# Ryan Timbrook

## **Applied Data Science**

## **IST687 Intro to Data Science**, Spring 2019

## **Due Date:** 06/11/2019

## **Homework:** 10

### NetID: RTIMBROO

### SUID: 386792749

## #R Code - unexecuted

## Homework Week 10: Text Mining

## -Create our own termDocMatrix

##---- Instructions ----------------------------------------------------------

# Word ranking scale ranking 'AFINN', scale of -5 to 5 (negative to positive)

# This homework task is to adapt the lab that we did in class, to compute the

# score for the MLK speech using the AFINN word list (as opposed to the positive

# and negative word lists).

# Learning Goals for this activity:

# A. Consider how a simple text mining technique can be applied to a variety of kinds of source data

# B. Provide practice in conditioning data to prepare for analysis

# C. Develop skill in the setup, execution, and interpretation of text mining analytics

# D. Increase familiarity with bringing external data sets into R.

#--- Preprocess Steps:----------------------------------------------------------------------

### Clear objects from Memory

rm(list=ls())

### Clear Console:

cat("\014")

### Set Working Directory

setwd("C:\\workspaces\\ms\_datascience\_su\\IST687-IntroDataScience\\R\_workspace\\hw")

#---- Load Required Packages -------------------------------------------------

if(!require("tidytext")){install.packages("tidytext")}

if(!require("readtext")){install.packages("readtext")}

if(!require("OptimalCutpoints")){install.packages("OptimalCutpoints")}

if(!require("tm")){install.packages("tm")}

if(!require("wordcloud")){install.packages("wordcloud")}

if(!require("slam")){install.packages("slam")}

if(!require("ggplot2")){install.packages("ggplot2")}

if(!require("dplyr")) {install.packages("dplyr")}

if(!require("webshot")){install.packages("webshot")}

#---- Global Variable Assignments --------------------------------------------

mlkFileName <- "MLK.txt"

afinnFileName <- "AFINN.txt"

##---- TDM Step 1: Process in the MLK speech ----

#3) Read the text file

#4) Create a term matrix

#5) Create a list of counts for each word

# TDM 1.1: Read in the MLK Speech file

# returns a vector with as many elements as there are lines of text in the file

# grabs each line and is stored as a character string in a vector of character strings

mlkFile <- readLines(file(mlkFileName))

mlkFile <- mlkFile[which(mlkFile != "")] # remove all blank lines

str(mlkFile)

head(mlkFile,10)

View(mlkFile)

# TDM 1.2: Create a term matrix

wordsVec <- VectorSource(mlkFile)

wordsCorpus <- Corpus(wordsVec)

wordsCorpus

View(wordsCorpus)

# make all words in corpus lowercase

wordsCorpus <- tm\_map(wordsCorpus, content\_transformer(tolower))

wordsCorpus <- tm\_map(wordsCorpus, removePunctuation)

wordscorpus <- tm\_map(wordsCorpus, removeNumbers)

wordsCorpus <- tm\_map(wordsCorpus, removeWords, stopwords("english"))

# create a term document matrix

tdm <- TermDocumentMatrix(wordsCorpus)

tdm

View(tdm)

inspect(tdm)

# 1.3: Create a list of counts for each word

# The wordcloud function expects two vectors as input arguments

# - The first a list of the terms

# - The second a list of the frequencies of occurrence of the terms, both must be sorted

m <- as.matrix(tdm)

wordCounts <- rowSums(m)

wordCounts <- sort(wordCounts, decreasing=TRUE)

head(wordCounts,10)

# calculate the total number of words

totalWords <- sum(wordCounts)

totalWords

# vector of all the words

words <- names(wordCounts)

head(words,10)

# build a word cloud

cloudFrame <- data.frame(word=names(wordCounts),freq=wordCounts)

View(cloudFrame)

wordcloud(cloudFrame$word, cloudFrame$freq)

# prety up the wordcloud & save as

wordcloud(names(wordCounts), wordCounts, min.freq = 2, max.words = 50, rot.per = .35, colors = brewer.pal(8,"Dark2"))

#---- AFINN Step 1: Load the data ---------------------------------------------------

# First read in the AFINN word list. Each line is both a word and a score (between -5 and 5)

# Split the line and create two vectors (one for words and one for scores).

# Read Affinity file

afinn <- read.delim(afinnFileName, sep="\t", header = FALSE)

str(afinn)

colnames(afinn) <- c("Word", "Score")

View(afinn)

# join the df match with afinn by "word" col in match and "Word" col in afinn

mergedTable <- merge(cloudFrame, afinn, by.x = "word", by.y = "Word")

str(mergedTable)

View(mergedTable)

#---- Step 2: Compute the overall score ----------------------------------------

# Compute the overall score for the MLK speech using the AFINN word list (as opposed

# to the positive and negative word lists).

overallScore <- sum(mergedTable$freq\*mergedTable$Score)

overallScore

#---- Step 3: Compute the sentiment score --------------------------------------

# Compute the sentiment score for each quarter (25%) of the speech to see how this sentiment

# analysis is the same or different than what was computing with just the positive and negative word files.

getQuarterlySentimentScore <- function(table,quarter){

score <- 0

qRange <- round(nrow(table)/4)

switch(quarter,

"Q1" = {

t <- table[1:qRange,]

score <- sum(t$freq\*t$Score)

},

"Q2" = {

t <- table[(qRange+1):(qRange\*2),]

score <- sum(t$freq\*t$Score)

},

"Q3" = {

t <- table[(qRange\*2+1):(qRange\*3),]

score <- sum(t$freq\*t$Score)

},

"Q4" = {

t <- table[(qRange\*3+1):nrow(table),]

score <- sum(t$freq\*t$Score)

},

stop("Error, invalid quarter value. Acceptiable values are: 'Q1', 'Q2', 'Q3', 'Q4'

")

)

return(score)

}

mlkSplitQuarters <- c('Q1','Q2','Q3','Q4')

mlkSentimentScores <- NULL

for(e in mlkSplitQuarters){

mlkSentimentScores <- c(mlkSentimentScores,getQuarterlySentimentScore(mergedTable,e))

print(paste0(e," = ",getQuarterlySentimentScore(mergedTable,e)))

}

mlkSentimentScores

mlkSentQuartDf <- data.frame(mlkSplitQuarters,mlkSentimentScores)

colnames(mlkSentQuartDf) <- c('Quarter', 'Score')

View(mlkSentQuartDf)

#---- Step 4: Plot the results -------------------------------------------------

# 4 numbers via a bar chart

ggplot(data=mlkSentQuartDf) +

geom\_col(aes(x=Quarter, y=Score)) +

labs(title="MLK Quarterly Sentiment Scores")

ggsave("MLK\_Quarterly\_Sentiment\_Scores.jpg", width = 6, height = 6)

## #R Code – executed

|  |
| --- |
| ### Set Working Directory  > setwd("C:\\workspaces\\ms\_datascience\_su\\IST687-IntroDataScience\\R\_workspace\\hw")  >  >  > #---- Load Required Packages -------------------------------------------------  > if(!require("tidytext")){install.packages("tidytext")}  > if(!require("readtext")){install.packages("readtext")}  > if(!require("OptimalCutpoints")){install.packages("OptimalCutpoints")}  > if(!require("tm")){install.packages("tm")}  > if(!require("wordcloud")){install.packages("wordcloud")}  > if(!require("slam")){install.packages("slam")}  > if(!require("ggplot2")){install.packages("ggplot2")}  > if(!require("dplyr")) {install.packages("dplyr")}  > if(!require("webshot")){install.packages("webshot")}  >  > #---- Global Variable Assignments --------------------------------------------  > mlkFileName <- "MLK.txt"  > afinnFileName <- "AFINN.txt"  >  >  > ##---- TDM Step 1: Process in the MLK speech ----  > #3) Read the text file  > #4) Create a term matrix  > #5) Create a list of counts for each word  >  > # TDM 1.1: Read in the MLK Speech file  > # returns a vector with as many elements as there are lines of text in the file  > # grabs each line and is stored as a character string in a vector of character strings  > mlkFile <- readLines(file(mlkFileName))  > mlkFile <- mlkFile[which(mlkFile != "")] # remove all blank lines  >  > str(mlkFile)  chr [1:29] "I am happy to join with you today in what will go down in history as the greatest demonstration for freedom in "| \_\_truncated\_\_ ...  > head(mlkFile,10)  [1] "I am happy to join with you today in what will go down in history as the greatest demonstration for freedom in the history of our nation."  [2] "Five score years ago, a great American, in whose symbolic shadow we stand today, signed the Emancipation Proclamation. This momentous decree came as a great beacon light of hope to millions of Negro slaves who had been seared in the flames of withering injustice. It came as a joyous daybreak to end the long night of their captivity."  [3] "But one hundred years later, the Negro still is not free. One hundred years later, the life of the Negro is still sadly crippled by the manacles of segregation and the chains of discrimination. One hundred years later, the Negro lives on a lonely island of poverty in the midst of a vast ocean of material prosperity. One hundred years later, the Negro is still languishing in the corners of American society and finds himself an exile in his own land. So we have come here today to dramatize a shameful condition."  [4] "In a sense we have come to our nation's capital to cash a check. When the architects of our republic wrote the magnificent words of the Constitution and the Declaration of Independence, they were signing a promissory note to which every American was to fall heir. This note was a promise that all men, yes, black men as well as white men, would be guaranteed the unalienable rights of life, liberty, and the pursuit of happiness."  [5] "It is obvious today that America has defaulted on this promissory note insofar as her citizens of color are concerned. Instead of honoring this sacred obligation, America has given the Negro people a bad check, a check which has come back marked \"insufficient funds.\" But we refuse to believe that the bank of justice is bankrupt. We refuse to believe that there are insufficient funds in the great vaults of opportunity of this nation. So we have come to cash this check -- a check that will give us upon demand the riches of freedom and the security of justice. We have also come to this hallowed spot to remind America of the fierce urgency of now. This is no time to engage in the luxury of cooling off or to take the tranquilizing drug of gradualism. Now is the time to make real the promises of democracy. Now is the time to rise from the dark and desolate valley of segregation to the sunlit path of racial justice. Now is the time to lift our nation from the quick sands of racial injustice to the solid rock of brotherhood. Now is the time to make justice a reality for all of God's children."  [6] "It would be fatal for the nation to overlook the urgency of the moment. This sweltering summer of the Negro's legitimate discontent will not pass until there is an invigorating autumn of freedom and equality. Nineteen sixty-three is not an end, but a beginning. Those who hope that the Negro needed to blow off steam and will now be content will have a rude awakening if the nation returns to business as usual. There will be neither rest nor tranquility in America until the Negro is granted his citizenship rights. The whirlwinds of revolt will continue to shake the foundations of our nation until the bright day of justice emerges."  [7] "But there is something that I must say to my people who stand on the warm threshold which leads into the palace of justice. In the process of gaining our rightful place we must not be guilty of wrongful deeds. Let us not seek to satisfy our thirst for freedom by drinking from the cup of bitterness and hatred."  [8] "We must forever conduct our struggle on the high plane of dignity and discipline. We must not allow our creative protest to degenerate into physical violence. Again and again we must rise to the majestic heights of meeting physical force with soul force. The marvelous new militancy which has engulfed the Negro community must not lead us to a distrust of all white people, for many of our white brothers, as evidenced by their presence here today, have come to realize that their destiny is tied up with our destiny. They have come to realize that their freedom is inextricably bound to our freedom. We cannot walk alone."  [9] "As we walk, we must make the pledge that we shall always march ahead. We cannot turn back. There are those who are asking the devotees of civil rights, \"When will you be satisfied?\" We can never be satisfied as long as the Negro is the victim of the unspeakable horrors of police brutality. We can never be satisfied, as long as our bodies, heavy with the fatigue of travel, cannot gain lodging in the motels of the highways and the hotels of the cities. We cannot be satisfied as long as the Negro's basic mobility is from a smaller ghetto to a larger one. We can never be satisfied as long as our children are stripped of their selfhood and robbed of their dignity by signs stating \"For Whites Only\". We cannot be satisfied as long as a Negro in Mississippi cannot vote and a Negro in New York believes he has nothing for which to vote. No, no, we are not satisfied, and we will not be satisfied until justice rolls down like waters and righteousness like a mighty stream."  [10] "I am not unmindful that some of you have come here out of great trials and tribulations. Some of you have come fresh from narrow jail cells. Some of you have come from areas where your quest for freedom left you battered by the storms of persecution and staggered by the winds of police brutality. You have been the veterans of creative suffering. Continue to work with the faith that unearned suffering is redemptive."  >  > View(mlkFile)  >  > # TDM 1.2: Create a term matrix  > wordsVec <- VectorSource(mlkFile)  > wordsCorpus <- Corpus(wordsVec)  > wordsCorpus  <<SimpleCorpus>>  Metadata: corpus specific: 1, document level (indexed): 0  Content: documents: 29  > View(wordsCorpus)  >  > # make all words in corpus lowercase  > wordsCorpus <- tm\_map(wordsCorpus, content\_transformer(tolower))  Warning message:  In tm\_map.SimpleCorpus(wordsCorpus, content\_transformer(tolower)) :  transformation drops documents  > wordsCorpus <- tm\_map(wordsCorpus, removePunctuation)  Warning message:  In tm\_map.SimpleCorpus(wordsCorpus, removePunctuation) :  transformation drops documents  > wordscorpus <- tm\_map(wordsCorpus, removeNumbers)  Warning message:  In tm\_map.SimpleCorpus(wordsCorpus, removeNumbers) :  transformation drops documents  > wordsCorpus <- tm\_map(wordsCorpus, removeWords, stopwords("english"))  Warning message:  In tm\_map.SimpleCorpus(wordsCorpus, removeWords, stopwords("english")) :  transformation drops documents  >  > # create a term document matrix  > tdm <- TermDocumentMatrix(wordsCorpus)  > tdm  <<TermDocumentMatrix (terms: 463, documents: 29)>>  Non-/sparse entries: 682/12745  Sparsity : 95%  Maximal term length: 14  Weighting : term frequency (tf)  > View(tdm)  > inspect(tdm)  <<TermDocumentMatrix (terms: 463, documents: 29)>>  Non-/sparse entries: 682/12745  Sparsity : 95%  Maximal term length: 14  Weighting : term frequency (tf)  Sample :  Docs  Terms 18 2 21 29 3 4 5 6 8 9  come 0 0 0 0 1 1 3 0 2 0  day 2 0 1 1 0 0 0 1 0 0  dream 1 0 0 0 0 0 0 0 0 0  freedom 0 0 1 1 0 0 1 1 2 0  let 0 0 0 1 0 0 0 0 0 0  nation 0 0 1 0 0 0 2 3 0 0  negro 0 1 0 1 4 0 1 2 1 3  one 2 0 1 0 4 0 0 0 0 1  ring 0 0 0 2 0 0 0 0 0 0  will 1 0 4 2 0 0 1 5 0 2  >  > # 1.3: Create a list of counts for each word  > # The wordcloud function expects two vectors as input arguments  > # - The first a list of the terms  > # - The second a list of the frequencies of occurrence of the terms, both must be sorted  >  > m <- as.matrix(tdm)  > wordCounts <- rowSums(m)  > wordCounts <- sort(wordCounts, decreasing=TRUE)  > head(wordCounts,10)  will freedom negro one let ring day dream nation come  26 20 13 13 13 12 11 11 10 10  >  > # calculate the total number of words  > totalWords <- sum(wordCounts)  > totalWords  [1] 841  >  > # vector of all the words  > words <- names(wordCounts)  > head(words,10)  [1] "will" "freedom" "negro" "one" "let" "ring" "day" "dream" "nation"  [10] "come"  >  > # build a word cloud  > cloudFrame <- data.frame(word=names(wordCounts),freq=wordCounts)  > View(cloudFrame)  > wordcloud(cloudFrame$word, cloudFrame$freq)  > # prety up the wordcloud & save as  > wordcloud(names(wordCounts), wordCounts, min.freq = 2, max.words = 50, rot.per = .35, colors = brewer.pal(8,"Dark2"))  >  >  > #---- AFINN Step 1: Load the data ---------------------------------------------------  > # First read in the AFINN word list. Each line is both a word and a score (between -5 and 5)  > # Split the line and create two vectors (one for words and one for scores).  >  > # Read Affinity file  > afinn <- read.delim(afinnFileName, sep="\t", header = FALSE)  > str(afinn)  'data.frame': 2477 obs. of 2 variables:  $ V1: Factor w/ 2477 levels "abandon","abandoned",..: 1 2 3 4 5 6 7 8 9 10 ...  $ V2: int -2 -2 -2 -2 -2 -2 -3 -3 -3 -3 ...  > colnames(afinn) <- c("Word", "Score")  > View(afinn)  >  > # join the df match with afinn by "word" col in match and "Word" col in afinn  > mergedTable <- merge(cloudFrame, afinn, by.x = "word", by.y = "Word")  > str(mergedTable)  'data.frame': 61 obs. of 3 variables:  $ word : Factor w/ 463 levels "able","ago","ahead",..: 6 8 19 21 25 36 74 89 91 93 ...  $ freq : num 2 1 1 1 1 1 2 1 1 2 ...  $ Score: int 1 -2 -3 -3 3 1 2 -1 -1 -3 ...  > View(mergedTable)  >  > #---- Step 2: Compute the overall score ----------------------------------------  > # Compute the overall score for the MLK speech using the AFINN word list (as opposed  > # to the positive and negative word lists).  >  > overallScore <- sum(mergedTable$freq\*mergedTable$Score)  > overallScore  [1] 113  >  > #---- Step 3: Compute the sentiment score --------------------------------------  > # Compute the sentiment score for each quarter (25%) of the speech to see how this sentiment  > # analysis is the same or different than what was computing with just the positive and negative word files.  > getQuarterlySentimentScore <- function(table,quarter){  + score <- 0  + qRange <- round(nrow(table)/4)  + switch(quarter,  + "Q1" = {  + t <- table[1:qRange,]  + score <- sum(t$freq\*t$Score)  + },  + "Q2" = {  + t <- table[(qRange+1):(qRange\*2),]  + score <- sum(t$freq\*t$Score)  + },  + "Q3" = {  + t <- table[(qRange\*2+1):(qRange\*3),]  + score <- sum(t$freq\*t$Score)  + },  + "Q4" = {  + t <- table[(qRange\*3+1):nrow(table),]  + score <- sum(t$freq\*t$Score)  + },  + stop("Error, invalid quarter value. Acceptiable values are: 'Q1', 'Q2', 'Q3', 'Q4'  + ")  + )  +  + return(score)  + }  >  > mlkSplitQuarters <- c('Q1','Q2','Q3','Q4')  > mlkSentimentScores <- NULL  > for(e in mlkSplitQuarters){  + mlkSentimentScores <- c(mlkSentimentScores,getQuarterlySentimentScore(mergedTable,e))  + print(paste0(e," = ",getQuarterlySentimentScore(mergedTable,e)))  + }  [1] "Q1 = 5"  [1] "Q2 = 75"  [1] "Q3 = 28"  [1] "Q4 = 5"  > mlkSentimentScores  [1] 5 75 28 5  >  > mlkSentQuartDf <- data.frame(mlkSplitQuarters,mlkSentimentScores)  > colnames(mlkSentQuartDf) <- c('Quarter', 'Score')  > View(mlkSentQuartDf)  >  > #---- Step 4: Plot the results -------------------------------------------------  > # 4 numbers via a bar chart  > ggplot(data=mlkSentQuartDf) +  + geom\_col(aes(x=Quarter, y=Score)) +  + labs(title="MLK Quarterly Sentiment Scores")  > ggsave("MLK\_Quarterly\_Sentiment\_Scores.jpg", width = 6, height = 6) |
|  |
| |  | | --- | | A close up of a white wall  Description automatically generated | |