HW10

Thulasiram Ruppa Krishnan

March 25, 2019

```
library(tm)
```

```
## Loading required package: NLP
```

```
library(wordcloud)
```

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

```
library(ggplot2)
```

```
##
## Attaching package: 'ggplot2'
```

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

```
# Clear objects
rm(list=ls())


pos <- "C:/Users/rkrishnan/Documents/01 Personal/MS/IST 687/opinion-lexicon-English/positive-wor
ds.txt"
neg <- "C:/Users/rkrishnan/Documents/01 Personal/MS/IST 687/opinion-lexicon-English/negative-wor
ds.txt"

# read the files
p <- scan(pos,character(0),sep = "\n")
n <- scan(neg,character(0),sep = "\n")

#remove the 1st 34 lines (Header Info)

p <- p[-1:-34]
n <- n[-1:-34]

head(p,10)
```

```
##  [1] "accessable"    "accessible"    "acclaim"       "acclaimed"
##  [5] "acclamation"   "accolade"      "accolades"     "accommodative"
##  [9] "accomodative"  "accomplish"
```

```
head(n,10)
```

```
##  [1] "abominable"  "abominably"  "abominate"   "abomination" "abort"
##  [6] "aborted"     "aborts"      "abrade"      "abrasive"    "abrupt"
```

```
sbaFile <- "http://www.historyplace.com/speeches/anthony.htm"

sba <- readLines(sbaFile)

str(sba)
```

```
##  chr [1:145] "<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 3.2//EN\">" ...
```

```
# Text Transformation

words.vec <-VectorSource(sba)

words.corpus <-Corpus(words.vec)

words.corpus
```

```
## <<SimpleCorpus>>
## Metadata:  corpus specific: 1, document level (indexed): 0
## Content:  documents: 145
```

```
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
```

```
tdm <-TermDocumentMatrix(words.corpus)
tdm
```

```
## <<TermDocumentMatrix (terms: 388, documents: 145)>>
## Non-/sparse entries: 561/55699
## Sparsity           : 99%
## Maximal term length: 63
## Weighting          : term frequency (tf)
```

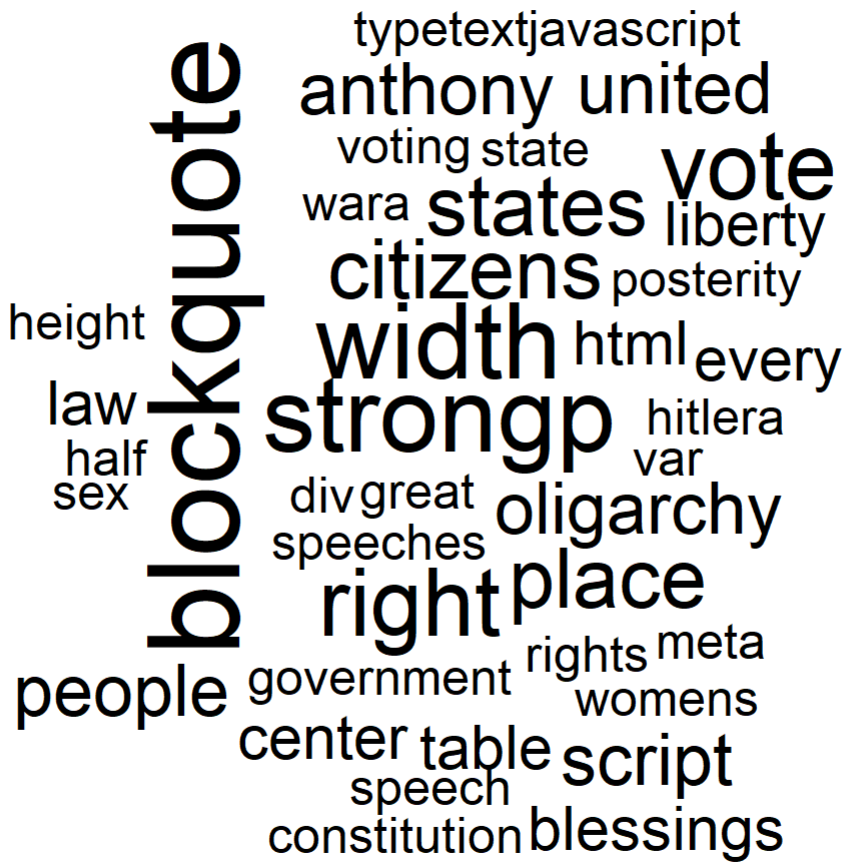
```
m <-as.matrix(tdm)
wordCounts <- rowSums(m)
wordCounts <- sort(wordCounts,decreasing = TRUE)
head(wordCounts)
```

```
##      women blockquote    history      width    strongp      right
##         10         10          8          8          8          7
```

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

```
## Warning in wordcloud(cloudFrame$word, cloudFrame$freq): history could not
## be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(cloudFrame$word, cloudFrame$freq): women could not be
## fit on page. It will not be plotted.
```




```
wordcloud(names(wordCounts),wordCounts,min.freq = 2,max.words = 50,rot.per = 0.35,colors = brewe
r.pal(8,"Dark2"))
```

```
## Warning in wordcloud(names(wordCounts), wordCounts, min.freq = 2, max.words
## = 50, : blockquote could not be fit on page. It will not be plotted.
```


```
#calculate the total number of words
totalwords <- sum(wordCounts)

#have a vector that just has all the words
words <-names(wordCounts)
matched <- match(words,p,nomatch=0)
head(matched,10)
```

```
##  [1]    0    0    0    0    0 1528    0    0    0    0
```

```
matched[6]
```

```
## [1] 1528
```

```
p[1528]
```

```
## [1] "right"
```

```
words[6]
```

```
## [1] "right"
```

```
mCounts <-wordCounts[which(matched !=0)]
length(mCounts)
```

```
## [1] 16
```

```
mWords <- names(mCounts)
nPos <- sum(mCounts)
nPos
```

```
## [1] 29
```

```
matched <- match(words,n,nomatch=0)
head(matched,100)
```

```
##   [1]    0    0    0    0    0    0    0    0    0    0    0    0    0    0
##  [15]    0    0    0    0    0    0    0    0    0    0    0    0    0    0
##  [29]    0    0    0    0    0    0    0    0    0    0    0    0    0    0
##  [43]    0    0    0    0    0    0    0    0    0    0    0    0    0  750
##  [57]    0    0    0    0    0    0    0    0    0    0    0    0    0    0
##  [71]    0 1977    0    0    0    0    0    0    0    0    0    0    0    0
##  [85]    0    0    0    0    0    0    0    0    0    0    0    0    0    0
##  [99]    0    0
```

```
matched[56]
```

```
## [1] 750
```

```
n[750]
```

```
## [1] "crime"
```

```
words[56]
```

```
## [1] "crime"
```

```
nCounts <-wordCounts[which(matched !=0)]
length(nCounts)
```

```
## [1] 15
```

```
nWords <- names(nCounts)
nNeg <- sum(nCounts)
nNeg
```

```
## [1] 17
```

```
# calculate the % of words that are positive and negative
totalWords <-length(words)

ratioPos <-nPos/totalWords
ratioPos
```

```
## [1] 0.07474227
```

```
ratioNeg <-nNeg/totalWords
ratioNeg
```

```
## [1] 0.04381443
```

```
library(tidytext)
```

```
## Warning: package 'tidytext' was built under R version 3.5.3
```

```
sentiments
```

```
## # A tibble: 27,314 x 4
##    word        sentiment lexicon score
##    <chr>       <chr>     <chr>   <int>
##  1 abacus      trust     nrc        NA
##  2 abandon     fear      nrc        NA
##  3 abandon     negative  nrc        NA
##  4 abandon     sadness   nrc        NA
##  5 abandoned   anger     nrc        NA
##  6 abandoned   fear      nrc        NA
##  7 abandoned   negative  nrc        NA
##  8 abandoned   sadness   nrc        NA
##  9 abandonment anger     nrc        NA
## 10 abandonment fear      nrc        NA
## # ... with 27,304 more rows
```

```
affin <- get_sentiments("afinn")
get_sentiments("bing")
```

```
## # A tibble: 6,788 x 2
##    word        sentiment
##    <chr>       <chr>
##  1 2-faced     negative
##  2 2-faces     negative
##  3 a+          positive
##  4 abnormal    negative
##  5 abolish     negative
##  6 abominable  negative
##  7 abominably  negative
##  8 abominate   negative
##  9 abomination negative
## 10 abort       negative
## # ... with 6,778 more rows
```

```
get_sentiments("nrc")
```

```
## # A tibble: 13,901 x 2
##    word        sentiment
##    <chr>       <chr>
##  1 abacus      trust
##  2 abandon     fear
##  3 abandon     negative
##  4 abandon     sadness
##  5 abandoned   anger
##  6 abandoned   fear
##  7 abandoned   negative
##  8 abandoned   sadness
##  9 abandonment anger
## 10 abandonment fear
## # ... with 13,891 more rows
```

```
# compute the overall score using AFFIN word list

matched <-match(words,affin$word,nomatch=0)
matched
```

```
##   [1]    0    0    0    0    0    0    0    0    0    0    0    0 2323    0
##  [15]    0    0    0    0    0    0    0    0 1104    0    0    0    0    0
##  [29]    0    0    0    0    0    0    0    0    0    0    0    0    0    0
##  [43]    0    0    0    0    0    0    0    0    0    0    0    0    0  523
##  [57]    0    0    0    0    0 1981    0    0    0    0    0    0    0    0
##  [71]    0    0    0    0    0 2398    0    0    0    0    0    0    0    0
##  [85]    0    0 2428    0    0    0    0    0    0    0    0    0    0    0
##  [99]    0    0    0    0 2279    0    0    0    0    0    0    0    0    0
## [113]    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## [127]    0    0    0    0    0 1427    0    0  172    0 1243    0 1683 1850
## [141]    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## [155]    0    0    0    0    0    0  446    0    0    0    0    0    0  619
## [169]    0    0    0    0    0    0    0    0    0    0 1394 1689    0    0
## [183]    0    0    0 1780    0    0    0    0    0    0    0    0    0    0
## [197]    0    0    0  613    0    0    0    0    0    0    0    0    0    0
## [211]    0    0    0    0    0    0    0    0    0    0    0    0  478    0
## [225]    0    0    0    0    0    0    0    0    0    0    0 1732 1917 2413
## [239]    0    0 1238    0    0    0    0    0    0    0    0    0    0    0
## [253]    0    0    0    0    0    0    0    0  694    0    0    0    0    0
## [267] 1842    0    0    0    0    0    0  866    0    0    0    0    0    0
## [281]    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## [295]    0    0    0    0    0    0    0    0    0    0  568    0    0    0
## [309]    0    0    0  835    0    0    0    0    0    0    0 1833    0    0
## [323]    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## [337]    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## [351]    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## [365]    0    0    0  260    0    0    0    0    0    0    0    0    0    0
## [379]    0    0    0    0    0    0    0    0    0    0
```

```
matched[13]
```

```
## [1] 2323
```

```
words[13]
```

```
## [1] "united"
```

```
affin$word[2323]
```

```
## [1] "united"
```

```
affin$score[2323]
```

```
## [1] 1
```

```
wordCounts[which(matched !=0)]
```

```
##    united     great     crime    secure       war       win      true
##         5         3         2         2         2         1         1
##     legal    arrest   illegal       pay   refused committed      deny
##         1         1         1         1         1         1         1
##   justice   perfect   promote    denied   consent      poor      rich
##         1         1         1         1         1         1         1
##    wealth  ignorant   discord rebellion  entitled     death  endorsed
##         1         1         1         1         1         1         1
##  ratified      best
##         1         1
```

```
affin$word[matched[which(matched !=0)]]
```

```
##  [1] "united"    "great"     "crime"     "secure"    "war"
##  [6] "win"       "true"      "legal"     "arrest"    "illegal"
## [11] "pay"       "refused"   "committed" "deny"      "justice"
## [16] "perfect"   "promote"   "denied"    "consent"   "poor"
## [21] "rich"      "wealth"    "ignorant"  "discord"   "rebellion"
## [26] "entitled"  "death"     "endorsed"  "ratified"  "best"
```

```
affin$score[matched[which(matched !=0)]]
```

```
##  [1]  1  3 -3  2 -2  4  2  1 -2 -3 -1 -2  1 -2  2  3  1 -2  2 -2  2  3 -2
## [24] -2 -2  1 -2  2  2  3
```

```
mScore <- affin$score[matched[which(matched !=0)]]

pScore <- sum(ifelse(mScore >0, mScore, 0))
nScore <- abs(sum(ifelse(mScore <0, mScore, 0)))
totalScore <- sum(abs(mScore))

# Overall Score
totalScore
```

```
## [1] 62
```

```
pScore
```

```
## [1] 35
```

```
nScore
```

```
## [1] 27
```

```
#ratio of  postive and negative  Score

ratioPosScore <-pScore/totalScore
ratioNegScore <-nScore/totalScore

ratioPosScore
```

```
## [1] 0.5645161
```

```
ratioNegScore
```

```
## [1] 0.4354839
```

```
fnGetSentimentScore <- function(words,i){

matched <-match(words,affin$word,nomatch=0)
#print(paste("Matched :" ,matched))


wordCounts[which(matched !=0)]
affin$word[matched[which(matched !=0)]]
affin$score[matched[which(matched !=0)]]
mScore <- affin$score[matched[which(matched !=0)]]

pScore.m[i] <<- sum(ifelse(mScore >0, mScore, 0))
nScore.m[i] <<- abs(sum(ifelse(mScore <0, mScore, 0)))
totalScore.m[i] <<- sum(abs(mScore))

print(paste(i," - 25% of the speech" ))
print(paste("_____________________________________" ))
# Overall Score
print(paste("Total Score :" ,totalScore.m[i]))
print(paste("Positive Score :" ,pScore.m[i]))
print(paste("Negative Score :", nScore.m[i]))


#ratio of  postive and negative  Score

ratioPosScore.m[i] <<-pScore.m[i]/totalScore.m[i]
ratioNegScore.m[i] <<-nScore.m[i]/totalScore.m[i]


print(paste("Positive Score ratio :" ,ratioPosScore.m[i]))
print(paste("Negative Score ratio :" ,ratioNegScore.m[i]))

print(paste("_____________________________________" ))

}



fnGetSentimentScore2 <- function(words,i){

matched <-match(words,affin$word,nomatch=0)
#print(paste("Matched :" ,matched))


wordCounts[which(matched !=0)]
affin$word[matched[which(matched !=0)]]
affin$score[matched[which(matched !=0)]]
mScore <- affin$score[matched[which(matched !=0)]]

pScore <- sum(ifelse(mScore >0, mScore, 0))
nScore <- abs(sum(ifelse(mScore <0, mScore, 0)))
totalScore <- sum(abs(mScore))
```

```
print(paste(i," - 25% of the speech" ))
# Overall Score
totalScore
print(paste("Total Score :" ,totalScore))
pScore
print(paste("Positive Score :" ,pScore))
nScore
print(paste("Negative Score :", nScore))


#ratio of  postive and negative  Score

ratioPosScore <-pScore/totalScore
ratioNegScore <-nScore/totalScore

ratioPosScore
print(paste("Positive Score ratio :" ,ratioPosScore))
ratioNegScore
print(paste("Negative Score ratio :" ,ratioNegScore))



}
```

```
# remove the sorting as we are computing 25% of the speech words on an incremental basis
wordCounts <- rowSums(m)

totalWords <-length(wordCounts)
qtr_Words <- totalWords/4

totalScore.m <-matrix(nrow = 4, ncol = 1)
pScore.m <-matrix(nrow = 4, ncol = 1)
nScore.m <-matrix(nrow = 4, ncol = 1)
ratioPosScore.m <-matrix(nrow = 4, ncol = 1)
ratioNegScore.m <-matrix(nrow = 4, ncol = 1)


First.quarter.wordCounts <- wordCounts[1:94]
First.quarter.words <-names(First.quarter.wordCounts)

Second.quarter.wordCounts <- wordCounts[95:188]
Second.quarter.words <-names(Second.quarter.wordCounts)

Third.quarter.wordCounts <- wordCounts[189:282]
Third.quarter.words <-names(Third.quarter.wordCounts)

Fourth.quarter.wordCounts <- wordCounts[283:totalWords]
Fourth.quarter.words <-names(Fourth.quarter.wordCounts)
```

```
fnGetSentimentScore(First.quarter.words,1)
```

```
## [1] "1  - 25% of the speech"
## [1] "_____________________________________"
## [1] "Total Score : 13"
## [1] "Positive Score : 11"
## [1] "Negative Score : 2"
## [1] "Positive Score ratio : 0.846153846153846"
## [1] "Negative Score ratio : 0.153846153846154"
## [1] "_____________________________________"
```

```
fnGetSentimentScore(Second.quarter.words,2)
```

```
## [1] "2  - 25% of the speech"
## [1] "_____________________________________"
## [1] "Total Score : 22"
## [1] "Positive Score : 9"
## [1] "Negative Score : 13"
## [1] "Positive Score ratio : 0.409090909090909"
## [1] "Negative Score ratio : 0.590909090909091"
## [1] "_____________________________________"
```

```
fnGetSentimentScore(Third.quarter.words,3)
```

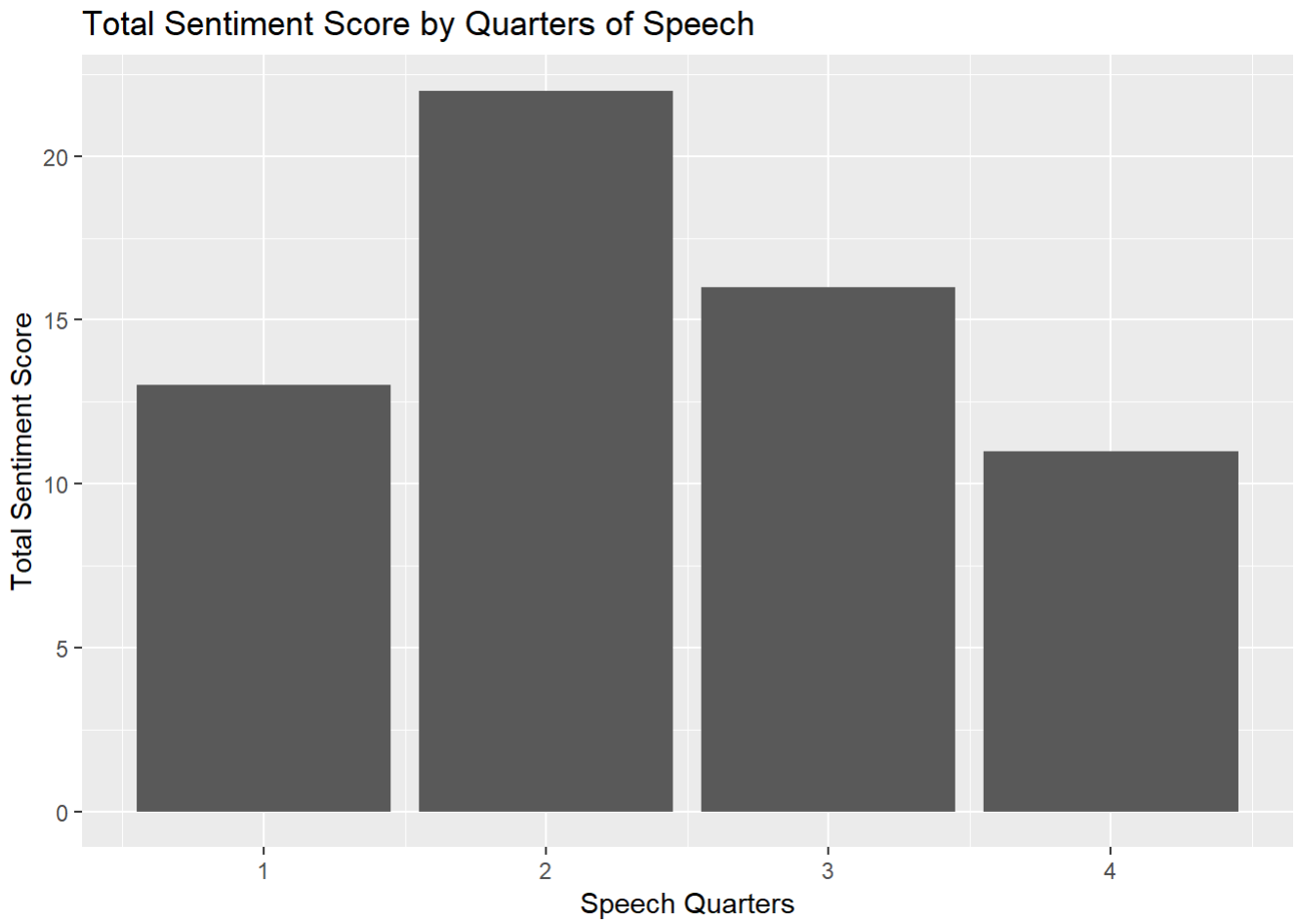
```
## [1] "3  - 25% of the speech"
## [1] "_____________________________________"
## [1] "Total Score : 16"
## [1] "Positive Score : 8"
## [1] "Negative Score : 8"
## [1] "Positive Score ratio : 0.5"
## [1] "Negative Score ratio : 0.5"
## [1] "_____________________________________"
```

```
fnGetSentimentScore(Fourth.quarter.words,4)
```

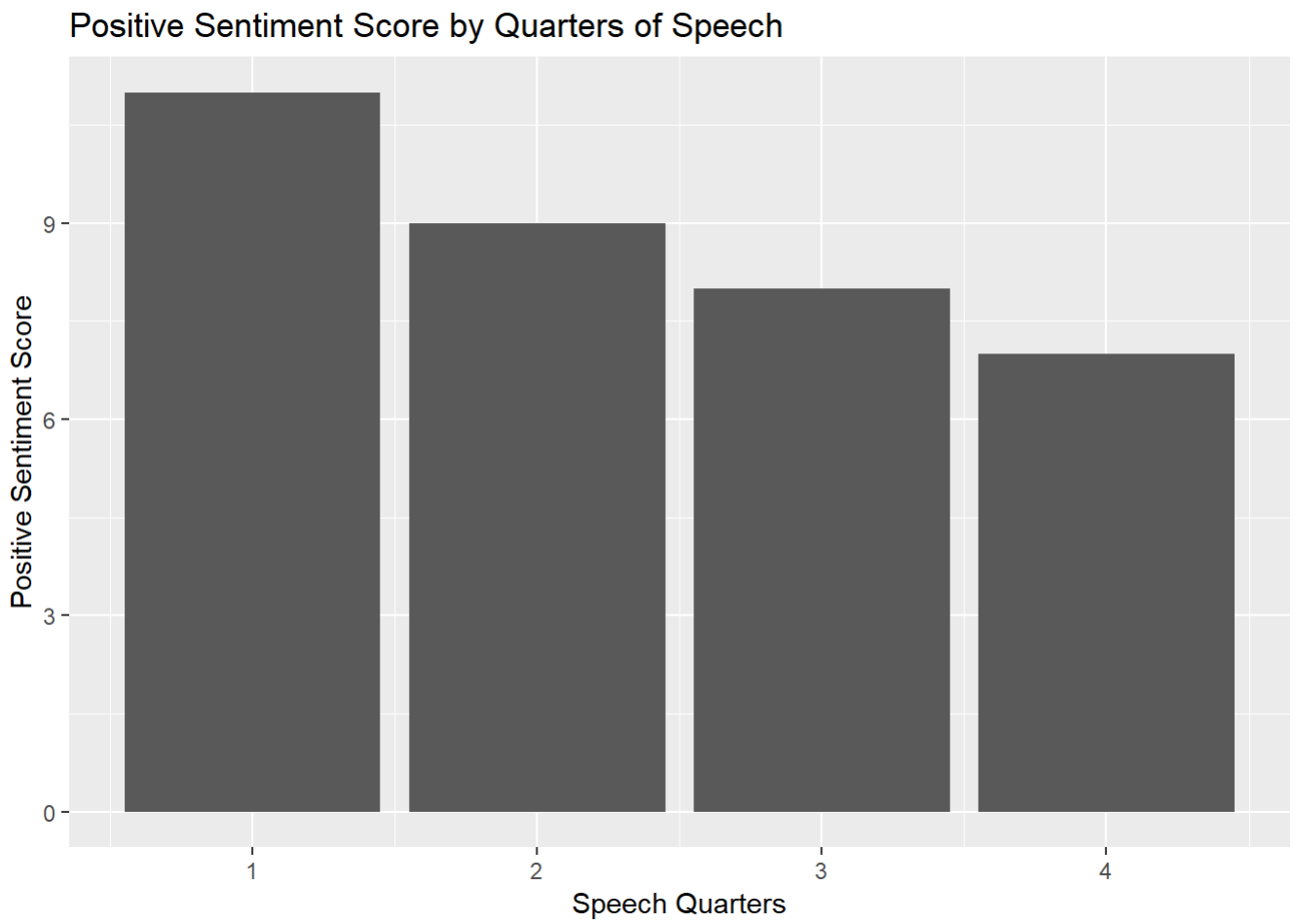
```
## [1] "4  - 25% of the speech"
## [1] "_____________________________________"
## [1] "Total Score : 11"
## [1] "Positive Score : 7"
## [1] "Negative Score : 4"
## [1] "Positive Score ratio : 0.636363636363636"
## [1] "Negative Score ratio : 0.363636363636364"
## [1] "_____________________________________"
```

```
#fnGetSentimentScore2(First.quarter.words,1)
#fnGetSentimentScore2(Second.quarter.words,2)
#fnGetSentimentScore2(Third.quarter.words,3)
#fnGetSentimentScore2(Fourth.quarter.words,4)
```

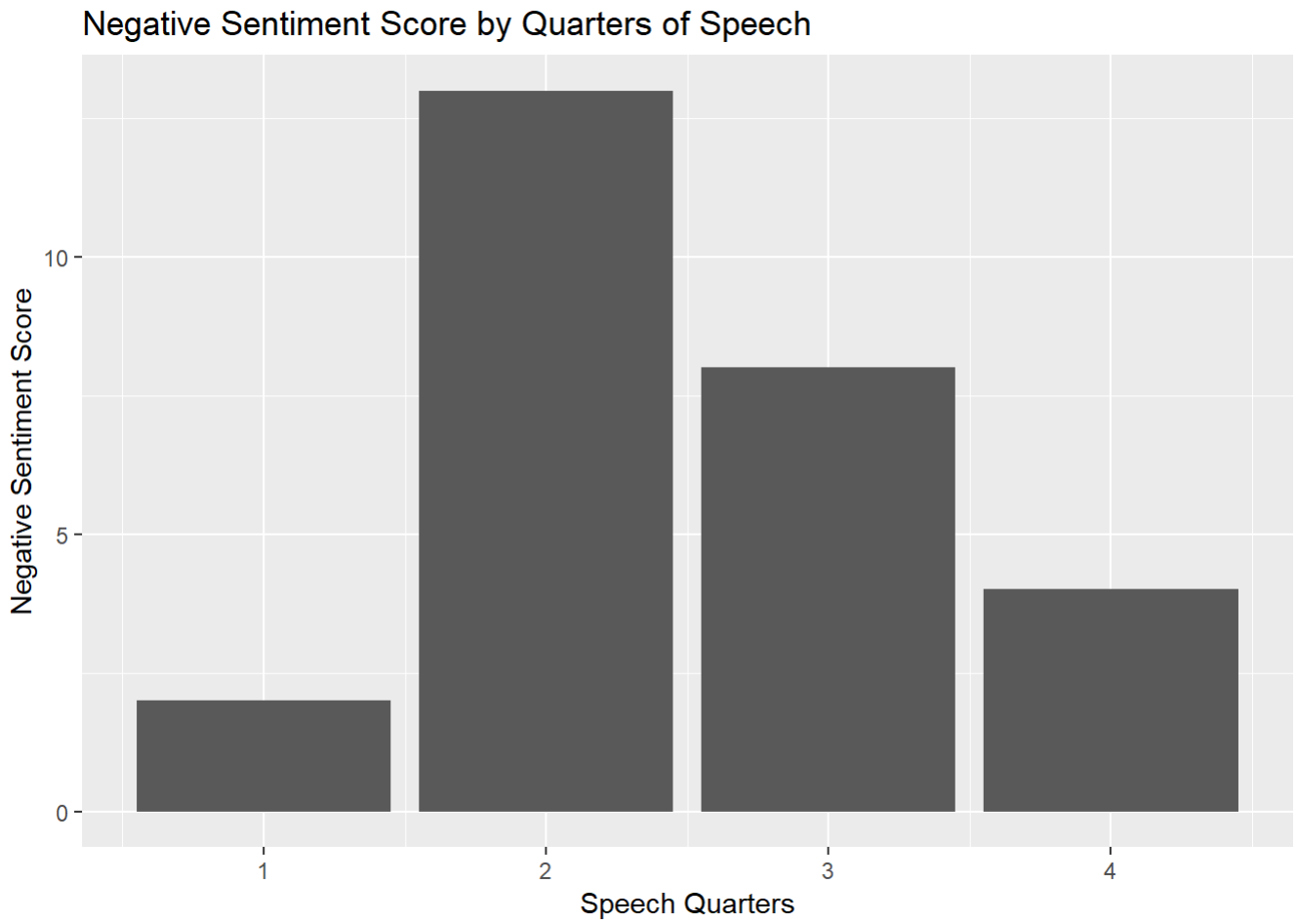
```
speech.sentiment.score <- data.frame(cbind(c(1:4),totalScore.m,pScore.m,nScore.m,ratioPosScore.
m,ratioNegScore.m))
speech.sentiment.score <-`colnames<-`(speech.sentiment.score,c("Quarter","Total_Score","Positive
_Score","Negative_Score","Positive_Ratio","Negative_Ratio"))

ggplot() + geom_bar(data = speech.sentiment.score,aes(x=Quarter,y=Total_Score),stat="identity")+
labs (x="Speech Quarters",y="Total Sentiment Score",title = "Total Sentiment Score by Quarters o
f Speech") + theme(legend.position = "bottom")
```



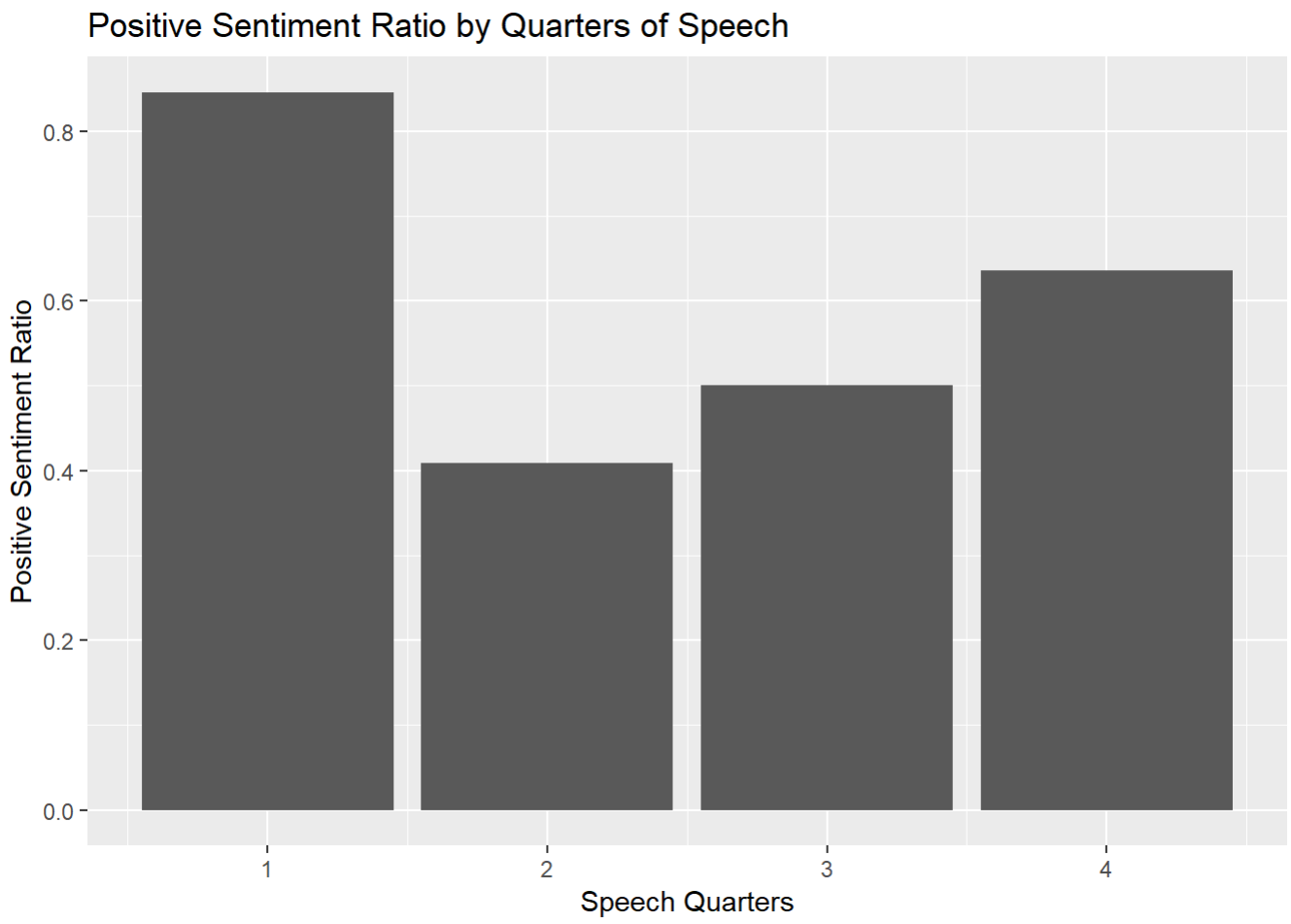

```
ggplot() + geom_bar(data = speech.sentiment.score,aes(x=Quarter,y=Positive_Score),stat="identit
y")+labs (x="Speech Quarters",y="Positive Sentiment Score",title = "Positive Sentiment Score by
 Quarters of Speech") + theme(legend.position = "bottom")
```



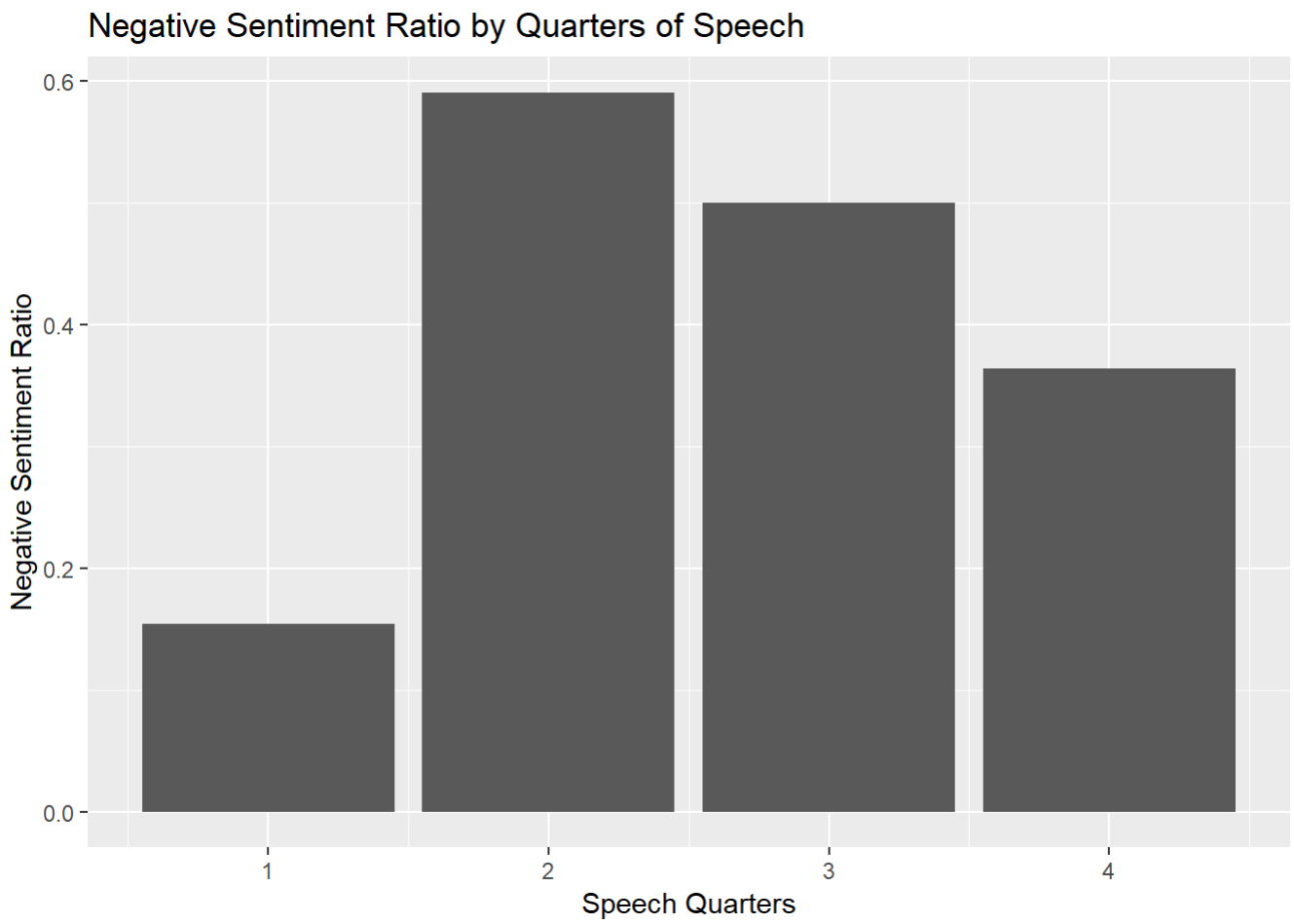

```
ggplot() + geom_bar(data = speech.sentiment.score,aes(x=Quarter,y=Negative_Score),stat="identit
y")+labs (x="Speech Quarters",y="Negative Sentiment Score",title = "Negative Sentiment Score by
 Quarters of Speech") + theme(legend.position = "bottom")
```




```
ggplot() + geom_bar(data = speech.sentiment.score,aes(x=Quarter,y=Positive_Ratio),stat="identit
y")+labs (x="Speech Quarters",y="Positive Sentiment Ratio",title = "Positive Sentiment Ratio by
 Quarters of Speech") + theme(legend.position = "bottom")
```




```
ggplot() + geom_bar(data = speech.sentiment.score,aes(x=Quarter,y=Negative_Ratio),stat="identit
y")+labs (x="Speech Quarters",y="Negative Sentiment Ratio",title = "Negative Sentiment Ratio by
 Quarters of Speech") + theme(legend.position = "bottom")
```


Negative Sentiment Ratio by Quarters of Speech
Negative Sentiment Ratio
0.6
0.4
0.2
0.0
1
2
3
4
Speech Quarters