```
# # # # # # # # # # # # # # # # # # # # #

## This program initializes the environment for the project.
##
## Taking the dataset as input, the data is chunked by its
##  content between the four banks, by its origin, and the
##  indicies are saved for use when examining each bank on
##  an individual basis.



# load functions
source("functions.r")



# required packages
reqPackages <- c("plyr", "tm", "quanteda", "stringr",
                 "ggplot2","SnowballC", "tau","RColorBrewer",
                 "wordcloud", "dplyr", "RWeka")

# load/install packages
dynamicRequire(reqPackages)



# set myStopWords
myStopWords <- unlist(strsplit(readLines("myStopWords.txt"), split=", "))



# list of valence rated words on a scale from -5 to +5
afinnwords <- unlist(strsplit(readLines("AFINN-words.txt"), split=", "))
```

```
# generate tabular data.frame of dataset
dfTable <- read.table('dataset.txt',header=TRUE, sep="|", stringsAsFactors = FALSE)


# generate a data frame with "MediaType" and "FullText" fields from dataset
dfTxtAndSrc <- dfTable[, c("MediaType","FullText")]
rm(dfTable)

#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#
#-#-#-#- Begin Cleansing Dataset -#-#-#-#
#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#
# remove non-ascii characters
dfTxt <- as.data.frame(iconv(dfTxtAndSrc$FullText, "latin1", "ASCII", sub=""))
dfTxtAndSrc$FullText <- dfTxt[, 1]
rm(dfTxt)


# migrate all characters to lowerCase
dfTxtAndSrc$FullText <- tolower(dfTxtAndSrc$FullText)

# corect common spelling errors, internet shorthand, and incidental
#  changes to non-target data during abstraction
dfTxtAndSrc$FullText <- gsub("banke", "ally", dfTxtAndSrc$FullText, perl = TRUE)
dfTxtAndSrc$FullText <- gsub("hobanka ", "how f", dfTxtAndSrc$FullText, perl = TRUE)
dfTxtAndSrc$FullText <- gsub("nebanka ", "new f", dfTxtAndSrc$FullText, perl = TRUE)
dfTxtAndSrc$FullText <- gsub("rebanka ", "rew f", dfTxtAndSrc$FullText, perl = TRUE)
dfTxtAndSrc$FullText <- gsub("viebanka ", "view f", dfTxtAndSrc$FullText, perl = TRUE)
```

```
dfTxtAndSrc$FullText <- gsub("nobanka ", "now f", dfTxtAndSrc$FullText, perl = TRUE)

dfTxtAndSrc$FullText <- gsub("automatibankc anke", "automatically", dfTxtAndSrc$FullText, perl =
TRUE)

dfTxtAndSrc$FullText <- gsub("cancell", "cancel", dfTxtAndSrc$FullText, perl = TRUE)



# remove unnecessary / unwanted characters

dfTxtAndSrc$FullText <- gsub("[[:punct:]]", " ", dfTxtAndSrc$FullText, perl = TRUE)

dfTxtAndSrc$FullText <- gsub("[[:digit:]]", "", dfTxtAndSrc$FullText, perl = TRUE)

dfTxtAndSrc$FullText <- gsub("[[:cntrl:]]", "", dfTxtAndSrc$FullText, perl = TRUE)



# remove custom StopWords

dfTxtAndSrc$FullText <- removeWords(dfTxtAndSrc$FullText, myStopWords)

# combine similar words

dfTxtAndSrc$FullText <- gsub(" credit card ", " card ", dfTxtAndSrc$FullText, perl = TRUE)

dfTxtAndSrc$FullText <- gsub(" debit card ", " card ", dfTxtAndSrc$FullText, perl = TRUE)

dfTxtAndSrc$FullText <- gsub(" debt card ", " card ", dfTxtAndSrc$FullText, perl = TRUE)

dfTxtAndSrc$FullText <- gsub(" cc ", " card ", dfTxtAndSrc$FullText, perl = TRUE)

dfTxtAndSrc$FullText <- gsub(" card card ", " card ", dfTxtAndSrc$FullText, perl = TRUE)

dfTxtAndSrc$FullText <- gsub(" card ", " credit_card ", dfTxtAndSrc$FullText, perl = TRUE)

dfTxtAndSrc$FullText <- gsub(" credit_card credit_card ", " credit_card ", dfTxtAndSrc$FullText, perl =
TRUE)

dfTxtAndSrc$FullText <- gsub(" customer service ", " customer_service ", dfTxtAndSrc$FullText, perl =
TRUE)



# reduce whitespace and remove NULL records left over from previous deletions
```

```
dfTxtAndSrc$FullText <- gsub("\\s+", " ", dfTxtAndSrc$FullText, perl = TRUE)
dfTxtAndSrc$FullText[dfTxtAndSrc$FullText==""] <- NA
dfTxtAndSrc <- na.omit(dfTxtAndSrc)


# remove any duplicate rows
dfTxtAndSrc <- distinct(dfTxtAndSrc)


# create a  textfile of the clean dataset
write.table(dfTxtAndSrc$FullText, file = "datasetCLEAN.txt")
#
#### CLEAN DATASET WRITTEN TO .TXT FILE

### # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # ###

### # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # ###
## ----------                    stop here     ----------###
### # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # ###

### # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # ###


# shorten variable names for readability
remWords <- c("banka","bankb","bankc","bankd", "bank")
df <- dfTxtAndSrc
```

```
###############
b <- "banka" ##<============= for each bank,
###############       change this to ["banka", "bankb", "bankc", "bankd"]
#                   I'm working on code that will let the user decide
#                    which features to load the DFM
#####################################################################

# retrieve texts for specified bank
txtdf <- df[which(sapply(df$FullText,function(x) grepl(b, x))), ]
txts <- as.character(txtdf$FullText)
bankTexts <- as.data.frame(removeWords(txts, remWords), stringsAsFactors = FALSE)
rm(txtdf, txts)




# # # # # # # # # # # # # # # # #
# Quanteda Package Applications #
# # # # # # # # # # # # # # # # #

qCorp <- corpus(bankTexts[, 1])

#####################################################################

##  RUN THESE NEXT SECTIONS ONE AT A TIME

#####################################################################
```

```
# create an unstemmed, 3-skip,  2 to 3-gram Document-Feature-Matrix
#  using the quanteda package
dfm <- dfm(qCorp,
        ngrams = 3,
        skip = 0:1,
        concatenator = " ",
        stem=FALSE)
# this one runs in about 5 seconds



# print, create data.frame, and plot wordcloud
topfeatures(dfm, n = 5000)
dfmNoStemFeat <- as.data.frame(topfeatures(dfm, n = 5000))
pal = brewer.pal(8,"Set1")
plot(dfm,
    max.words=50,
    scale=c(1.5, 0.5),
    random.order=FALSE,
    colors = pal)



#####################################################################




#####################################################################
```

```
# create a stemmed, 3-skip,  2 to 3-gram Document-Feature-Matrix
#  using the quanteda package
dfmStopWords <- dfm(bankTexts[, 1],
                    ignoredFeatures = myStopWords,
                    ngrams = 2:3,
                    skip = 0:1,
                    concatenator = " ",
                    stem=TRUE)
# this one runs in about 140 seconds



# print, create data.frame, and plot wordcloud
topfeatures(dfmStopWords, n = 5000)
dfmStopWordFeat <- topfeatures(dfmStopWords, n = 5000)
pal = brewer.pal(8,"Dark2")
plot(dfm,
    max.words=50,
    scale=c(2, 0.5),
    random.order=TRUE,
    colors = pal)

#####################################################################
```



```
#####################################################################
```

```
# create a stemmed, 2-skip,  2 to 3-gram Document-Feature-Matrix
#  using the quanteda package
dfmAfinn <- dfm(bankTexts[, 1],
                keptFeatures = afinnwords,
                ignoredFeatures = myStopWords,
                ngrams = 2:3,
                skip = 0:1,
                concatenator = " ",
                stem=TRUE)
# this one runs in about 180 seconds



# print, create data.frame, and plot wordcloud
topfeatures(dfmAfinn, n = 5000)
dfmStemAfinnFeat <- topfeatures(dfmStopWords, n = 5000)
pal = brewer.pal(8,"Set3")
plot(dfm,
    max.words=50,
    scale=c(2, 0.5),
    random.order=TRUE,
    colors = pal)

######################################################################




######################################################################
# # # # # # # # # # # # # #
```

```
# tm Package Applications
# # # # # # # # # # # # # #



# generate corpus of documents
docs <- Corpus(DataframeSource(bankTexts))
#docs <- tm_map(docs, stemDocument)  ## stem may not be needed



# generate Document-Term-Matrix and limit entries in
#DTM to words longer than 4 characters and occurring in >10% of documents
dtm <-DocumentTermMatrix(docs, control=list
                    (wordLengths = c(4, 20),
                    bounds = list(global = c(500,5000))))



# generate Term-Document-Matrix and limit entries in
#  TDM to words longer than 4 characters and occurring in >10% of documents
tdm <-TermDocumentMatrix(docs, control=list
                    (wordLengths = c(4, 20),
                    bounds = list(global = c(500,5000))))

freq <- colSums(as.matrix(tdm))
ord <- order(freq, decreasing=TRUE)



findFreqTerms(tdm, lowfreq = 4)
findAssocs(tdm, terms = "freedom", corlimit = 0.3)
```

```
termFreq <- rowSums(as.matrix(tdm))

termFreq <- subset(termFreq, termFreq>=1000)

qplot(names(termFreq), termFreq, main = "Term Frequencies", stat="identity", xlab="Terms") +
coord_flip()


termHistogram = data.frame(term = names(freq), occurrences = freq)

p <- ggplot(subset(termHistogram, freq>100), aes(term, occurrences))

p <- p + geom_bar(stat = "identity")

p <- p + theme(axis.text.x=element_text(angle=45, hjust=1))

p









# tdm <- res$tdm

# freqTable <- res$freqTable

# # Show the top10 words and their frequency

# head(freqTable, 10)

#

#

# # Bar plot of the frequency for the top10

# barplot(freqTable[1:10,]$freq, las = 2,

#         names.arg = freqTable[1:10,]$word,

#         col ="lightblue", main ="Most frequent words",

#         ylab = "Word frequencies")
```

### # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # ###

## ----------                    END      ----------###

### # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # ###