2.0818e+01	4.8173e-14	-9.3100
	DecisionTree	0.14±0.0070	0.15±0.0066	-1.4963e+00	1.5190e-01	-0.6692
	SVM	0.00±0.0000	**0.23±0.0066**	-3.3105e+01	1.4030e-17	-14.8052

Table 7.14: Performance for prediction disengagement with SMOTE or undersampling on the dataset of *Lyroke*

		Event Feature Balanced by Undersampling	Event Feature Balanced by SMOTE	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (non disengaging)	LogisticRegression	0.26±0.0042	0.27±0.0033	-2.4769e+00	2.3402e-02	-1.1077
	DecisionTree	0.30±0.0091	0.27±0.0073	2.6687e+00	1.5658e-02	1.1935
	SVM	0.25±0.0050	0.28±0.0075	-2.6159e+00	1.7503e-02	-1.1699
AUPRC (non disengaging)	LogisticRegression	0.95±0.0019	0.95±0.0014	9.1493e-01	3.7232e-01	0.4092
	DecisionTree	**0.97±0.0009**	0.95±0.0008	1.5193e+01	1.0409e-11	6.7944
	SVM	0.97±0.0006	0.97±0.0006	-2.7009e+00	1.4625e-02	-1.2079
AUROC	LogisticRegression	0.77±0.0044	0.76±0.0031	3.2515e-01	7.4882e-01	0.1454
	DecisionTree	**0.78±0.0054**	0.63±0.0086	1.3813e+01	5.0776e-11	6.1774
	SVM	0.78±0.0035	0.80±0.0039	-2.5976e+00	1.8190e-02	-1.1617
KAPPA	LogisticRegression	0.24±0.0053	0.24±0.0062	-2.4524e-01	8.0905e-01	-0.1097
	DecisionTree	**0.22±0.0077**	0.15±0.0066	6.4008e+00	5.0124e-06	2.8625
	SVM	0.21±0.0027	0.23±0.0066	-2.7530e+00	1.3089e-02	-1.2312

in some cases but is not able to offer a performance that can match the random-undersampling method. This observation will be further verified with respect to the other two games.

Lyroke

Experiments show that random-undersampling is also able to significantly improve the performance of classifiers trained in the imbalanced dataset of *Lyroke* . To determine the performance of SMOTE in this game, similar experiments have also been done for comparisons.

Tables 7.13 and 7.14 gave an overview of how SMOTE affects the performance of classifiers compared to the raw dataset and the random-undersampling method. As can be seen, SMOTE was able to significantly improve the quality of classifiers in five of 12 cases. However, as in the experiment with *I Am Playr* , the improvements brought by SMOTE are not as good as those in the random-undersampling method. According to Table 7.14, the classifiers trained with the random-undersampling method are able to behave significantly better in three cases, whereas there is no case in which SMOTE offered better results. As similar observation can be made with respect to both games produced by WeR Interactive. It will be meaningful to further test the experiment in *Race Team Manager* , which was developed by a different company.

Table 7.15: Performance for prediction disengagement with SMOTE on the dataset of *Race Team Manager*

		Event Feature without Balancing	Event Feature Balanced by SMOTE	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (disengaging)	LogisticRegression	0.27±0.0024	0.27±0.0033	1.8360e-01	8.5638e-01	0.0821
	DecisionTree	0.28±0.0029	0.27±0.0034	4.8327e-01	6.3473e-01	0.2161
	SVM	0.30±0.0050	0.29±0.0047	1.9462e+00	6.7409e-02	0.8704
AUPRC (non disengaging)	LogisticRegression	0.92±0.0025	0.94±0.0041	-2.4298e+00	2.5795e-02	-1.0867
	DecisionTree	**0.96±0.0006**	0.94±0.0012	1.0643e+01	3.3925e-09	4.7599
	SVM	0.97±0.0006	0.97±0.0005	1.3550e+00	1.9217e-01	0.6060
AUROC	LogisticRegression	0.73±0.0049	0.75±0.0062	-2.2445e+00	3.7611e-02	-1.0038
	DecisionTree	**0.75±0.0031**	0.65±0.0099	9.6382e+00	1.5700e-08	4.3103
	SVM	0.81±0.0030	0.80±0.0029	1.6140e+00	1.2391e-01	0.7218
KAPPA	LogisticRegression	0.05±0.0041	**0.25±0.0038**	-3.3734e+01	1.0051e-17	-15.0864
	DecisionTree	0.13±0.0051	**0.17±0.0058**	-4.3006e+00	4.3053e-04	-1.9233
	SVM	0.01±0.0034	**0.24±0.0038**	-4.3979e+01	8.9722e-20	-19.6681

Table 7.16: Performance for prediction disengagement with SMOTE or undersampling on the dataset of *Race Team Manager*

		Event Feature Balanced by Undersampling	Event Feature Balanced by SMOTE	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (disengaging)	LogisticRegression	0.27±0.0026	0.27±0.0033	-2.3665e-01	8.1560e-01	-0.1058
	DecisionTree	**0.31±0.0050**	0.27±0.0034	5.3366e+00	4.5066e-05	2.3866
	SVM	0.30±0.0031	0.29±0.0047	1.6147e+00	1.2377e-01	0.7221
AUPRC (non disengaging)	LogisticRegression	0.93±0.0018	0.94±0.0041	-7.0653e-01	4.8891e-01	-0.3160
	DecisionTree	**0.97±0.0006**	0.94±0.0012	1.6420e+01	2.8133e-12	7.3430
	SVM	0.97±0.0005	0.97±0.0005	1.2564e+00	2.2504e-01	0.5619
AUROC	LogisticRegression	0.75±0.0027	0.75±0.0062	-2.3608e-01	8.1604e-01	-0.1056
	DecisionTree	**0.79±0.0028**	0.65±0.0099	1.2739e+01	1.9175e-10	5.6970
	SVM	0.80±0.0026	0.80±0.0029	1.5351e+00	1.4215e-01	0.6865
KAPPA	LogisticRegression	0.25±0.0044	0.25±0.0038	7.1131e-01	4.8601e-01	0.3181
	DecisionTree	**0.25±0.0049**	0.17±0.0058	1.0488e+01	4.2691e-09	4.6904
	SVM	0.25±0.0038	0.24±0.0038	7.1627e-01	4.8301e-01	0.3203

Race Team Manager

Because experiments show that SMOTE was able to help (though not as much as random undersampling) in both games developed by WeR Interactive, experiments were also conducted using *Race Team Manager* , which was developed by a different company.

Results of experiments shown in both tables 7.15 and table 7.16 indicate worse results for SMOTE compared to the other two games. In *Race Team Manager* , while comparing the classifiers trained under the raw dataset, SMOTE was only able to significantly improve their performances in three cases, whereas those trained under the raw dataset also achieve a significantly better performance in two cases. This suggests that SMOTE did not help much in this case. Compared to random undersampling, the results turn out to be clearer, as the classifier trained with the random-undersampling method achieved significantly better performance in four cases whilst those trained with SMOTE did better in no cases.

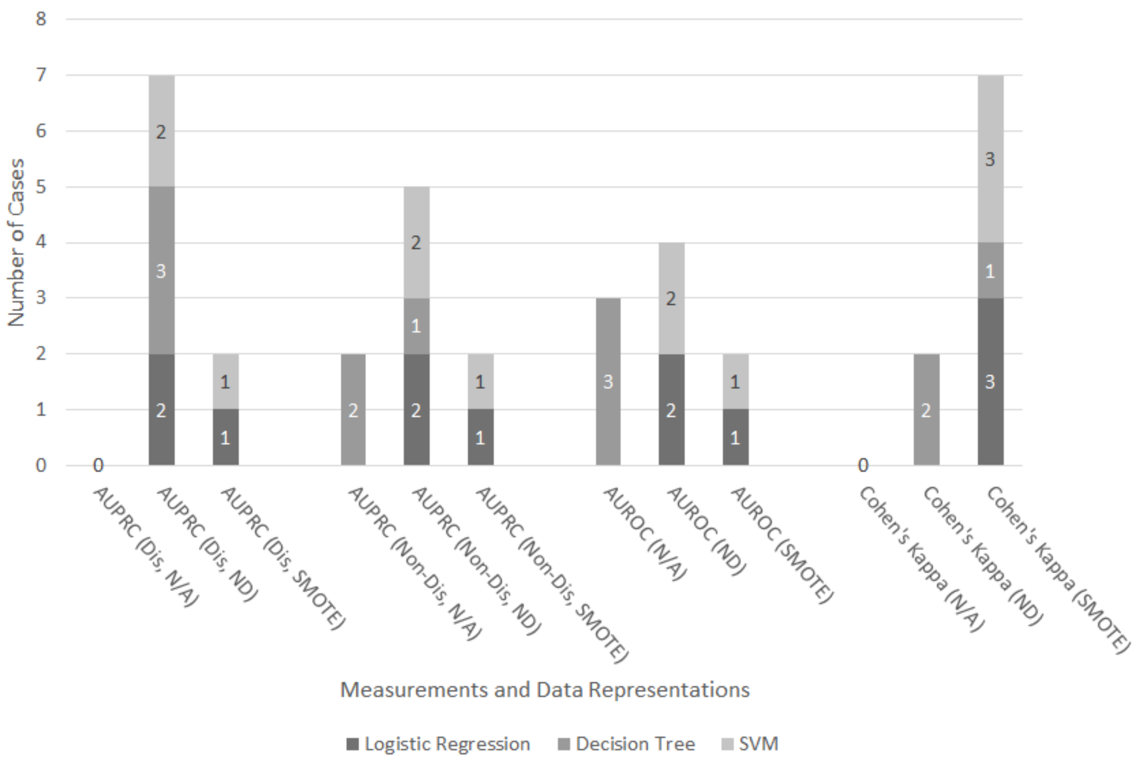

Figure 7.9: The number of cases where methods achieve significantly better performance and the number of cases where there is no difference found for predicting disengagement behaviours with SMOTE

7.6.1 Summary

In summary, according to the three experiments described in this section, SMOTE can be used for balancing purposes. As can be seen in Figure 7.9, SMOTE was able to help to improve the performance of the classifiers trained in some cases (especially for Cohen's Kappa). However, as can be found in Figure 7.10, similar to what was expected, while predicting disengagement in these three games, the random-undersampling method achieved better improvements in most cases. This suggests that the random-undersampling method is more helpful in cases where the dataset is quantitatively sufficient. This matches the conclusion that suggested by Dittman et al. (2014). In the next section, to determine whether random-undersampling can outperform SMOTE in cases in which the number of data samples is limited, results of experiments conducted for predicting the churning behaviours are discussed.

7.7 Churn Prediction – SMOTE

Experiments in the last section show that SMOTE is able to help predict players' disengagement behaviours in several cases. However, when the number of samples is sufficient, random-undersampling performs better than SMOTE for improving the performance of classifiers. In this section, SMOTE is tested with respect to predicting churn when the number of samples is insufficient for highly dimensional data representations such as the event-

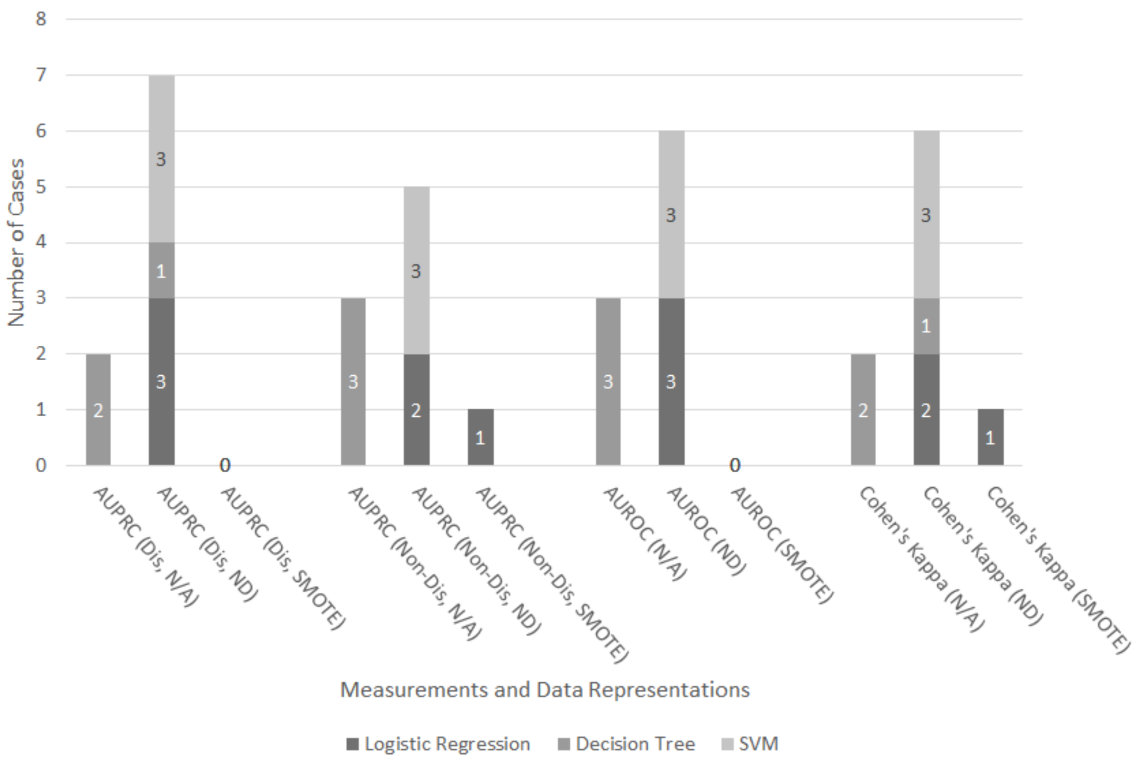

Figure 7.10: The number of cases where methods achieve significantly better performance and the number of cases where there is no difference found for predicting disengagement behaviours with SMOTE and undersampling

frequency-based method. As discussed in Section 7.4, given that the distribution of classes is close to balance in both *I Am Playr* and *Lyroke* , the difference might not be outstanding between classifiers trained with SMOTE and the random-undersampling method. However, due to the high bias shown in *Race Team Manager* , it is worth investigating if an oversampling method can be more helpful than the undersampling method in this case. Experiments in this section follow the procedure in Figure 7.11.

I Am Playr

As discussed above, the bias in *I Am Playr* is relatively small (132:124 between churning and non-churning players). Therefore, little difference was expected among the comparisons shown in tables 7.17 and 7.18. However, as the table displays, when comparing the classifiers trained with SMOTE and those trained with raw data, except for one case in which SVM behave significantly better, there is no significant different found between classifiers trained under both situations in most cases. Besides, there is even a case where the SVM classifier trained with the raw data behaves better. This indicates that SMOTE is not able to bring significant help to the training of classifiers when the class is relatively balanced. According to Table 7.18, SMOTE behaves like random undersampling in this case except where random-undersampling did better. Combining this experiment and that introduced in Section 7.4, a balancing method might not be needed for a classification problem which has a distribution that is close

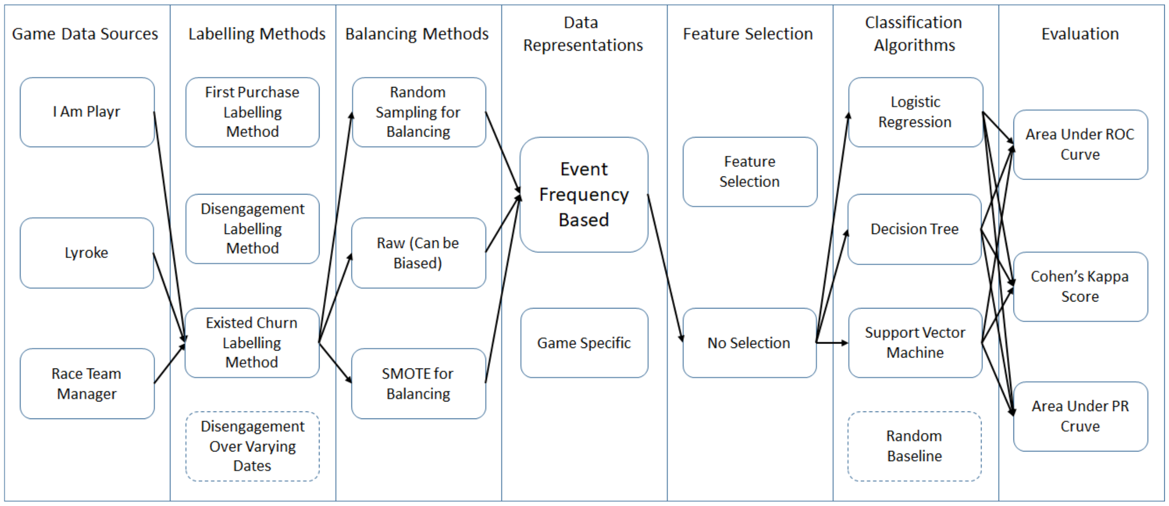

Figure 7.11: SMOTE Investigation Experiment of Churn Prediction

Table 7.17: Performance for prediction churn with SMOTE on the dataset of *I Am Playr*

		Event Feature without Balancing	Event Feature Balanced by SMOTE	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (churning)	LogisticRegression	0.49±0.0144	0.49±0.0142	-1.3194e-01	8.9650e-01	-0.0590
	DecisionTree	0.66±0.0219	0.66±0.0239	-2.2462e-01	8.2480e-01	-0.1005
	SVM	0.71±0.0279	0.64±0.0430	1.2936e+00	2.1214e-01	0.5785
AUPRC (non churning)	LogisticRegression	0.54±0.0229	0.54±0.0202	-4.5157e-02	9.6448e-01	-0.0202
	DecisionTree	0.61±0.0344	0.59±0.0265	2.9936e-01	7.6810e-01	0.1339
	SVM	**0.67±0.0443**	0.45±0.0572	2.9113e+00	9.3150e-03	1.3020
AUROC	LogisticRegression	0.49±0.0261	0.49±0.0244	-8.0133e-02	9.3702e-01	-0.0358
	DecisionTree	0.59±0.0294	0.58±0.0296	3.4477e-01	7.3426e-01	0.1542
	SVM	0.51±0.0208	0.55±0.0369	-9.0066e-01	3.7966e-01	-0.4028
KAPPA	LogisticRegression	0.02±0.0534	0.04±0.0591	-2.8251e-01	7.8078e-01	-0.1263
	DecisionTree	0.09±0.0557	0.13±0.0521	-5.0808e-01	6.1757e-01	-0.2272
	SVM	0.00±0.0000	**0.10±0.0326**	-3.0074e+00	7.5620e-03	-1.3450

Table 7.18: Performance for prediction churn with SMOTE or undersampling on the dataset of *I Am Playr*

		Event Feature Balanced by Undersampling	Event Feature Balanced by SMOTE	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (churning)	LogisticRegression	0.50±0.0146	0.49±0.0142	1.9610e-01	8.4673e-01	0.0877
	DecisionTree	0.69±0.0198	0.66±0.0239	7.3316e-01	4.7290e-01	0.3279
	SVM	0.66±0.0339	0.64±0.0430	4.1570e-01	6.8254e-01	0.1859
AUPRC (non churning)	LogisticRegression	0.54±0.0203	0.54±0.0202	-1.9626e-01	8.4660e-01	-0.0878
	DecisionTree	**0.72±0.0302**	0.59±0.0265	2.9393e+00	8.7672e-03	1.3145
	SVM	0.47±0.0495	0.45±0.0572	2.0629e-01	8.3888e-01	0.0923
AUROC	LogisticRegression	0.50±0.0254	0.49±0.0244	3.7938e-01	7.0884e-01	0.1697
	DecisionTree	0.66±0.0324	0.58±0.0296	1.8213e+00	8.5217e-02	0.8145
	SVM	0.51±0.0283	0.55±0.0369	-6.8266e-01	5.0351e-01	-0.3053
KAPPA	LogisticRegression	0.09±0.0597	0.04±0.0591	4.5671e-01	6.5334e-01	0.2042
	DecisionTree	0.29±0.0558	0.13±0.0521	2.0464e+00	5.5603e-02	0.9152
	SVM	0.06±0.0297	0.10±0.0326	-9.7859e-01	3.4075e-01	-0.4376

Table 7.19: Performance for prediction churn with SMOTE on the dataset of *Lyroke*

		Event Feature without Balancing	Event Feature Balanced by SMOTE	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (churning)	LogisticRegression	0.75±0.0289	0.75±0.0283	5.5481e-02	9.5637e-01	0.0248
	DecisionTree	0.85±0.0305	0.84±0.0291	1.2715e-01	9.0023e-01	0.0569
	SVM	0.79±0.0291	0.79±0.0199	8.9577e-02	9.2961e-01	0.0401
AUPRC (non churning)	LogisticRegression	0.80±0.0301	0.80±0.0321	-4.2521e-02	9.6655e-01	-0.0190
	DecisionTree	0.83±0.0274	0.79±0.0336	7.7132e-01	4.5053e-01	0.3449
	SVM	0.63±0.0417	0.60±0.0239	5.5410e-01	5.8633e-01	0.2478
AUROC	LogisticRegression	0.78±0.0267	0.78±0.0268	6.6880e-02	9.4741e-01	0.0299
	DecisionTree	0.81±0.0344	0.79±0.0378	4.8699e-01	6.3214e-01	0.2178
	SVM	0.72±0.0310	0.72±0.0181	-3.5678e-02	9.7193e-01	-0.0160
KAPPA	LogisticRegression	0.46±0.0518	0.47±0.0498	-1.3587e-01	8.9343e-01	-0.0608
	DecisionTree	0.49±0.0699	0.37±0.0731	1.0682e+00	2.9953e-01	0.4777
	SVM	0.03±0.0481	0.26±0.0565	-2.8775e+00	1.0021e-02	-1.2868

Table 7.20: Performance for prediction churn with SMOTE or undersampling on the dataset of *Lyroke*

		Event Feature Balanced by Undersampling	Event Feature Balanced by SMOTE	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (churning)	LogisticRegression	0.77±0.0361	0.75±0.0283	3.4565e-01	7.3362e-01	0.1546
	DecisionTree	0.82±0.0283	0.84±0.0291	-5.3299e-01	6.0056e-01	-0.2384
	SVM	0.78±0.0198	0.79±0.0199	-4.0285e-01	6.9180e-01	-0.1802
AUPRC (non churning)	LogisticRegression	0.79±0.0317	0.80±0.0321	-1.1661e-01	9.0846e-01	-0.0521
	DecisionTree	0.81±0.0248	0.79±0.0336	5.1784e-01	6.1087e-01	0.2316
	SVM	0.56±0.0358	0.60±0.0239	-1.0888e+00	2.9059e-01	-0.4869
AUROC	LogisticRegression	0.78±0.0310	0.78±0.0268	6.6629e-02	9.4761e-01	0.0298
	DecisionTree	0.80±0.0272	0.79±0.0378	2.0239e-01	8.4188e-01	0.0905
	SVM	0.71±0.0163	0.72±0.0181	-4.0488e-01	6.9034e-01	-0.1811
KAPPA	LogisticRegression	0.48±0.0631	0.47±0.0498	1.0277e-01	9.1928e-01	0.0460
	DecisionTree	0.42±0.0530	0.37±0.0731	4.5240e-01	6.5639e-01	0.2023
	SVM	0.24±0.0270	0.26±0.0565	-2.5797e-01	7.9936e-01	-0.1154

to balanced. Instead of improvement, they might even bring negative effects in some cases.

Lyroke

As with *I Am Playr* , because the bias is not serious (127 churners and 103 non-churner) in *Lyroke* , little difference was expected. Tables 7.19 and Table 7.20 further verify this expectation, as no significant difference is found between any of the comparisons of the three datasets (the raw dataset, the SMOTE-balanced dataset and the random-undersampling-balanced dataset). As with *I Am Playr* , combining the results from Section 7.4 suggests that, whenever The bias is not serious, it might not be ideal to include a balancing method.

Race Team Manager

Unlike the two previous experiments, the dataset labelled by the churn definition in *Race Team Manager* shows high bias towards the churning side. In this case, a balancing method is needed. However, because the number of samples in this dataset is already small while being used for training a highly dimensional model, it is risky to use the random undersampling method. SMOTE in this is more likely to be helpful in this case as it utilises the limited information to create synthetic training examples. But, due to the small number of samples, any balancing method may hardly make a

Table 7.21: Performance for prediction churn with SMOTE on the dataset of *Race Team Manager*

		Event Feature without Balancing	Event Feature Balanced by SMOTE	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (churning)	LogisticRegression	0.78±0.0209	0.79±0.0230	-2.1467e-01	8.3244e-01	-0.0960
	DecisionTree	0.81±0.0097	**0.89±0.0153**	-3.9578e+00	9.2270e-04	-1.7700
	SVM	0.79±0.0306	0.83±0.0247	-8.4412e-01	4.0968e-01	-0.3775
AUPRC (non churning)	LogisticRegression	0.27±0.0370	0.28±0.0377	-3.4201e-02	9.7309e-01	-0.0153
	DecisionTree	0.31±0.0491	0.38±0.0398	-1.0030e+00	3.2914e-01	-0.4486
	SVM	0.23±0.0292	0.25±0.0242	-2.8195e-01	7.8119e-01	-0.1261
AUROC	LogisticRegression	0.46±0.0408	0.47±0.0486	-9.7642e-02	9.2330e-01	-0.0437
	DecisionTree	0.49±0.0320	0.61±0.0414	-2.1929e+00	4.1691e-02	-0.9807
	SVM	0.46±0.0554	0.55±0.0387	-1.2300e+00	2.3453e-01	-0.5501
KAPPA	LogisticRegression	-0.06±0.0518	-0.02±0.0548	-4.3425e-01	6.6926e-01	-0.1942
	DecisionTree	0.07±0.0583	0.15±0.0641	-8.4417e-01	4.0965e-01	-0.3775
	SVM	0.00±0.0000	-0.06±0.0496	1.2267e+00	2.3574e-01	0.5486

Table 7.22: Performance for prediction churn with SMOTE or undersampling on the dataset of *Race Team Manager*

		Event Feature Balanced by Undersampling	Event Feature Balanced by SMOTE	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (churning)	LogisticRegression	0.78±0.0253	0.79±0.0230	-1.7591e-01	8.6233e-01	-0.0787
	DecisionTree	0.81±0.0175	**0.89±0.0153**	-3.0554e+00	6.8114e-03	-1.3664
	SVM	0.83±0.0285	0.83±0.0247	-8.4299e-02	9.3375e-01	-0.0377
AUPRC (non churning)	LogisticRegression	0.36±0.0490	0.28±0.0377	1.2465e+00	2.2854e-01	0.5575
	DecisionTree	0.27±0.0329	0.38±0.0398	-2.0026e+00	6.0515e-02	-0.8956
	SVM	0.30±0.0460	0.25±0.0242	9.4540e-01	3.5697e-01	0.4228
AUROC	LogisticRegression	0.52±0.0432	0.47±0.0486	7.9619e-01	4.3630e-01	0.3561
	DecisionTree	0.50±0.0376	0.61±0.0414	-1.9496e+00	6.6972e-02	-0.8719
	SVM	0.55±0.0577	0.55±0.0387	3.1666e-02	9.7509e-01	0.0142
KAPPA	LogisticRegression	0.03±0.0417	-0.02±0.0548	6.7551e-01	5.0794e-01	0.3021
	DecisionTree	0.03±0.0375	0.15±0.0641	-1.5934e+00	1.2847e-01	-0.7126
	SVM	-0.04±0.0451	-0.06±0.0496	3.5504e-01	7.2669e-01	0.1588

significant difference. As can be seen in Table 7.21, compared to the classifiers trained on the balanced dataset, SMOTE was able to provide significantly better performance in one case where the decision tree is the classifier and the area under PRC was applied as the measurement. And in the same case, in Table 7.22, SMOTE also achieved better performance than the random-undersampling method. Except in this case, no differences among the three datasets were found. This shows that, in a biased dataset in which the number of data samples is insufficient for the training, both the undersampling method and the SMOTE are unable to help much, although SMOTE shows some potentials that might work better than the random-undersampling method.

7.7.1 Summary

SMOTE has been applied to all three games for predicting players' churn behaviours in this section. As has been introduced in the experiments in **I Am Playr** and **Lyroke**, as the bias was not serious, no significant difference is expected to be brought by SMOTE. In addition, although the bias condition is worse in **Race Team Manager**, because the quantity of data samples is small, no significant difference is expected as well. Figure 7.12 verifies the expectations that SMOTE was not able to bring significant improvements in most cases. By

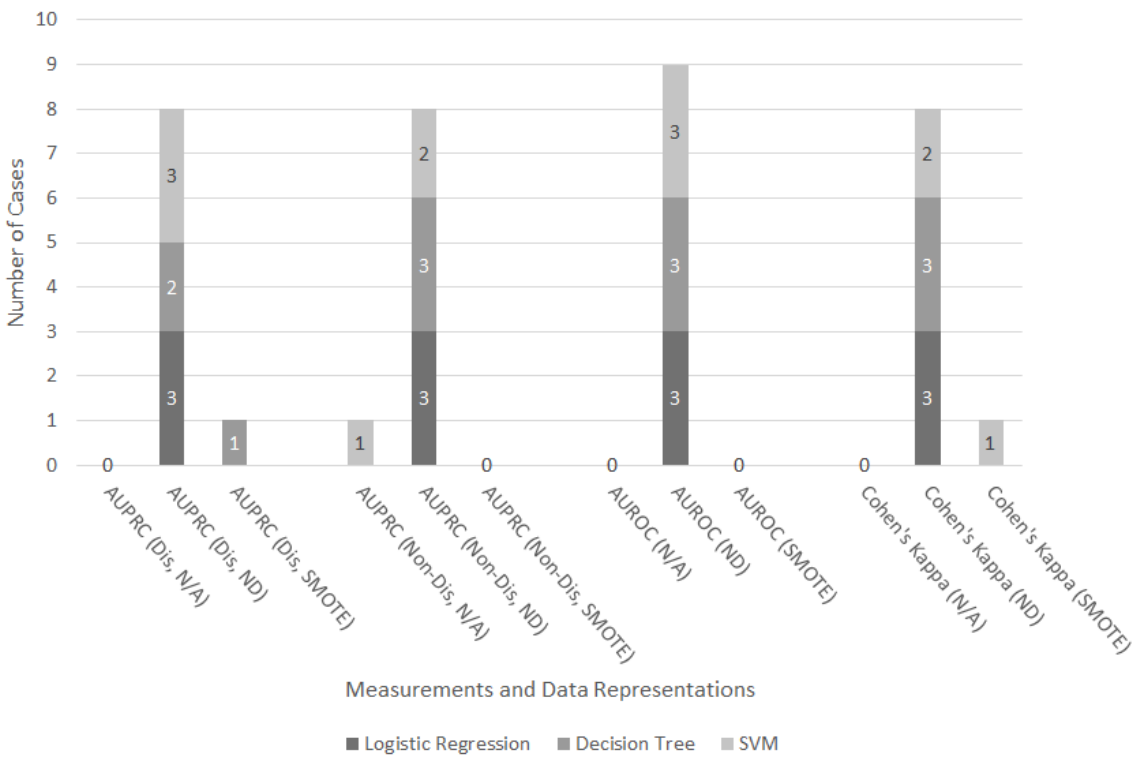

Figure 7.12: The number of cases where methods achieve significantly better performance and the number of cases where there is no difference found for predicting churn behaviours with SMOTE

comparing both the random undersampling method and SMOTE in Figure 7.13, although no significant differences can be found in most cases, it is notable that there was one case that random undersampling method achieved significantly better performance and one case SMOTE did better. The case when SMOTE did significantly better happens in *Race Team Manager* (can also be found in Table 7.21). This case is special as the size of this data sample is small. It indicates that SMOTE might have the potential to behave better than random-undersampling given a smaller dataset. This is reasonable because SMOTE combines the information of the minor class for generating neighbours. However, this improvement may not be significant when the number of minor examples is too small and important information is not included.

Combining results until now in this chapter, as can be found in Figure a balancing method such as random undersampling or SMOTE would help in most cases when the number of data samples is sufficient for a predictive task. However, though the labelling method has led to a dataset in which the quantity of data examples is relatively small for training highly dimensional classifiers, neither of the methods could make significant improvements to the performance of the classifiers. When the bias of the population is serious or the dataset is too small to contain important information, traditional approaches can hardly make an effective change to the variety of the original dataset. Taking SMOTE as an example, although synthetic examples can be generated for filling the gap between the classes, because the generated examples are combinations of the existing samples, a breakthrough can hardly be made to add in fresh populations to the dataset. To find a solution to this problem, this

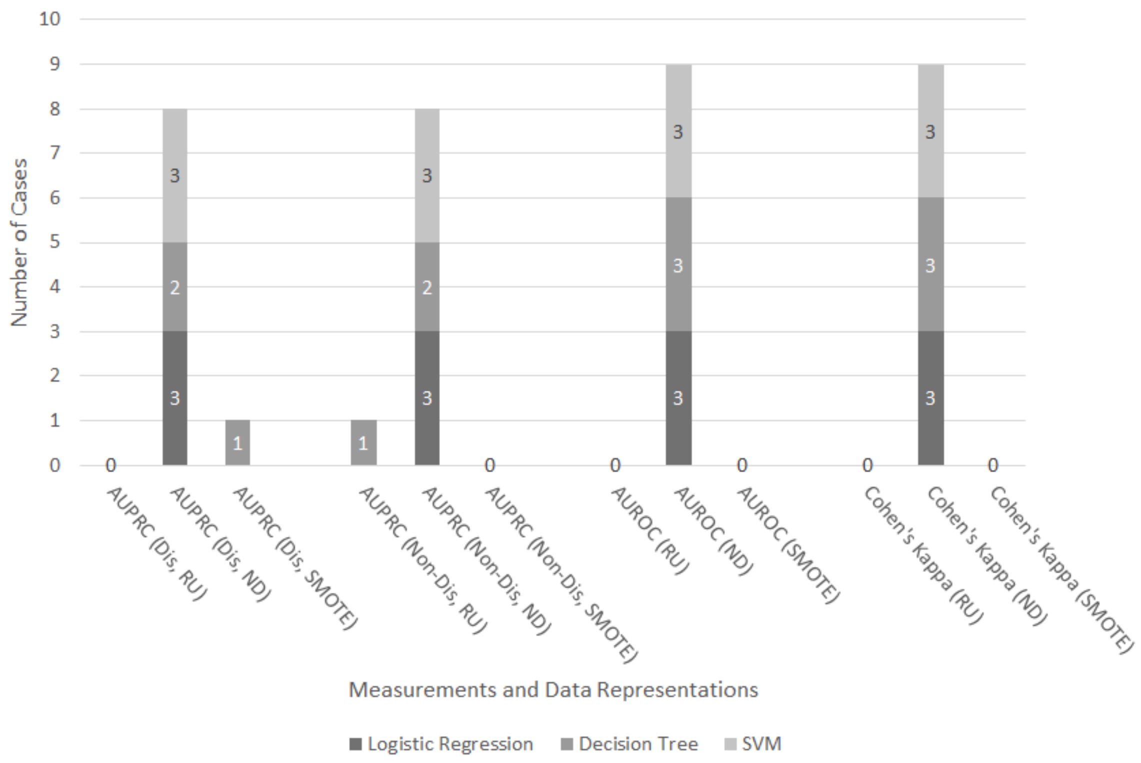

Figure 7.13: The number of cases where methods achieve significantly better performance and the number of cases where there is no difference found for predicting churn behaviours with SMOTE and undersampling

work proposes an alternative disengagement-labelling method that may avoid the appearance of bias. While aimed at predicting player's disengagement behaviours, this new approach is named *disengagement over varying dates*. Preliminary research have been done to investigate if this method can be used for balancing the classification, details of this method can be found in Appendix A.

7.8 Conclusion

Prior works have provided several approaches to predicting player-behaviour trends including disengagement and purchasing: Hadiji et al. (2014); Runge et al. (2014); Xie et al. (2014, 2015). Many of these are already able to provide promising performances. However, most of the labelling methods applied in these works were overly restrictive: The instances in the datasets were split into classes by satisfying some specific conditions. For this reason, the resultant class distributions of these definitions are often imbalanced. This type of issue can easily lead to biased classifiers during the training process. In addition, when the labelling method results in too few data samples for training, highly dimensional data representation can hardly work.

Two main conclusions can be addressed with the results of this chapter. First, based on the fact that event-frequency-based data representation was used to predict players' behaviours such as disengagement, first purchase and churn without being balanced, it is suggested that a biased dataset without balancing can easily lead to not well-trained classifiers.

To deal with this problem, random undersampling and SMOTE were applied to improve the performance of classifiers; however, when the important information is missing from the minor class or the total number of data samples is too small, neither of these two methods help.

To deal with this case, a new labelling method called *disengagement over varying dates* is also proposed. It can be applied to ensure the distribution of data samples to be more closed to balanced without removing any information. The details of this method can be found in Appendix A.

Chapter 8

Conclusion

Previous chapters have covered all experiments that were conducted in this research. During these experiments, several contributions have been made for addressing various purposes. In this chapter, a review is given for these contributions based on results from the experiments. Next, the limitations of some of the contributions are covered and suggestions are given that may help in these cases. Finally, some potential future research is discussed which may help to improve the contributions of this work.

8.1 Contributions

8.1.1 Main Research Hypothesis

In this work, two main research hypotheses have been explored. The first is that, ***event-based game-data representation can be used to predict player behaviour with supervised learning and can provide a better performance compared to random-guess and other state-of-the-art methods in a wide range of games***. This study aims to investigate the ability of the proposed *event-frequency-based data representation.*

The other main hypothesis is that, ***the disengagement-over-varying-dates labelling method can be used to make use of all data samples while maintaining an approximately balanced dataset, and that parameters optimised for balancing can be used as indicators of a game's health***. This study investigates the new labelling method (named disengagement over varying dates), which is designed to give an alternative way to deal with the possible bias and lack of data problems in some datasets caused by the widely used churn-labelling method.

8.1.2 Contribution Summary

Several contributions have been made over the course of this study. In this section, all the contributions made by this work are summarised and reviewed. Discussions are given on how well the hypotheses were supported based on the results of the experiments.

Event Frequency-based data Representation

In this work, a review has been given of several studies that have attempted to predict player behaviours using data-mining approaches in games. One of the common limitations of these works lies in their inability to be generic: The proposed methods cannot readily be migrated to different products. Aimed at working out a possible solution to the limitation from the selection of data representation, this work first analysed the reasons for the low generality of existing works (in Section 4.5.1). It was claimed that the generality of a data representation often comes from two aspects: *game-specific* and *availability*. Some existing works have worked to solve the *game-specific* issues by relying only on some session-based features; however, the session related information is not always collected in games, and it often may not be accurate enough. To cope with these two problems at the same time, as the first main contribution, this work takes the counts of any collected events (player behaviours or system events) as the data representation. In this way, because only counts of events are used instead of their actual meanings, there is no need to understand the true meaning of the events before the method can be applied. For the same reason, this method is not restricted to any specific game, and the same implementation of the data representation can be easily extended to work for different games. Besides, since this approach takes any events that are collected in games, it is able to maximise the use of the data and reduce the chances of encountering availability issues. To investigate its performance, the first hypothesis in Section 8.1.1 was proposed.

Player First-purchase behaviour Prediction

To verity this hypothesis, this work uses event-frequency-based data representation to predict players' first-purchase behaviours in three different commercial games. In the relevant experiments, event-frequency-based data representation exhibited its generality, as it shows robust performance across all three different genres of games without any special pre-processing and is able to provide promising predictive performance which is significantly better than random guesses. This gives positive support to the hypothesis made for this data-representation method.

Disengagement-labelling method

The churn-labelling method has been widely used to represent disengaging behaviours in games. It focuses on predicting disengaging actions in which players entirely stop playing a game. However, Runge et al. (2014) discuss the fact that players who have been predicted to churn can hardly be held back by simple in-game rewards. To give developers an earlier chance to deal with possible churning users, a new labelling method called 'disengagement' was introduced that focuses instead on players' disengaging trends.

Player Disengagement/Churn Behaviour Prediction

To further validate the hypothesis concerning event-frequency-based data representation, it has been further used to predict players' disengaging behaviours with both the newly proposed disengagement-labelling method and the churn-labelling method used in the work by Runge et al. (2014). During the experiments, the game-specific data representation introduced in the work by Runge et al. (2014) was also added for comparison. Experiments predicting the disengagement (labelled) behaviours show that event-frequency-based data representation exhibits good generality and can achieve competitive performance. Combined with the experiments for predicting the first-purchase behaviours, both experiments offer positive support to the first hypothesis made in this study. However, during the experiment in which event-frequency-based data representation was applied to predict players' churn behaviours, we saw that classifiers trained with the event frequency-based data representation can in most cases still achieve competitive performance compared with those trained with game-specific data representations. This hypothesis was not well supported, as there are cases in which the classifiers trained with the event frequency-based data representation behaved like random classifiers. As discussed in Section 6.4, this might be caused by the limited number of examples available for training, as classifiers (especially higher-dimensional ones) can hardly be well trained under this situation.

A Review of Two Popular Class-balancing Methods

A general problem in data-mining research is that the class distribution can be biased if the labelling method distributes training examples into different classes by some fixed conditions. In this work, the general causes of a biased classification problem were discussed and two widely used methods–*random undersampling* and *SMOTE (Synthetic Minority Over-sampling Technique)*–were introduced. In Chapter 7, statistical comparisons were given to show whether random undersampling and SMOTE would help reduce class bias such that the performance of the classifiers can be improved. Based on experiments, summaries were given for these two methods, and it was suggested that random undersampling is more likely to achieve better performance when the number of data samples is large enough, whereas SMOTE is better able to be used with smaller datasets. If the data sample is too small, none of the methods help significantly.

Disengagement-over-varying-dates labelling method

Two important problems faced by some experiments in this study are *lack of data samples* and *biased class distribution.* As discussed in sections 6.5 and 7.1, both of the issues can be found when the churn-labelling method is applied, as the predictive methods and classifiers trained in this situation cannot behave well. To provide more reliable classifiers for predicting player behaviours, the second main contribution of this work is to propose an alternative labelling method that also reflects players' disengaging behaviours but is able to use all data samples while maintaining an approximately balanced dataset. In addition, parameters optimised for balancing in this method can be used as indicators of a game's health. To investigate its effects, the second hypothesis in Section 8.1.1 was proposed. Experiments in Section A show that this labelling method can help balance the class distribution and provide a larger dataset in all three commercial games for predicting players' disengagement behaviours. Classifiers trained under this situation were able to behave significantly better than a random classifier. This result supports the hypothesis proposed and suggests that this labelling method can be an alternative when reliable classifiers cannot be reached for predicting the original churn behaviours.

8.2 Limitations

The previous section offers a summary of all the contributions that have been made in this work. As the main contribution, event-frequency-based data representation can be migrated to different games and is able to offer competitive performance for the prediction of various player behaviours when the data sample is sufficient. However, when a game contains a large number of game events, event-frequency-based data representation may result in a higher-dimensional feature space. This is known as the 'curse of dimensionality' in data-mining. It is often faced while facing large-scaled data-mining problems (Marimont and Shapiro, 1979). In terms of the computing resources, the system memory might often be filled with the highly dimensional feature space. A solution offered in this work is to use sparse matrices rather than dense ones for storage. This is because players' behaviours are rather sparse in the highly dimensional space, as some behaviours may only be exhibited by specific players. Regarding interpretations, while facing a highly dimensional data space, although an interpretable algorithm can be applied for the prediction of future behaviours, the resultant model may still be barely readable. As discussed in Section 4.6, a feature-selection process before training is able to help decrease the dimension of the resultant models and thereby render them more interpretable.

8.3 Future Work

Incremental Event-frequency-based Data Representation

As has been explained, the most notable limitation of event-frequency-based data representation is its high dimensionality when the number of events in games is large. Several approaches (including disengagement over varying dates introduced in Appendix A) considered in this work have already been attempted to ensure that the highly dimensional models are well trained. Another limitation might be the model through periods, because, while operating a game, developers often release new contents with patches or DLCs. In this case, some new events will be added to the system and others will be removed. To make sure the model can handle unseen game content in the future, a potential improvement might be to use a hashing-based event-frequency feature space to replace the current method that includes all living events in a game. This method may offer two benefits and a potential drawback. At first, it will help the model trained to be used for predicting the player behaviours in the game when minor changes happen to the games. For unseen events, a hashing mechanism can always ensure that any event will have some location (collisions may happen) in the feature space. In addition, this feature can also be used to limit the dimensions of the model. Developers can limit how many events they would like to handle by reducing the entries of the hashing algorithm. On the contrary, when the number of entries for a game is too small, some hashing collision may frequently happen, which might affect the performance of the model. Future research might focus on investigating the relationships between the number of hash entries and the performance of the model to find optimal solutions for this alternative method.

In addition, as has been introduced, the concept behind event-frequency-based data representation is similar to that behind the 'bag-of-words' algorithm (Wallach, 2006) used in NLP (natural-language processing) problems. From the perspective of NLP, this method treats a game as an article for text mining. Therefore, similarly, to improve this event-frequency-based data representation method, other text-mining approaches such as words embedding (Levy and Goldberg, 2014) might also be brought in for further analysis.

Sequence Mining Data Representation and Transfer Learning

As has been reviewed in Section 3.7, Martínez and Yannakakis (2011) applied sequence mining for behaviour modelling method achieving better generality. This idea can also be used as a variation of the event-frequency-based data representation. In order to investigate the sequence patterns, instead of aggregating the counts of events happened in game, an alternative way is to represent a player's behaviour as a list of events ordered by timestamps. Although similar methods of what Martínez and Yannakakis

(2011) did can be used for building up the feature space from this point, alternatively, because temporal information (timestamp) is included in this case, deep learning models such as Convolutional Neural Networks (Collobert et al., 2011) and Long-Short Term Memory (Hochreiter and Schmidhuber, 1997) might be used for automatically computing the feature space from the raw sequences of events in their shallow layers. The benefits of doing so is that transfer learning methods can be then naturally applied to these models. For example, as introduced by Oquab et al. (2014), while a CNN model needs to be transferred to another dataset, while making sure the input shape the same in both dataset (can be achieved by fixing number of events observed or grouping them into same sized chunks), one can freeze the parameters of the CNN layers while adapting (re-train) the deeper layers of the models to the new dataset (which is lack of data samples). Similarly, in an LSTM model, one can freeze the LSTM layers for the same purpose. This research direction may be further investigate as it could help to bring the rapidly developed deep learning approaches to the event frequency based data representation for better predictive abilities.

Trend Over Varying Dates

Preliminary experiments (details in Appendix A) show that disengagement-over-varying-dates labelling is able to provide an effectively trained classifier should the original labelling method (such as churn or disengagement) lead to biased or quantitatively insufficient datasets. Except for predicting players' retention-related concept, this labelling method might also be extended to substitutions of other predictive targets that often lead to biased or small datasets. For example, while predicting whether a current paying user might still pay in the future, players are usually categorized into binary classes. In this case, the class distribution might often be biased to 'won't pay for other things'. In the original disengagement-over-varying-dates labelling method, players are distributed into different classes by observing if those who have finished *prr* rounds of games can still play another *por* rounds. Similarly, in the context of purchasing behaviours, to achieve a better environment (more balance, more training samples) for training classifiers, the strategy for labelling players can be changed to consider whether a player can still purchase another *por* item after they have purchased *prr* items.

8.4 Closing Remarks

This work has delivered a generic data-representation method that can be migrated to different games for predicting player behaviours. It has achieved competitive results in most cases introduced in this work. Because of its generality, applications of this data-representation method can easily maintain a consistent pipeline across games. Recent research has shown that, although the game industry is growing quickly and its expected growth is faster than that of the film industry in 2013 and 2018, the life cycles of games (especially for mobile games) are, on the contrary, getting shorter (Egenfeldt-Nielsen et al., 2016). This situation may create extra application opportunities for this method, because, when more games are produced in a shorter period of time, a predictive system designed based on event-frequency-based data representation may remain the same.

Appendices

Appendix A

Disengagement Over Varying Dates

A.1 Definition

Regarding its definition, to solve the problem faced by the current definitions, the class labels of players in disengagement predictions are defined by two varying parameters: ***prr*** (prior rounds) and ***por*** (post rounds). Prior rounds stands for the quantities of rounds that a player has completed before a splitting date $\boldsymbol{T}$ (T may vary for different players), whilst post rounds represents the number of rounds he/she finished after that date. Note that *prr* and *por* are parameters that need to be manually decided, whereas T is simply the date when a player finished *prr* rounds of games. Based on this, a player is considered to be disengaging who finished *prr* rounds before some T but cannot finish *por* rounds afterwards. Equation A.1 shows a formal way to describe this labelling method. To get the best distribution between the resultant two classes, an optimisation algorithm such as genetic algorithm (which is used in this work) is recommended for finding the best combination of *prr* and *por*. The optimal combination of parameters can not only help to group players into more balanced classification tasks; it also give extra insight into the game. The details are introduced in Section A.2. In this work, the genetic algorithm was used with 5000 generations, 10 candidates and 0.5 mutation rate to find the smallest distance between the two classes.

From the perspective of predictive purpose, unlike traditional churn–which aims to describe players who are entirely leaving the game–this disengagement over varying dates stresses detecting disengaging trends, which is similar to the original disengagement-labelling method introduced in Chapter 6.

A.2 Insights from parameters

As has been mentioned, whenever the optimal combination of *prr* and *por* is found, extra important insights can be fetched from them. After optimisation, the resultant *prr* can be seen as an indicator of the game's health with regard to retention. Because players are evenly split into disengaging and engaging groups after the date T, a higher *prr* shows that the game keeps players engaged longer: i.e., most players have played many (*prr*) rounds before half of them will show a disengaging trend. On the other hand, a lower *prr* indicates that half of the players start to display a disengaging trend after only a few plays. This suggests that a negative first impression of the game is an important factor in discouraging players from continuing to engage with the game. Additionally, *por* indicates how long a company has to prevent players' disengaging decisions by attempting an intervention (e.g. offering in-game

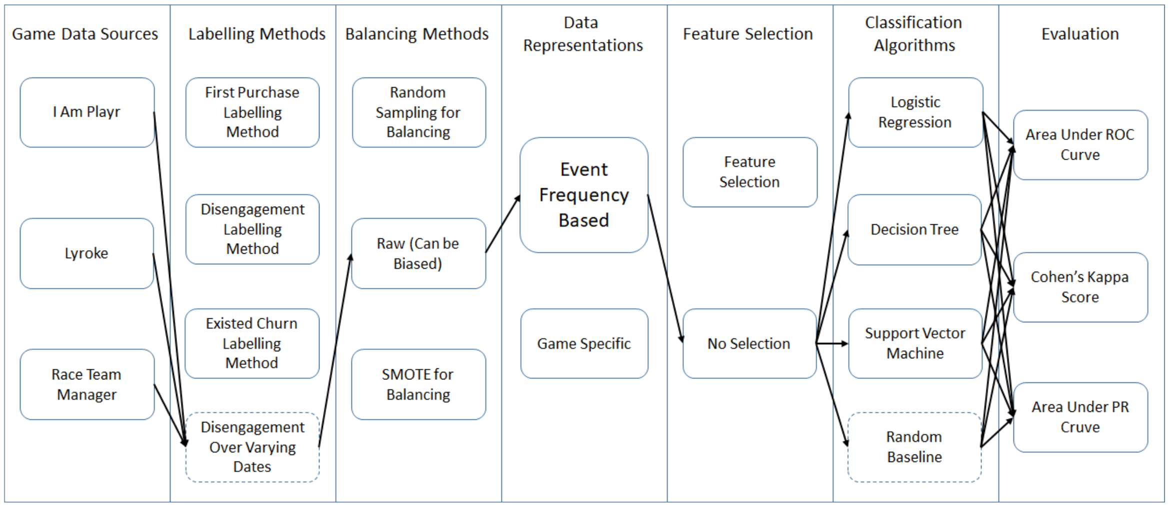

Figure A.1: Experiment of Disengagement Over Varying Dates Prediction

bonuses). A larger *por* means that most players are able to play many rounds of games after T and before disengaging, whilst on the contrary, a small *por* means that most players will disengage soon after T.

A.3 Experiment Details and Results

This section defines and considers the benefits of the new labelling method named disengagement over varying dates. To determine whether it is a competitive replacement for the disengagement or original churn method, its application to all three games used in previous experiments is shown in the next section. It is notable that, when the labelling method has been changed, the predictive problem cannot be considered the same as the previous one because the datasets are different. Because of this, classifiers trained with the disengagement over varying date-labelling methods will not be compared with the performance of classifiers trained with either the disengagement-labelling method or the churn-labelling method. Instead, the experiments will first focus on showing whether disengagement-over-varying-dates labelling results in a well-balanced dataset which uses all available data points to ensure a quantitatively sufficient dataset for dealing with highly dimensional algorithms. By comparing random classifiers, experiments will also show if the classifiers can perform well with the highly dimensional event-frequency-based data representation when the dataset has been labelled by this definition. Experiments in this section follow the procedure in Figure A.1.

$$\begin{aligned}
&\text{total} = \textit{total rounds played} \\
&\text{player label} = \begin{cases} \textit{disengaging}, & \text{if } \textit{total} - \textit{prr} < \textit{por} \\ \textit{engaging}, & \text{otherwise} \end{cases}
\end{aligned} \tag{A.1}$$

I Am Playr

In *I Am Playr* , though balanced by disengagement labelling, the dataset is highly biased with 1,354 disengaging and 12,044 disengaging players (1:8.895). In addition, when the churn-labelling method was applied, although the dataset is not seriously

Table A.1: Performance for Comparisons on Disengagement Over Varying Dates on the dataset of *I Am Playr*

		Event Feature without Balancing	Random Classifier	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (disengaging over varying dates)	LogisticRegression	**0.97±0.0009**	0.53	4.9431e+02	2.8897e-21	221.0601
	DecisionTree	**0.97±0.0006**	0.53	6.4673e+02	2.5723e-22	289.2264
	SVM	**0.95±0.0020**	0.53	2.0674e+02	7.3745e-18	92.4556
AUPRC (non disengaging over varying dates)	LogisticRegression	**0.96±0.0010**	0.47	4.8341e+02	3.5312e-21	216.1897
	DecisionTree	**0.95±0.0014**	0.47	3.2613e+02	1.2195e-19	145.8502
	SVM	**0.91±0.0060**	0.47	6.8851e+01	1.4531e-13	30.7913
AUROC	LogisticRegression	**0.97±0.0008**	0.50	5.4234e+02	1.2543e-21	242.5407
	DecisionTree	**0.96±0.0011**	0.50	4.0607e+02	1.6959e-20	181.5987
	SVM	**0.94±0.0030**	0.50	1.3716e+02	2.9574e-16	61.3408
KAPPA	LogisticRegression	**0.79±0.0031**	0.00	2.4340e+02	1.6971e-18	108.8522
	DecisionTree	**0.77±0.0039**	0.00	1.8886e+02	1.6641e-17	84.4604
	SVM	**0.72±0.0067**	0.00	1.0212e+02	4.1999e-15	45.6706

imbalanced, the number of available examples is only 256 (132 churners and 124 non-churners). On the contrary, disengagement over varying dates was able to provide better training conditions. While the labelling method was applied, the *prr* and *por* found by the genetic algorithm, which minimised the distances between the two classes were both 2s in *I Am Playr* . This optimal combination of parameters suggests that the game is not retaining players very well because half of the players can only finish two rounds before deciding to leave the game. In addition, because *por* is also two, it means that players that decide to leave will churn after two rounds. This suggests that actions need to be made quickly to prevent this trend. While being categorised by the *prr* and *por*, the ratio of the number of players in two classes is 1:1.118, which is close to balance. Regarding the number of data samples, there are 19,243 positive examples (who are unable to finish two rounds after two gameplay rounds) and 17,217 negative examples (who are still able to finish two rounds after two gameplay rounds). This is much larger than the dimension of the event frequency-based data representation, which is 4,740.

Results for predicting disengagement over varying dates can be found in Table A.1. As can be seen, because the dataset is both quantitatively sufficient and close to balanced, the performance of classifiers trained in this case behaves significantly better than the random guess. This can also be verified by the effect-size column. To verify the observation found, experiments will also be tested on the other two games.

Lyroke

Similar to *I Am Playr* , the disengagement-labelled dataset in *Lyroke* is highly biased to one side (3,495 disengaging players and 25,012 non-disengaging users) whereas the churn-labelled dataset is balanced but lacks data samples (127 churners and 103 non-churner). To improve the situation, disengagement over varying dates was applied as well. When applied, the optimal combination of *prr* and *por* is found to be one and two, respectively. Compared to *I Am Playr* , the health of *Lyroke* is worse, as the *prr* is smaller, which means half its players can finish only one round of the game before they churn. Because *por* is two, which is the same as the *por* of *I Am Playr* , quick actions need to made before players disengage.

Table A.2 depicts the experiment for predicting players' disengagement over varying

Table A.2: Performance for Comparisons on Disengagement Over Varying Dates on the dataset of *Lyroke*

		Event Feature without Balancing	Random Classifier	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC (disengaging over varying dates)	LogisticRegression	**0.98±0.0004**	0.59	9.0437e+02	1.2582e-23	404.4465
	DecisionTree	**0.97±0.0002**	0.59	1.4506e+03	1.7899e-25	648.7470
	SVM	**0.97±0.0007**	0.59	4.9656e+02	2.7735e-21	222.0701
AUPRC (non disengaging over varying dates)	LogisticRegression	**0.96±0.0006**	0.41	9.3065e+02	9.7222e-24	416.2005
	DecisionTree	**0.96±0.0003**	0.41	1.9206e+03	1.4318e-26	858.9276
	SVM	**0.95±0.0017**	0.41	3.0056e+02	2.5431e-19	134.4129
AUROC	LogisticRegression	**0.97±0.0005**	0.50	9.3238e+02	9.5615e-24	416.9721
	DecisionTree	**0.96±0.0003**	0.50	1.4482e+03	1.8168e-25	647.6722
	SVM	**0.96±0.0012**	0.50	3.6281e+02	4.6733e-20	162.2540
KAPPA	LogisticRegression	**0.81±0.0027**	0.00	2.8528e+02	4.0666e-19	127.5810
	DecisionTree	**0.79±0.0019**	0.00	3.8599e+02	2.6765e-20	172.6203
	SVM	**0.76±0.0039**	0.00	1.8636e+02	1.8760e-17	83.3427

dates in *Lyroke* . As can be seen from the table, the performance of classifiers trained with event-frequency-based data representation under this balanced and quantitatively sufficient dataset is better than that of the random classifiers in all cases. This matches what has been observed in the experiment of *Lyroke*. Finally, because *I Am Playr* and *Lyroke* are developed by the same developer, the same labelling method was also applied on *Race Team Manager* .

Race Team Manager

Unlike the two previous games, *Race Team Manager* is special not only because it is developed by a different company but also because its dataset is more challenging. As is reviewed in Section 7.4 and 7.7, when it is labelled by the churn-labelling method, the number of data samples available for training is much more limited than in the other games. This makes matters very challenging for highly dimensional data representations such as the event-frequency-based approach. In addition, the dataset also displays high bias towards the positive (churning) side, which is yet another factor that may affect the quality of trained classifiers. Experiments show that both the random-undersampling method and SMOTE are unable to improve performance significantly. Therefore, the target of this experiment is focused on investigating whether the dataset labelled by disengagement over varying dates can provide a better training environment for the highly dimensional classifiers that result in more reliable performance. During the experiment, when disengagement over varying dates was applied, the best combination found by the genetic algorithm was $prr = 3$ and $por = 1$, respectively. Unlike the previous two games, half of the players are able to play three rounds of the game before they leave it, which is better than the other two games. However, given that por is only one, there is an urgent requirement for this game to prevent the trend of player disengagement. This can also be verified by the fact that, when churn was used as the labelling method, more players are labelled as churners than non-churners. After being labelled by disengagement over varying dates, there are 62,355 disengaging players and 51,518 non-disengaging players, which is close to balanced.

As can be seen from Table A.3, as before, the classifiers trained with event-frequency-based data representation are able to perform significantly better than the random classifiers. This difference can also be verified with the effect size.

Table A.3: Performance for Comparisons on Disengagement Over Varying Dates on the dataset of *Race Team Manager*

		Event Feature without Balancing	Random Classifier	T-Value	P-Value	Effect-Size (Cohen's D)
AUPRC	LogisticRegression	**0.98±0.0005**	0.55	7.7313e+02	5.1588e-23	345.7564
(disengaging	DecisionTree	**0.98±0.0006**	0.55	6.4817e+02	2.5213e-22	289.8706
over varying	SVM	**0.99±0.0005**	0.55	8.9983e+02	1.3164e-23	402.4173
dates) AUPRC (non	LogisticRegression	**0.98±0.0003**	0.45	1.5306e+03	1.1043e-25	684.5068
disengaging	DecisionTree	**0.98±0.0004**	0.45	1.3281e+03	3.9605e-25	593.9504
over varying	SVM	**0.98±0.0002**	0.45	2.3330e+03	2.4864e-27	1043.3645
dates)	LogisticRegression	**0.99±0.0003**	0.50	1.6530e+03	5.5265e-26	739.2369
AUROC	DecisionTree	**0.98±0.0005**	0.50	9.1315e+02	1.1534e-23	408.3721
	SVM	**0.98±0.0002**	0.50	2.1068e+03	6.2275e-27	942.1757
	LogisticRegression	**0.88±0.0010**	0.00	8.7478e+02	1.6972e-23	391.2150
KAPPA	DecisionTree	**0.87±0.0011**	0.00	7.7735e+02	4.9124e-23	347.6417
	SVM	**0.87±0.0010**	0.00	8.6994e+02	1.7842e-23	389.0482

A.4 Summary

This section introduced a new disengagement-labelling approach named *disengagement over varying dates* that is able to maintain an approximately balanced distribution of resultant classes without losing any data samples. In a disengagement prediction, rather than selecting disengaging players by hard-coded specific conditions, this method partitions the database by defining a varying splitting date that is controlled and generated from two constant parameters, *prr* and *por*, for individual players. A player was said to be disengaging or not by Equation A.1. To get the smallest distance between the resultant classes, a searching algorithm is needed for optimising the two parameters. In this work, a standard genetic algorithm was applied to the optimisation; but other methods (e.g., gradient search) may also work. To evaluate the performance of this labelling method, it was first applied to three different commercial games for creating approximately balanced classes, and the classifiers trained in this situation were compared to those that were trained in the dataset labelled by the original churn-labelling method with balancing processes. In all three games, disengagement over varying dates successfully balances the distribution of classes (between disengagement and non-disengagement) and provides more reliable classifiers.

It is notable that, though disengagement over varying dates is a concept similar to that involved in the widely used churn or disengagement-labelling methods, they are still different predictive tasks. Although the predictive targets are not exactly the same, this method can be a good substitution whenever a reliable classifier cannot be successfully trained given the original churn or disengagement. This can be considered a trade-off between *an original predictive target with unreliable classifiers* and *a similar predictive target with much more reliable classifiers plus extra game-health indicator parameters*, where the latter is competitive in terms of benefits for practical problems.

Reference

Mohamed Abou-Zleikha and Noor Shaker. Evolving random forest for preference learning. In *European Conference on the Applications of Evolutionary Computation*, pages 318–330. Springer, 2015.

Edgar Acuña and Caroline Rodriguez. On detection of outliers and their effect in supervised classification. *University of Puerto Rico at Mayaguez*, 2004.

Lalita Admuthe, Shaila Apte, and Sunil Admuthe. Topology and parameter optimization of ann using genetic algorithm for application of textiles. In *Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, 2009. IDAACS 2009. IEEE International Workshop on*, pages 278–282. IEEE, 2009.

Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. Mining association rules between sets of items in large databases. *ACM SIGMOD Record*, 22(2):207–216, 1993.

M.A. Ahmad, B. Keegan, J. Srivastava, D. Williams, and N. Contractor. Mining for gold farmers: Automatic detection of deviant players in mmogs. In *Computational Science and Engineering, 2009. CSE '09. International Conference on*, volume 4, pages 340–345, Aug 2009. doi: 10.1109/CSE.2009.307.

AI-Factory. Spades. `http://www.aifactory.co.uk/`, 2011.

Fabio Aiolli and Claudio Enrico Palazzi. Enhancing artificial intelligence in games by learning the opponent's playing style. In *New Frontiers for Entertainment Computing*, pages 1–10. Springer, 2008.

E. Alpaydin. *Introduction to Machine Learning*. Adaptive computation and machine learning. MIT Press, 2004. ISBN 9780262012119.

E. Alpaydin. *Introduction to machine learning*. Adaptive computation and machine learning. MIT Press, 2010. ISBN 9780262012430.

Joao Alves, Sascha Lange, Michael Lenz, and Martin Riedmiller. Case study: behavioral prediction of future revenues in freemium games. In *Workshop New Challenges in Neural Computation 2014*, page 26. Citeseer, 2014.

Chidanand Apté and Sholom Weiss. Data mining with decision trees and decision rules. *Future Generation Computer Systems*, 13(2–3):197 – 210, 1997. ISSN 0167-739X. Data Mining.

Sylvain Arlot, Alain Celisse, et al. A survey of cross-validation procedures for model selection. *Statistics surveys*, 4:40–79, 2010.

Sander CJ Bakkes, Pieter HM Spronck, and H Jaap Van Den Herik. Opponent modelling for case-based adaptive game ai. *Entertainment Computing*, 1(1):27–37, 2009.

Daniel Barbar'a, William DuMouchel, Christos Faloutsos, Peter J Haas, Joseph M Hellerstein, Yannis Ioannidis, HV Jagadish, Theodore Johnson, Raymond Ng, Viswanath Poosala, et al. The new jersey data reduction report. In *IEEE Data Engineering Bulletin*. Citeseer, 1997.

C. Bauckhage, K. Kersting, R. Sifa, C. Thurau, A. Drachen, and A. Canossa. How players lose interest in playing a game: An empirical study based on distributions of total playing times. In *Computational Intelligence and Games (CIG), 2012 IEEE Conference on*, pages 139–146, Sept 2012.

Christian Bauckhage, Christian Thurau, and Gerhard Sagerer. Learning human like opponent behavior for interactive computer games. In *Pattern Recognition*, pages 148–155. Springer, 2003.

Christian Bauckhage, Anders Drachen, and Rafet Sifa. Clustering game behavior data. *Computational Intelligence and AI in Games, IEEE Transactions on*, 7(3):266–278, 2015.

Robin Baumgarten. Towards automatic player behaviour characterisation using multiclass linear discriminant analysis. In *Proceedings of the AISB Symposium: AI and Games*, 2010.

Bigbit-Ltd. Race team manager. [App Store], 2014.

Blizzard. World of warcraft. [Online], 2004.

Blizzard. Hearthstone. Blizzard BattleNet, 2014.

Bluehole. Tera: Rising. [DVD-ROM], 2011.

Zoheb Borbora, Jaideep Srivastava, Kuo-Wei Hsu, and Dmitri Williams. Churn prediction in mmorpgs using player motivation theories and an ensemble approach. In *Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third Inernational Conference on Social Computing (SocialCom), 2011 IEEE Third International Conference on*, pages 157–164. IEEE, 2011.

Zoheb H Borbora and Jaideep Srivastava. User behavior modelling approach for churn prediction in online games. In *Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Confernece on Social Computing (SocialCom)*, pages 51–60. IEEE, 2012.

Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

RW Butler, PL Davies, and M Jhun. Asymptotics for the minimum covariance determinant estimator. *The Annals of Statistics*, pages 1385–1400, 1993.

Elizabeth Camilleri, Georgios N Yannakakis, and Antonios Liapis. Towards general models of player affect. In *Affective Computing and Intelligent Interaction (ACII), 2017 International Conference on*, 2017.

C. Campbell and Y. Ying. *Learning with Support Vector Machines*. Synthesis lectures on artificial intelligence and machine learning. Morgan & Claypool, 2011. ISBN 9781608456161.

Rich Caruana and Alexandru Niculescu-Mizil. An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd international conference on Machine learning*, pages 161–168. ACM, 2006.

Darryl Charles and Michaela Black. Dynamic player modeling: A framework for player-centered digital games. In *Proc. of the International Conference on Computer Games: Artificial Intelligence, Design and Education*, pages 29–35, 2004.

Nitesh V Chawla. Data mining for imbalanced datasets: An overview. In *Data mining and knowledge discovery handbook*, pages 853–867. Springer, 2005.

Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16: 321–357, 2002.

Wei Chu and Zoubin Ghahramani. Preference learning with gaussian processes. In *Proceedings of the 22nd international conference on Machine learning*, pages 137–144. ACM, 2005.

Jacob Cohen. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46, 1960.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537, 2011.

Gordon V Cormack. Email spam filtering: A systematic review. *Foundations and Trends in Information Retrieval*, 1(4):335–455, 2007.

Peter I Cowling, Sam Devlin, Edward J Powley, Daniel Whitehouse, and Jeff Rollason. Player preference and style in a leading mobile card game. *IEEE Transactions on Computational Intelligence and AI in Games*, 7(3):233–242, 2015.

Jesse Davis and Mark Goadrich. The relationship between precision-recall and roc curves. In *Proceedings of the 23rd international conference on Machine learning*, pages 233–240. ACM, 2006.

Thomas Debeauvais, Cristina V Lopes, Nick Yee, and Nicolas Ducheneaut. Retention and progression: Seven months in world of warcraft. In *Proceedings of the 9th international conference on foundations of digital games*, 2014.

David J Dittman, Taghi M Khoshgoftaar, Randall Wald, and Amri Napolitano. Comparison of data sampling approaches for imbalanced bioinformatics data. In *FLAIRS Conference*, 2014.

A Rogier T Donders, Geert JMG van der Heijden, Theo Stijnen, and Karel GM Moons. Review: a gentle introduction to imputation of missing values. *Journal of clinical epidemiology*, 59(10):1087–1091, 2006.

A. Drachen, A. Canossa, and G.N. Yannakakis. Player modeling using self-organization in tomb raider: Underworld. In *Computational Intelligence and Games, 2009. CIG 2009. IEEE Symposium on*, pages 1–8, Sept 2009. doi: 10.1109/CIG.2009.5286500.

A. Drachen, R. Sifa, C. Bauckhage, and C. Thurau. Guns, swords and data: Clustering of player behavior in computer games in the wild. In *Computational Intelligence and Games (CIG), 2012 IEEE Conference on*, pages 163–170, Sept 2012. doi: 10.1109/CIG.2012.6374152.

Anders Drachen, Magy Seif El-Nasr, and Alessandro Canossa. Game analytics – the basics. In Magy Seif El-Nasr, Anders Drachen, and Alessandro Canossa, editors, *Game Analytics*, pages 13–40. Springer London, 2013. ISBN 978-1-4471-4768-8. doi: 10.1007/978-1-4471-4769-5_2.

Anders Drachen, Eric Thurston Lundquist, Yungjen Kung, Pranav Simha Rao, Diego Klabjan, Rafet Sifa, and Julian Runge. Rapid prediction of player retention in free-to-play mobile games. *arXiv preprint arXiv:1607.03202*, 2016.

EA. Battlefield 2: Bad company 2. [DVD-ROM], 2010.

S. Egenfeldt-Nielsen, J.H. Smith, and S.P. Tosca. *Understanding Video Games: The Essential Introduction*. Routledge, 2016. ISBN 9781317533139.

Eidos-Interactive. Tomb raider: Underworld. [DVD-ROM], 2008.

Sony Entertainment. Ever quest ii. `https://www.everquest2.com/home`, 2004.

P. Flach. *Machine Learning: The Art and Science of Algorithms that Make Sense of Data*. Cambridge University Press, 2012. ISBN 9781107096394.

Susan M Fox-Wasylyshyn and Maher M El-Masri. Handling missing data in self-report measures. *Research in nursing & health*, 28(6):488–495, 2005.

Johannes Fürnkranz and Eyke Hüllermeier. Preference learning: An introduction. In *Preference learning*, pages 1–17. Springer, 2010.

Robin Genuer, Jean-Michel Poggi, and Christine Tuleau-Malot. Variable selection using random forests. *Pattern Recognition Letters*, 31(14):2225–2236, 2010.

Zoubin Ghahramani. Unsupervised learning. In *Advanced lectures on machine learning*, pages 72–112. Springer, 2004.

Google. Google analytics. `https://analytics.google.com/analytics/web/`, 2005.

Jeremy Gow, Robin Baumgarten, Paul Cairns, Simon Colton, and Paul Miller. Unsupervised modeling of player style with lda. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(3):152–166, 2012.

Rolf HH Groenwold, Ian R White, A Rogier T Donders, James R Carpenter, Douglas G Altman, and Karel GM Moons. Missing covariate data in clinical research: when and when not to use the missing-indicator method for analysis. *Canadian Medical Association Journal*, 184(11):1265–1269, 2012.

Jerzy W Grzymala-Busse, Jerzy Stefanowski, and Szymon Wilk. A comparison of two approaches to data mining from imbalanced data. *Journal of Intelligent Manufacturing*, 16 (6):565–573, 2005.

Qiong Gu, Zhihua Cai, Li Zhu, and Bo Huang. Data mining on imbalanced data sets. In *Advanced Computer Theory and Engineering, 2008. ICACTE'08. International Conference on*, pages 1020–1024. IEEE, 2008.

Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of machine learning research*, 3(Mar):1157–1182, 2003.

Fabian Hadiji, Rafet Sifa, Anders Drachen, Christian Thurau, Kristian Kersting, and Christian Bauckhage. Predicting player churn in the wild. In *Proceedings of the Conference on Computational Intelligence and Games (CIG)*, 2014.

J. Han and M. Kamber. *Data Mining: Concepts and Techniques: Concepts and Techniques.* The Morgan Kaufmann Series in Data Management Systems. Elsevier Science, 2011. ISBN 9780123814807.

J. Han, J. Pei, and M. Kamber. *Data Mining, Southeast Asia Edition.* The Morgan Kaufmann Series in Data Management Systems. Elsevier Science, 2006. ISBN 9780080475585.

D.J. Hand, H. Mannila, and P. Smyth. *Principles of Data Mining.* A Bradford book. MIT Press, 2001. ISBN 9780262082907.

Martin Hilbert. Big data for development: From information-to knowledge societies. *Available at SSRN 2205145*, 2013.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

D.W. Hosmer and S. Lemeshow. *Applied Logistic Regression.* Applied Logistic Regression. Wiley, 2004. ISBN 9780471654025.

Wang Hua, Ma Cuiqin, and Zhou Lijuan. A brief review of machine learning and its application. In *Information Engineering and Computer Science, 2009. ICIECS 2009. International Conference on*, pages 1–4, 2009. doi: 10.1109/ICIECS.2009.5362936.

Robin Hunicke and Vernell Chapman. Ai for dynamic difficulty adjustment in games. In *Challenges in Game Artificial Intelligence AAAI Workshop*, pages 91–96. sn, 2004.

László A Jeni, Jeffrey F Cohn, and Fernando De La Torre. Facing imbalanced data–recommendations for the use of performance metrics. In *Affective Computing and Intelligent Interaction (ACII), 2013 Humaine Association Conference on*, pages 245–251. IEEE, 2013.

Ah Reum Kang, Huy Kang Kim, and Jiyoung Woo. Chatting pattern based game bot detection: do they talk like us? *KSII Transactions on Internet & Information Systems*, 6 (11), 2012.

K Senthamarai Kannan, P Sailapathi Sekar, M Mohamed Sathik, and P Arumugam. Financial stock market forecast using data mining techniques. In *Proceedings of the International Multiconference of Engineers and computer scientists*, volume 1, page 4, 2010.

Jaya Kawale, Aditya Pal, and Jaideep Srivastava. Churn prediction in mmorpgs: A social influence based approach. In *Computational Science and Engineering, 2009. CSE'09. International Conference on*, volume 4, pages 423–428. IEEE, 2009.

Jens Keilwagen, Ivo Grosse, and Jan Grau. Area under precision-recall curves for weighted and unweighted data. *PLoS One*, 9(3):e92209, 2014.

Jin Baek Kim. An empirical study on consumer first purchase intention in online shopping: integrating initial trust and tam. *Electronic Commerce Research*, 12(2):125–150, 2012.

Ben Kirman and Shaun Lawson. Hardcore classification: Identifying play styles in social games using network analysis. In *Entertainment Computing–ICEC 2009*, pages 246–251. Springer, 2009.

P. Laurens, R.F. Paige, P.J. Brooke, and H. Chivers. A novel approach to the detection of cheating in multiplayer online games. In *Engineering Complex Computer Systems, 2007. 12th IEEE International Conference on*, pages 97–106, July 2007. doi: 10.1109/ICECCS.2007.11.

Seong Jae Lee, Yun-En Liu, and Zoran Popovic. Learning individual behavior in an educational game: A data-driven approach. In *Educational Data Mining 2014*, 2014.

Yunjin Lee, Seungyong Lee, Ioannis Ivrissimtzis, and Hans-Peter Seidel. Overfitting control for surface reconstruction. In *Symposium on Geometry Processing*, pages 231–234. Citeseer, 2006.

Omer Levy and Yoav Goldberg. Dependency-based word embeddings. In *ACL (2)*, pages 302–308, 2014.

Chong-U Lim and D.F. Harrell. Modeling player preferences in avatar customization using social network data: A case-study using virtual items in team fortress 2. In *Computational Intelligence in Games (CIG), 2013 IEEE Conference on*, pages 1–8, 2013. doi: 10.1109/CIG.2013.6633636.

Roderick JA Little. Regression with missing x's: a review. *Journal of the American Statistical Association*, 87(420):1227–1237, 1992.

Rushi Longadge and Snehalata Dongre. Class imbalance problem in data mining review. *arXiv preprint arXiv:1305.1707*, 2013.

Victoria López, Alberto Fernández, Salvador García, Vasile Palade, and Francisco Herrera. An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. *Information Sciences*, 250:113–141, 2013.

T. Mahlmann, A. Drachen, J. Togelius, A. Canossa, and G.N. Yannakakis. Predicting player behavior in tomb raider: Underworld. In *Computational Intelligence and Games (CIG), 2010 IEEE Symposium on*, pages 178–185, 2010. doi: 10.1109/ITW.2010.5593355.

André Marchand and Thorsten Hennig-Thurau. Value creation in the video game industry: Industry economics, consumer benefits, and research opportunities. *Journal of Interactive Marketing*, 27(3):141–157, 2013.

RB Marimont and MB Shapiro. Nearest neighbour searches and the curse of dimensionality. *IMA Journal of Applied Mathematics*, 24(1):59–70, 1979.

Markus-Persson. Infinite mario bros. [Online], 2008.

Héctor P Martínez and Georgios N Yannakakis. Mining multimodal sequential patterns: a case study on affect detection. In *Proceedings of the 13th international conference on multimodal interfaces*, pages 3–10. ACM, 2011.

Héctor P Martínez, Kenneth Hullett, and Georgios N Yannakakis. Extending neuro-evolutionary preference learning through player modeling. In *Computational Intelligence and Games (CIG), 2010 IEEE Symposium on*, pages 313–320. IEEE, 2010.

Olana Missura and Thomas Gärtner. Player modeling for intelligent difficulty adjustment. In João Gama, VítorSantos Costa, AlípioMário Jorge, and PavelB. Brazdil, editors, *Discovery Science*, volume 5808 of *Lecture Notes in Computer Science*, pages 197–211. Springer Berlin Heidelberg, 2009. ISBN 978-3-642-04746-6. doi: 10.1007/978-3-642-04747-3_17.

Stefan Mitterhofer, Christian Platzer, Christopher Kruegel, and Engin Kirda. Server-side bot detection in massively multiplayer online games. *IEEE Security & Privacy*, 7(3):0029–36, 2009.

K.P. Murphy. *Machine Learning: A Probabilistic Perspective*. Adaptive computation and machine learning series. MIT Press, 2012. ISBN 9780262018029.

Folorunsho Olaiya and Adesesan Barnabas Adeyemo. Application of data mining techniques in weather prediction and climate change studies. *International Journal of Information Engineering and Electronic Business*, 4(1):51, 2012.

Maxime Oquab, Leon Bottou, Ivan Laptev, and Josef Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

GV Otari and RV Kulkarni. A review of application of data mining in earthquake prediction. *International Journal of Computer Science and Information Technologies*, 3(2):3570–3574, 2012.

Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010.

C. Pedersen, J. Togelius, and G.N. Yannakakis. Modeling player experience in super mario bros. In *Computational Intelligence and Games, 2009. CIG 2009. IEEE Symposium on*, pages 132–139, 2009. doi: 10.1109/CIG.2009.5286482.

C. Pedersen, J. Togelius, and G.N. Yannakakis. Modeling player experience for content creation. *Computational Intelligence and AI in Games, IEEE Transactions on*, 2(1):54–67, March 2010. ISSN 1943-068X. doi: 10.1109/TCIAIG.2010.2043950.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Ondřej Pluskal and Jan Šedivý. Predicting players behavior in games with microtransactions. In *STAIRS 2014: Proceedings of the 7th European Starting AI Researcher Symposium*, 2014.

Yinsheng Qu, Bao-ling Adam, Mark Thornquist, John D Potter, Mary Lou Thompson, Yutaka Yasui, John Davis, Paul F Schellhammer, Lisa Cazares, MaryAnn Clements, et al. Data reduction using a discrete wavelet transform in discriminant analysis of very high dimensionality data. *Biometrics*, 59(1):143–151, 2003.

Alex Randolph. Ghosts. [Board Game], 1980.

Sarunas J Raudys, Anil K Jain, et al. Small sample size effects in statistical pattern recognition: Recommendations for practitioners. *IEEE Transactions on pattern analysis and machine intelligence*, 13(3):252–264, 1991.

L. Rokach and O. Maimon. *Data Mining with Decision Trees: Theory and Applications*. World Scientific Publishing Company, 2014. ISBN 9789814590099.

Peter J Rousseeuw and Katrien Van Driessen. A fast algorithm for the minimum covariance determinant estimator. *Technometrics*, 41(3):212–223, 1999.

Julian Runge, Peng Gao, Florent Garcin, and Boi Faltings. Churn prediction for high-value players in casual social games. *Proc. IEEE CIG*, 2014.

Takaya Saito and Marc Rehmsmeier. The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. *PloS one*, 10(3): e0118432, 2015.

Claude Sammut and Geoffrey I Webb. *Encyclopedia of machine learning*. Springer Science & Business Media, 2011.

M Saraee, S White, and J Eccleston. A data mining approach to analysis and prediction of movie ratings. *Transactions of the Wessex Institute*, pages 343–352, 2004.

Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Analysis of recommendation algorithms for e-commerce. In *Proceedings of the 2nd ACM conference on Electronic commerce*, pages 158–167. ACM, 2000.

Songwon Seo. *A review and comparison of methods for detecting outliers in univariate data sets*. PhD thesis, University of Pittsburgh, 2006.

Noor Shaker and Mohamed Abou-Zleikha. Transfer learning for cross-game prediction of player experience. In *Computational Intelligence and Games (CIG), 2016 IEEE Conference on*, pages 1–8. IEEE, 2016.

Noor Shaker, Georgios N Yannakakis, and Julian Togelius. Towards automatic personalized content generation for platform games. In *AIIDE*, 2010.

Noor Shaker, Mohammad Shaker, and Mohamed Abou-Zleikha. Towards generic models of player experience. In *Eleventh Artificial Intelligence and Interactive Digital Entertainment Conference*, 2015.

Rafet Sifa, Fabian Hadiji, Julian Runge, Anders Drachen, Kristian Kersting, and Christian Bauckhage. Predicting purchase decisions in mobile free-to-play games. In *Eleventh Artificial Intelligence and Interactive Digital Entertainment Conference*, 2015.

Noah A Smith. Log-linear models. *Cited by*, 3, 2004.

Jyoti Soni, Ujma Ansari, Dipesh Sharma, and Sunita Soni. Predictive data mining for medical diagnosis: An overview of heart disease prediction. *International Journal of Computer Applications*, 17(8):43–48, 2011.

Spring-Engine. Spring. `https://springrts.com/`, 2008.

SS Stevens. On the theory of scales of measurement. *SCIENCE*, 103(2684), 1946.

Ben Tan, Yangqiu Song, Erheng Zhong, and Qiang Yang. Transitive transfer learning. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1155–1164. ACM, 2015.

Pin-Yun Tarng, Kuan-Ta Chen, and Polly Huang. On prophesying online gamer departure. In *2009 8th Annual Workshop on Network and Systems Support for Games (NetGames)*, pages 1–2. IEEE, 2009.

Ruck Thawonmas, Masayoshi Kurashige, Keita Iizuka, and Mehmed Kantardzic. Clustering of online game users based on their trails using self-organizing map. In *Entertainment Computing-ICEC 2006*, pages 366–369. Springer, 2006.

C. Thurau and C. Bauckhage. Analyzing the evolution of social groups in world of warcraft;. In *Computational Intelligence and Games (CIG), 2010 IEEE Symposium on*, pages 170–177, 2010. doi: 10.1109/ITW.2010.5593358.

J. Togelius, R. De Nardi, and S.M. Lucas. Towards automatic personalised content creation for racing games. In *Computational Intelligence and Games, 2007. CIG 2007. IEEE Symposium on*, pages 252–259, April 2007. doi: 10.1109/CIG.2007.368106.

J. Togelius, G.N. Yannakakis, K.O. Stanley, and C. Browne. Search-based procedural content generation: A taxonomy and survey. *Computational Intelligence and AI in Games, IEEE Transactions on*, 3(3):172–186, Sept 2011. ISSN 1943-068X. doi: 10.1109/TCIAIG.2011.2148116.

Julian Togelius and Georgios N Yannakakis. General general game ai. In *Computational Intelligence and Games (CIG), 2016 IEEE Conference on*, pages 1–8. IEEE, 2016.

J.W. Tukey. *Exploratory Data Analysis*. Addison-Wesley series in behavioral science. Addison-Wesley Publishing Company, 1977. ISBN 9780201076165.

Anders Tychsen. Crafting user experience via game metrics analysis. In *Proceedings of the Workshop" Research Goals and Strategies for Studying User Experience and Emotion" at the 5th Nordic Conference on Human-computer interaction: building bridges (NordiCHI), Lund, Sweden*, pages 20–22, 2008.

Upsight. Upsight. `http://www.upsight.com/analytics`, 2007.

Valve. Team fortress 2. [Steam Platform], 2007.

Stefan Van Aelst and Peter Rousseeuw. Minimum volume ellipsoid. *Wiley Interdisciplinary Reviews: Computational Statistics*, 1(1):71–82, 2009.

Sofia Visa, Brian Ramsay, Anca L Ralescu, and Esther Van Der Knaap. Confusion matrix-based feature selection. In *MAICS*, pages 120–127, 2011.

Esteban Walker and Amy S Nowacki. Understanding equivalence and noninferiority testing. *Journal of general internal medicine*, 26(2):192–196, 2011.

Hanna M Wallach. Topic modeling: beyond bag-of-words. In *Proceedings of the 23rd international conference on Machine learning*, pages 977–984. ACM, 2006.

Juanjuan Wang, Mantao Xu, Hui Wang, and Jiwu Zhang. Classification of imbalanced data by using the smote algorithm and locally linear embedding. In *2006 8th international Conference on Signal Processing*, volume 3. IEEE, 2006.

Ben G Weber, Michael Mateas, and Arnav Jhala. Using data mining to model player experience. In *FDG Workshop on Evaluating Player Experience in Games*, 2011a.

Ben George Weber, Michael John, Michael Mateas, and Arnav Jhala. Modeling player retention in madden nfl 11. In *IAAI*, 2011b.

Jianping Wei. Research on data preprocessing in supermarket customers data mining. In *Information Engineering and Computer Science (ICIECS), 2010 2nd International Conference on*, pages 1–4, 2010. doi: 10.1109/ICIECS.2010.5677884.

Jiyoung Woo, Hwa Jae Choi, and Huy Kang Kim. An automatic and proactive identity theft detection model in mmorpgs. *Appl. Math*, 6(1S):291S–302S, 2012.

Wooga-Games. Jelly splash. [App Store], 2013.

Hanting Xie, Daniel Kudenko, Sam Devlin, and Peter Cowling. Predicting player disengagement in online games. In *Workshop on Computer Games*, pages 133–149. Springer, 2014.

Hanting Xie, Sam Devlin, Daniel Kudenko, and Peter Cowling. Predicting player disengagement and first purchase with event-frequency based data representation. In *2015 IEEE Conference on Computational Intelligence and Games (CIG)*, pages 230–237. IEEE, 2015.

Geogios N Yannakakis. Game ai revisited. In *Proceedings of the 9th conference on Computing Frontiers*, pages 285–292. ACM, 2012.

Georgios N Yannakakis, Manolis Maragoudakis, and John Hallam. Preference learning for cognitive modeling: a case study on entertainment preferences. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 39(6):1165–1175, 2009.

Georgios N Yannakakis, Pieter Spronck, Daniele Loiacono, and Elisabeth André. Player modeling. *Artificial and Computational Intelligence in Games*, 6:45–59, 2013.

G.N. Yannakakis and J. Togelius. Experience-driven procedural content generation. *Affective Computing, IEEE Transactions on*, 2(3):147–161, July 2011. ISSN 1949-3045. doi: 10.1109/T-AFFC.2011.6.

Yang C Yuan. Multiple imputation for missing data: Concepts and new development (version 9.0). *SAS Institute Inc, Rockville, MD*, 49, 2010.

Yin Zhang, Rong Jin, and Zhi-Hua Zhou. Understanding bag-of-words model: a statistical framework. *International Journal of Machine Learning and Cybernetics*, 1(1-4):43–52, 2010.

Georg Zoeller. Game development telemetry in production. In Magy Seif El-Nasr, Anders Drachen, and Alessandro Canossa, editors, *Game Analytics*, pages 111–135. Springer London, 2013. ISBN 978-1-4471-4768-8. doi: 10.1007/978-1-4471-4769-5_7.