```
# clear environment
rm(list = ls())

# load libraries
library('tm')
library('RWeka')
library('tabulizer')
library('dplyr')
library('pdftools')
library("SnowballC")
library("RColorBrewer")
library("wordcloud")
library("tau")
library("ggplot2")

# configure run
typeOfNGram <- 4
minTypeOfNGram <- typeOfNGram
maxTypeOfNGram <- typeOfNGram
scaleFactor <- 1.2
maxNrWordsInFrequencyPlot <- 24
fileWordsToAvoid<-"configuration/words-to-avoid.txt"
languageForAnalysis<-"english"

# define functions
Rpdf <- readPDF(engine = "xpdf", control = list(text = "-layout"))
GramTokenizer <- function(x) NGramTokenizer(x,
Weka_control(min=minTypeOfNGram,
                                                                max=maxTypeOfNGram))

# directory where the pdf files are stored
originalDir<-getwd()
pdfFilesDir<-"/home/kt529/Development/R/data/pdf/"
txtFilesDir<-"/home/kt529/Development/R/data/txt/MM/"
txtFilesInputDir<-"/home/kt529/Development/R/data/txt/MM/"
wipFilesDir<-"/home/kt529/Development/R/data/working/mm/"
configurationFileDir<-"/home/kt529/Development/R/data/working/"

filesTXT <- list.files(path = txtFilesInputDir, pattern = "txt",  full.names =
TRUE)

# configuration files
# listOfWordsToRemove <-
read.csv2(paste(wipFilesDir,fileWordsToAvoid,sep=""),header = F)
listOfWordsToRemove <-
read.csv2(paste(configurationFileDir,fileWordsToAvoid,sep=""),header = F)

# set working directory
setwd(wipFilesDir)

# create corpus from PDF files
corpusFromTXTFiles <- Corpus(URISource(filesTXT))

# set all text in corpus to lower case
```

```
corpusFromPDFFilesClean <- tm_map(x = corpusFromTXTFiles,
content_transformer(tolower))

# clean corpus from unicode punctuation
corpusFromPDFFilesClean <- tm_map(x = corpusFromPDFFilesClean,
                                  removePunctuation,
                                  preserve_intra_word_contractions = FALSE,
                                  preserve_intra_word_dashes = FALSE,
                                  ucp = TRUE)

# clean corpus from non unicode punctuation to solve the ASCII poisoning
corpusFromPDFFilesClean <- tm_map(x = corpusFromPDFFilesClean,
                                  removePunctuation,
                                  preserve_intra_word_contractions = FALSE,
                                  preserve_intra_word_dashes = FALSE,
                                  ucp = F)


# clean corpus from numbers
corpusFromPDFFilesClean <- tm_map(x = corpusFromPDFFilesClean,
                                  removeNumbers,
                                  ucp = TRUE
                                    )

# clean corpus from econded numbers to solve the ASCII poisoning
corpusFromPDFFilesClean <- tm_map(x = corpusFromPDFFilesClean,
                                  removeNumbers,
                                  ucp = F
)

# stem corpus. ucomment to stem
# corpusFromPDFFilesClean <- tm_map(x = corpusFromPDFFilesClean, stemDocument,
language = languageForAnalysis)

#clean corpus from stopwords
corpusFromPDFFilesClean <- tm_map(x = corpusFromPDFFilesClean,
                                  removeWords,
                                  stopwords(languageForAnalysis)
                                  )

corpusFromPDFFilesClean <- tm_map(x = corpusFromPDFFilesClean,
                                  removeWords,
                                  letters
                                  )

corpusFromPDFFilesClean <- tm_map(x = corpusFromPDFFilesClean,
                                  removeWords,
                                  c(
                                    as.vector(outer(letters, letters, paste,
sep="")),
                                    as.vector(outer(LETTERS, letters, paste,
sep="")),
                                    as.vector(outer(letters, LETTERS, paste,
sep="")),
```

```
                                      as.vector(outer(LETTERS, LETTERS, paste,
sep=""))
                                    )
                               )

corpusFromPDFFilesClean <- tm_map(x = corpusFromPDFFilesClean,
                                  removeWords,
                                  as.vector(t(listOfWordsToRemove))
                                  )



# clean corpus from whitespaces
corpusFromPDFFilesClean <- tm_map(x = corpusFromPDFFilesClean,
stripWhitespace)

# clean from stopwords
corpusFromPDFFilesClean <- tm_map(x = corpusFromPDFFilesClean,
                                  removeWords,
                                  stopwords(languageForAnalysis)
)

# create the term document matrix
opinions.tdm <- TermDocumentMatrix(corpusFromPDFFilesClean,
                                   control =
                                     list(
                                          tokenize=GramTokenizer
                                                       )
                                          )

# generate coloured 1-gram word clouds. plots as PNG image.
matrixOpinions <- as.matrix(opinions.tdm)
matrixFrequencies <- sort(rowSums(matrixOpinions),decreasing=TRUE)
dataframeFrequencies <- data.frame(word =
names(matrixFrequencies),freq=matrixFrequencies)
set.seed(8888) # set the seed for the random number generator, in case it is
needed

nameWordcloudOutput <- paste("wordcloud",
                             minTypeOfNGram,
                             maxTypeOfNGram,
                             format(Sys.time(),
                                    "%Y%m%d%H%M%S"),
                             ".png",
                             sep = "-")

png(nameWordcloudOutput, width=2400,height=2400)
  wordcloud(words = dataframeFrequencies$word,
            freq = dataframeFrequencies$freq,
            min.freq = 1,
            max.words= 48,
            random.order=FALSE,
            rot.per=0.08,
            scale = c(scaleFactor*12,0),
```

```
            colors=colorRampPalette(brewer.pal(9,"Reds"))(100)
  )
dev.off()

# = frequency analysis. plots as PNG image.

# == construct data frame for ggplot
df<-dataframeFrequencies[1:maxNrWordsInFrequencyPlot,]
df$word<-reorder(df$word,df$freq)

# == construct name of output plot of frequencies
nameBarPlotOutput <- paste("barplot",
                           minTypeOfNGram,
                           maxTypeOfNGram,
                           format(Sys.time(),
                                  "%Y%m%d%H%M%S"),
                           ".png",
                           sep = "-")

# == open device, plot, close device
png(nameBarPlotOutput, width=2400,height=2400)
  ggplot(df,aes(word,freq)) +
    geom_bar(stat = "identity") +
    coord_flip() +
    labs(title = "Most Frequent Token / N-Gram") +
    theme(
      text=element_text(
        size=48,
        family="Roboto"
        )
      )
dev.off()

# return to original directory
setwd(originalDir)
```