R Notebook

Title: "IST687 – Text Mining HW"
Name: Sathish Kumar Rajendiran
Week: 10
Date: 06/10/2020

Exercise: Text Mining HW

    Your task for this homework is to adapt the lab that we did in class, to compute the score for the l

```
# import libraries

install.packages(pkgs=c("ggplot2","reshape2","ggeasy","data.table","tm","pdftools","wordcloud"),repos =
```

```
##
## The downloaded binary packages are in
##  /var/folders/_z/ltmjkt4156b37rsk7cgvj7l80000gn/T//RtmpyUXhGt/downloaded_packages
```

```
library(ggplot2)
library(ggeasy)
library(reshape2)
library(data.table)
```

```
##
## Attaching package: 'data.table'

## The following objects are masked from 'package:reshape2':
##
##     dcast, melt
```

```
library(stats)
library(tm)
```

```
## Loading required package: NLP

##
## Attaching package: 'NLP'

## The following object is masked from 'package:ggplot2':
##
##     annotate
```

```r
library(wordcloud)
```

```
## Loading required package: RColorBrewer
```

```r
library(pdftools)
```

```
## Using poppler version 0.73.0
```

```r
# import affin list into dataset
affin <- read.delim("/Users/sathishrajendiran/Documents/AFINN-111.txt", header=FALSE)
colnames(affin) <- c("word","score")

summary(affin)
```

```
##          word          score       
##  abandon   :   1   Min.   :-5.0000  
##  abandoned :   1   1st Qu.:-2.0000  
##  abandons  :   1   Median :-2.0000  
##  abducted  :   1   Mean   :-0.5894  
##  abduction :   1   3rd Qu.: 2.0000  
##  abductions:   1   Max.   : 5.0000  
##  (Other)   :2471
```

```r
str(affin)
```

```
## 'data.frame':    2477 obs. of  2 variables:
##  $ word : Factor w/ 2477 levels "abandon","abandoned",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ score: int  -2 -2 -2 -2 -2 -2 -3 -3 -3 -3 ...
```

```r
dim(affin)
```

```
## [1] 2477    2
```

```r
      # import mlk speech list into dataset
      filename <- "/Users/sathishrajendiran/Documents/mlk_speech.pdf"

      # Read the PDF file
      mlk_speech <- readPDF(control = list(text = "-layout"))(elem = list(uri = filename), language = "
      mlk_speech <- mlk_speech[which(mlk_speech!="")]
      # str(mlk_speech)

      # tail(mlk_speech,20 )

      words.vec <- VectorSource(mlk_speech)
      words.corpus <- Corpus(words.vec)
      # str(words.vec)
      # str(words.corpus)

      # tm transformation | to lowercase, remove puntuation, numbers and stop words
      words.corpus <- tm_map(words.corpus,content_transformer(tolower))
```

```
## Warning in tm_map.SimpleCorpus(words.corpus, content_transformer(tolower)):
## transformation drops documents
```

```
    words.corpus <- tm_map(words.corpus,removePunctuation)
```

```
## Warning in tm_map.SimpleCorpus(words.corpus, removePunctuation): transformation
## drops documents
```

```
    words.corpus <- tm_map(words.corpus,removeNumbers)
```

```
## Warning in tm_map.SimpleCorpus(words.corpus, removeNumbers): transformation
## drops documents
```

```
    words.corpus <- tm_map(words.corpus,removeWords,stopwords("english"))
```

```
## Warning in tm_map.SimpleCorpus(words.corpus, removeWords, stopwords("english")):
## transformation drops documents
```

```
    # create term document matrix of the words corpus
    tdm <- TermDocumentMatrix(words.corpus)
    str(tdm)
```

```
## List of 6
##  $ i       : int [1:546] 1 2 3 4 5 6 7 8 9 10 ...
##  $ j       : int [1:546] 1 1 1 1 1 1 1 1 1 1 ...
##  $ v       : num [1:546] 8 1 1 3 2 1 1 1 1 4 ...
##  $ nrow    : int 546
##  $ ncol    : int 2
##  $ dimnames:List of 2
##   ..$ Terms: chr [1:546] "able" "ago" "ahead" "alabama" ...
##   ..$ Docs : chr [1:2] "1" "2"
##  - attr(*, "class")= chr [1:2] "TermDocumentMatrix" "simple_triplet_matrix"
##  - attr(*, "weighting")= chr [1:2] "term frequency" "tf"
```

```
    inspect(tdm[1:50,1:2])
```

```
## <<TermDocumentMatrix (terms: 50, documents: 2)>>
## Non-/sparse entries: 50/50
## Sparsity           : 50%
## Maximal term length: 17
## Weighting          : term frequency (tf)
## Sample             :
##          Docs
## Terms     1 2
##   able    8 0
##   ago     1 0
##   ahead   1 0
##   alabama 3 0
##   allow   2 0
##   america 4 0
```

```
##   american 3 0
##   back     8 0
##   black    3 0
##   boys     2 0
```

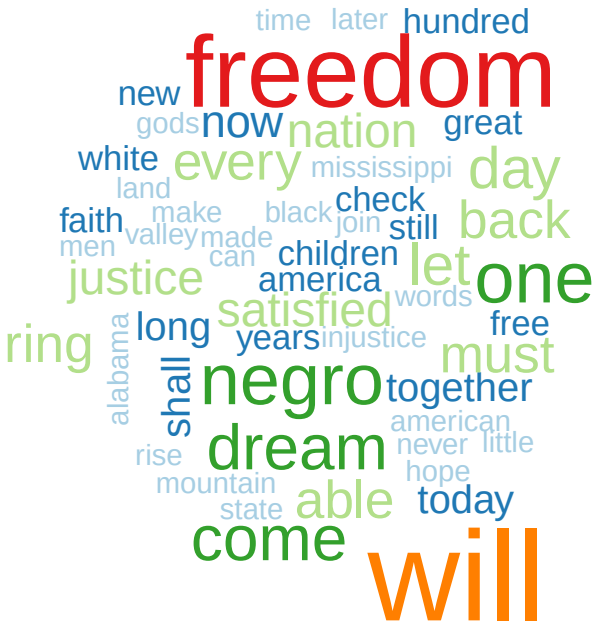
```
# create matrix
m <- as.matrix(tdm)
# str(m)

# compute word counts
wordCounts <- rowSums(m)
wordCounts <- sort(wordCounts,decreasing = TRUE)


head(wordCounts)
```

```
##    will freedom   negro     one    come   dream
##      24      18      12      12      10      10
```

```
# create word cloud
wordcloud(names(wordCounts),wordCounts,rot.per = .08,colors = brewer.pal(8,"Paired"))
```




```
#Compute metrics
totalwords <- sum(wordCounts)
words <- names(wordCounts)
```

```
    #find matching AFFIN words from MLK Speech and return 0 for non matching words
    speech_affin <- match(words,affin$word,nomatch = 0)

    # speech_affin
    matchCounts <- wordCounts[which(speech_affin != 0)]
    # matchCounts

    # create a data frame for mlk speech
    speech_df <- data.frame(names(matchCounts),matchCounts,row.names = c(1:length(matchCounts)))
    colnames(speech_df) <- c("word","counts")

    # speech_df[1:10,]
    # affin[1:10,]

    # merge speech and affin dataframe to include the score

    affin_speech <- merge(speech_df,affin, by="word")
    affin_speech[1:10,]
```

```
##             word counts score
## 1          allow      2     1
## 2            bad      1    -3
## 3       bankrupt      1    -3
## 4      beautiful      1     3
## 5       creative      2     2
## 6         demand      1    -1
## 7  demonstration      1    -1
## 8        despair      1    -3
## 9           died      1    -3
## 10      distrust      1    -3
```

```
    # calculate overall score
    final.score <- sum(affin_speech$counts * affin_speech$score)/totalwords
    # final.score

    cat("\n Overall score for the MLK speech using the AFINN word list is: ",round(final.score*100,2)
```

```
##
##  Overall score for the MLK speech using the AFINN word list is: 10.48%
```

```
  # import mlk speech list into dataset
  filename <- "/Users/sathishrajendiran/Documents/mlk_speech.pdf"

  # Read the PDF file into a dataframe
  mlk_speech <- readPDF(control = list(text = "-layout"))(elem = list(uri = filename), language = "en
  mlk_speech <- mlk_speech[which(mlk_speech!="")]
  df1 <- data.frame(strwrap(mlk_speech[[1]]),stringsAsFactors=FALSE)
  # View(df1)

  corpusFunction <- function(i){

      #split the data into 4 quarters
```

```
nrows <- nrow(df1)
cutPoint_Start <- floor(nrows * (i-1)/4) +1
cutPoint_End <- floor(nrows * i/4)
# df1[cutPoint_Start:cutPoint_End,]

#Create words corpus
dfCorpus = Corpus(VectorSource(df1[cutPoint_Start:cutPoint_End,]))
# inspect(dfCorpus)

# tm transformation | to lowercase, remove puntuation, numbers and stop words
dfCorpus <- tm_map(dfCorpus,content_transformer(tolower))
dfCorpus <- tm_map(dfCorpus,removePunctuation)
dfCorpus <- tm_map(dfCorpus,removeNumbers)
dfCorpus <- tm_map(dfCorpus,stripWhitespace)
dfCorpus <- tm_map(dfCorpus,removeWords,stopwords("english"))

# create term document matrix of the words corpus
dftdm <- TermDocumentMatrix(dfCorpus)
# inspect(dftdm[1:50,1:2])

# create matrix
dfm <- as.matrix(dftdm)

# compute word counts
dfwordCounts <- rowSums(dfm)
dfwordCounts <- sort(dfwordCounts,decreasing = TRUE)

# head(dfwordCounts)

# create word cloud
dfwordcloud <- wordcloud(names(dfwordCounts),dfwordCounts,rot.per = .08,colors = brewer.pal(8,"
cat("\n cutPoint_start:",cutPoint_Start,"\n cutPoint_end:",cutPoint_End,"\n")

dfwordcloud

#Compute metrics
dftotalwords <- sum(dfwordCounts)
dfwords <- names(dfwordCounts)

#find matching AFFIN words from MLK Speech and return 0 for non matching words
speech_affin <- match(dfwords,affin$word,nomatch = 0)

# speech_affin
dfmatchCounts <- dfwordCounts[which(speech_affin != 0)]
# matchCounts

# create a data frame for mlk speech
speech_df <- data.frame(names(dfmatchCounts),dfmatchCounts,row.names = c(1:length(dfmatchCounts
colnames(speech_df) <- c("word","counts")

# speech_df[1:10,]
# affin[1:10,]
```

```
        # merge speech and affin dataframe to include the score

        affin_speech <- merge(speech_df,affin, by="word")
        # affin_speech[1:10,]

        # calculate overall score
        final.score <- round((sum(affin_speech$counts * affin_speech$score)/totalwords)*100,2)
        # final.score

        cat("\n Overall score for the MLK speech using the AFINN word list is: ",final.score,"%",sep =

        return(final.score)
    }

    # Process first quarter
    q1 <- corpusFunction(1)
```

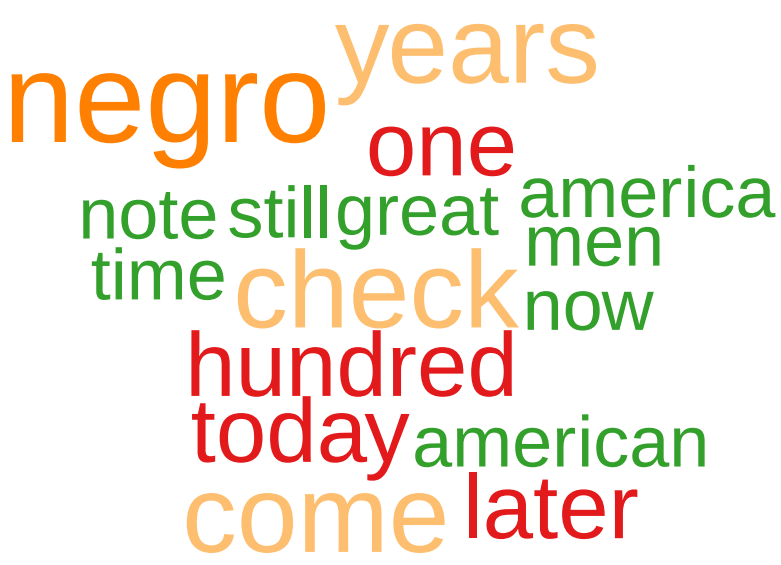
```
## Warning in tm_map.SimpleCorpus(dfCorpus, content_transformer(tolower)):
## transformation drops documents

## Warning in tm_map.SimpleCorpus(dfCorpus, removePunctuation): transformation
## drops documents

## Warning in tm_map.SimpleCorpus(dfCorpus, removeNumbers): transformation drops
## documents

## Warning in tm_map.SimpleCorpus(dfCorpus, stripWhitespace): transformation drops
## documents

## Warning in tm_map.SimpleCorpus(dfCorpus, removeWords, stopwords("english")):
## transformation drops documents
```




```
##
##  cutPoint_start: 1
##  cutPoint_end: 33
##
##  Overall score for the MLK speech using the AFINN word list is: 2.36%
```

```
    # Process first quarter
    q2 <- corpusFunction(2)
```

```
## Warning in tm_map.SimpleCorpus(dfCorpus, content_transformer(tolower)):
## transformation drops documents
```

```
## Warning in tm_map.SimpleCorpus(dfCorpus, removePunctuation): transformation
## drops documents
```

```
## Warning in tm_map.SimpleCorpus(dfCorpus, removeNumbers): transformation drops
## documents
```

```
## Warning in tm_map.SimpleCorpus(dfCorpus, stripWhitespace): transformation drops
## documents
```

```
## Warning in tm_map.SimpleCorpus(dfCorpus, removeWords, stopwords("english")):
## transformation drops documents
```


```
##
##  cutPoint_start: 34
##  cutPoint_end: 66
##
##  Overall score for the MLK speech using the AFINN word list is: 2%
```

```
    # Process first quarter
    q3 <- corpusFunction(3)
```

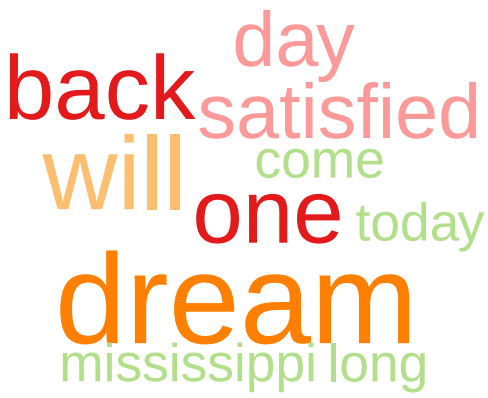
```
## Warning in tm_map.SimpleCorpus(dfCorpus, content_transformer(tolower)):
## transformation drops documents

## Warning in tm_map.SimpleCorpus(dfCorpus, removePunctuation): transformation
## drops documents

## Warning in tm_map.SimpleCorpus(dfCorpus, removeNumbers): transformation drops
## documents

## Warning in tm_map.SimpleCorpus(dfCorpus, stripWhitespace): transformation drops
## documents

## Warning in tm_map.SimpleCorpus(dfCorpus, removeWords, stopwords("english")):
## transformation drops documents
```




```
##
##  cutPoint_start: 67
##  cutPoint_end: 99
##
##  Overall score for the MLK speech using the AFINN word list is: 3.06%
```

```
    # Process first quarter
    q4 <- corpusFunction(4)
```

```
## Warning in tm_map.SimpleCorpus(dfCorpus, content_transformer(tolower)):
## transformation drops documents

## Warning in tm_map.SimpleCorpus(dfCorpus, removePunctuation): transformation
## drops documents

## Warning in tm_map.SimpleCorpus(dfCorpus, removeNumbers): transformation drops
## documents

## Warning in tm_map.SimpleCorpus(dfCorpus, stripWhitespace): transformation drops
## documents

## Warning in tm_map.SimpleCorpus(dfCorpus, removeWords, stopwords("english")):
## transformation drops documents
```

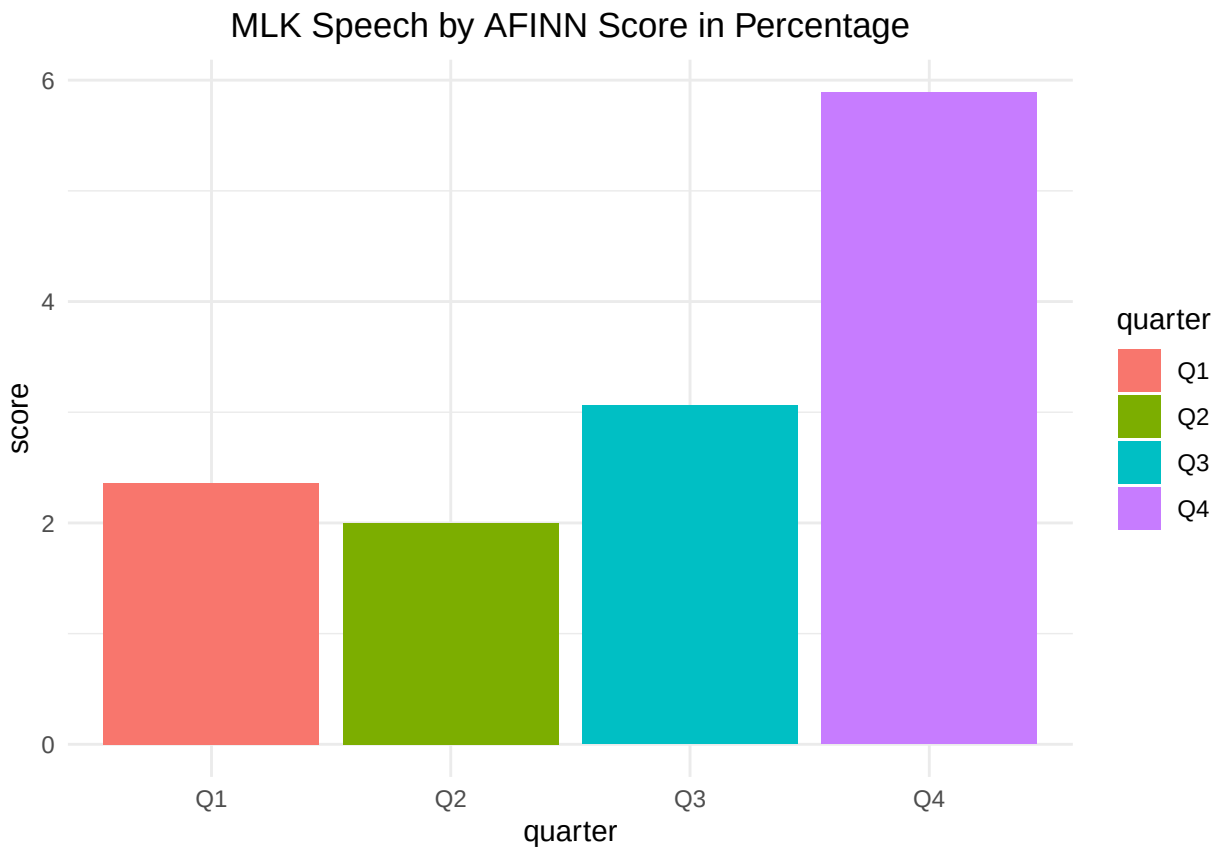

```
##
##  cutPoint_start: 100
##  cutPoint_end: 132
##
##  Overall score for the MLK speech using the AFINN word list is: 5.89%
```

```
  #create a dataframe combining all 4 scores from Q1, Q2, Q3 and Q4

  df_scores <- data.frame(quarter=c("Q1","Q2","Q3","Q4"),score=c(q1,q2,q3,q4))

  theme <-theme(plot.title = element_text(hjust = 0.5),axis.title = element_text())
  dfplot <- ggplot(df_scores,aes(x=score, y=quarter, fill=quarter)) + geom_bar(stat="identity")+theme_mi
  dfplot <- dfplot + coord_flip()+ ggtitle("MLK Speech by AFINN Score in Percentage")+ theme

  dfplot
```


MLK Speech by AFINN Score in Percentage
6
4
2
0
score
Q1
Q2
Q3
Q4
quarter
quarter
Q1
Q2
Q3
Q4