String Manipulation

Allen Rahrooh

03 April 2020

Contents

Test Set Accuracy 63.33%

String Manipulation

```
library(readr)
bernie <- read_csv("bernie.csv")
pete <- read_csv("pete.csv")

library(stringr)
library(stringi)
library(tm)
library(magrittr)

bernie <- Corpus(VectorSource(bernie))
pete <- Corpus(VectorSource(pete))

#plain text document
bernie <- tm_map(bernie, PlainTextDocument)
pete <- tm_map(pete, PlainTextDocument)

#convert letters to all lowercase, removing punctuations
#eliminating white spaces, remove numbers
#remove english common stopwords, stemming

skipWords <- function(x) removeWords(x, stopwords("english"))
funcs <- list(tolower, removePunctuation, removeNumbers, stripWhitespace,
              skipWords, stemDocument)

bernie <- tm_map(bernie, FUN = tm_reduce, tmFuns = funcs)
pete <- tm_map(pete, FUN = tm_reduce, tmFuns = funcs)
```

Converting to Document Term Matrix

```
set.seed(984)

bernie_matrix <- DocumentTermMatrix(bernie, control = list(weighting = weightTfIdf))
bernie_matrix <- removeSparseTerms(x = bernie_matrix, sparse = 0.95)
inspect(bernie_matrix)
```

```
## <<DocumentTermMatrix (documents: 90, terms: 100)>>
## Non-/sparse entries: 618/8382
## Sparsity           : 93%
## Maximal term length: 14
## Weighting          : term frequency - inverse document frequency (normalized) (tf-idf)
## Sample             :
##     Terms
## Docs app   berni berniesanders         cna     false  sanders      true twitter
##   11   0 0.00000      0.000000 0.000000000 3.3273685 0.000000 0.5403931       0
##   12   0 0.00000      0.000000 0.000000000 3.1580699 0.000000 0.7000547       0
##   4    0 0.00000      3.316392 0.000000000 0.0000000 0.000000 0.0000000       0
##   47   0 0.00000      0.000000 0.006572612 1.3608271 0.000000 2.3597348       0
```

```
##   62   0 0.00000      0.000000 0.005086630 0.6115133 0.000000 3.0757550       0
##   73   0 1.95019      0.000000 0.000000000 0.0000000 1.842249 0.0000000       0
##   74   0 0.00000      0.000000 0.000000000 0.0000000 0.000000 0.0000000       0
##   75   0 0.00000      0.000000 0.000000000 0.0000000 0.000000 0.0000000       0
##   77   0 0.00000      0.000000 0.000000000 3.9003791 0.000000 0.0000000       0
##   84   0 0.00000      0.000000 0.000000000 0.0000000 0.000000 3.6783573       0
##     Terms
## Docs     unite   vermont
##   11 0.0000000 0.0000000
##   12 0.0000000 0.0000000
##   4  0.0000000 0.0000000
##   47 0.0000000 0.0000000
##   62 0.0000000 0.0000000
##   73 0.0000000 0.0000000
##   74 0.0000000 3.9003791
##   75 0.5535009 0.6509676
##   77 0.0000000 0.0000000
##   84 0.0000000 0.0000000
```

```
freq_bernie = data.frame(sort(colSums(as.matrix(bernie_matrix)), decreasing=TRUE))

pete_matrix <- DocumentTermMatrix(pete,
                                  control = list(weighting = weightTfIdf))
pete_matrix <- removeSparseTerms(x = pete_matrix, sparse = 0.95)
inspect(pete_matrix)
```

```
## <<DocumentTermMatrix (documents: 90, terms: 73)>>
## Non-/sparse entries: 458/6112
## Sparsity           : 93%
## Maximal term length: 14
## Weighting          : term frequency - inverse document frequency (normalized) (tf-idf)
## Sample             :
##     Terms
## Docs     bend buttigieg         cna      false     pete petebuttigieg
##   10 0.000000  0.000000 0.000000000 0.00000000 0.000000      2.941690
##   11 0.000000  0.000000 0.000000000 3.80158163 0.000000      0.000000
##   12 0.000000  0.000000 0.000000000 3.32204025 0.000000      0.000000
##   4  0.000000  0.000000 0.000000000 0.00000000 0.000000      3.164642
##   47 0.000000  0.000000 0.012433266 0.78677830 0.000000      0.000000
##   62 0.000000  0.000000 0.005401337 0.06835943 0.000000      0.000000
##   73 0.000000  1.953445 0.000000000 0.00000000 1.658196      0.000000
##   74 1.163951  0.000000 0.000000000 0.00000000 0.000000      0.000000
##   77 0.000000  0.000000 0.000000000 4.16297513 0.000000      0.000000
##   84 0.000000  0.000000 0.000000000 0.00000000 0.000000      0.000000
##     Terms
## Docs peteforamerica    south      true twitter
##   10              0 0.000000 0.0000000       0
##   11              0 0.000000 0.3613935       0
##   12              0 0.000000 0.8409349       0
##   4               0 0.000000 0.0000000       0
##   47              0 0.000000 3.3044689       0
##   62              0 0.000000 4.0673859       0
##   73              0 0.000000 0.0000000       0
##   74              0 1.054881 0.0000000       0
```

```
##   77              0 0.000000 0.0000000       0
##   84              0 0.000000 4.1629751       0
```

```
freq_pete <- data.frame(sort(colSums(as.matrix(pete_matrix)), decreasing = TRUE))


bernie_matrix_tdm <- TermDocumentMatrix(bernie)
bernie_matrix_tdm <- removeSparseTerms(x = bernie_matrix_tdm, sparse = 0.95)
inspect(bernie_matrix_tdm)
```

```
## <<TermDocumentMatrix (terms: 100, documents: 90)>>
## Non-/sparse entries: 618/8382
## Sparsity           : 93%
## Maximal term length: 14
## Weighting          : term frequency (tf)
## Sample             :
##                Docs
## Terms            11  12   4  5   6  73  74  75  77  84
##   berni           0   0   0 30   0 599   0   0   0   0
##   berniesanders   0   0 599  7   0   0   0   0   0   0
##   candid          0   0   0 20   0   0   0 600   0   0
##   false         511 485   0  1   0   0   0   0 599   0
##   presid          0   0   0 25   0   0   0 600   0   0
##   states          0   0   0  3   0   0   0 600   0   0
##   true           88 114   0  3   0   0   0   0   0 599
##   twitter         0   0   0  1 549   0   0   0   0   0
##   unite           0   0   0  6   0   0   0 600   0   0
##   vermont         0   0   0  5   0   0 599 600   0   0
```

```
bernie.matrix_tdm <- as.matrix(bernie_matrix_tdm)
bernie.freq <- sort(rowSums(bernie.matrix_tdm), decreasing = TRUE)
bernie.freq <- data.frame(word = names(bernie.freq), freq = bernie.freq)

pete_matrix_tdm <- TermDocumentMatrix(pete)
pete_matrix_tdm <- removeSparseTerms(x = pete_matrix_tdm, sparse = 0.95)
inspect(pete_matrix_tdm)
```

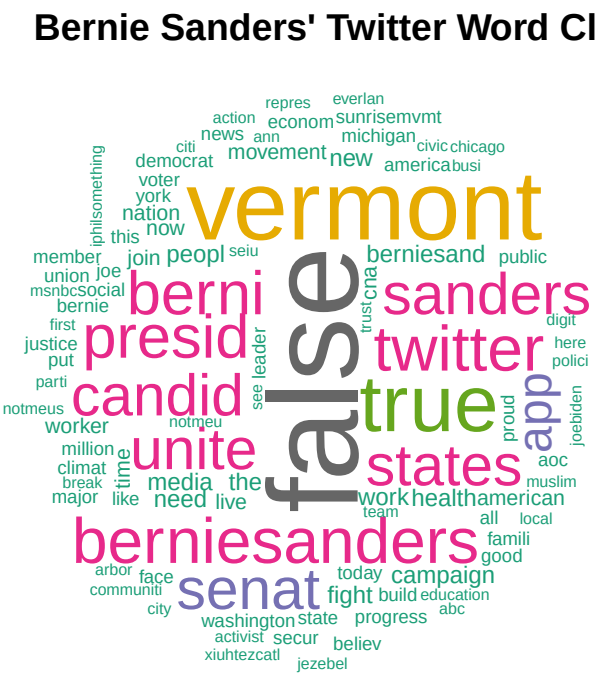
```
## <<TermDocumentMatrix (terms: 73, documents: 90)>>
## Non-/sparse entries: 458/6112
## Sparsity           : 93%
## Maximal term length: 14
## Weighting          : term frequency (tf)
## Sample             :
##                Docs
## Terms            11  12   4  5 61  73  74  75  77  84
##   bend            0   0   0 23  0   0 600   0   0   0
##   buttigieg       0   0   0  6  0 600   0   0   0   0
##   democrat        0   0   0 20  2   0   0 600   0   0
##   false         547 478   0  0  0   0   0   0 599   0
##   mayor           0   0   0 12  4   0   0 600   0   0
##   pete            0   0   0 11 65 599   0 600   0   0
##   petebuttigieg   0   0 599 65 64   0   0   0   0   0
##   south           0   0   0 39  3   0 599 600   0   0
```

```
##   true           52 121   0  0  0   0   0   0   0 599
##   twitter         0   0   0  3  9   0   0   0   0   0
```

```
pete.matrix_tdm <- as.matrix(pete_matrix_tdm)
pete.freq <- sort(rowSums(pete.matrix_tdm), decreasing = TRUE)
pete.freq <- data.frame(word = names(pete.freq), freq = pete.freq)
```

Visualizations

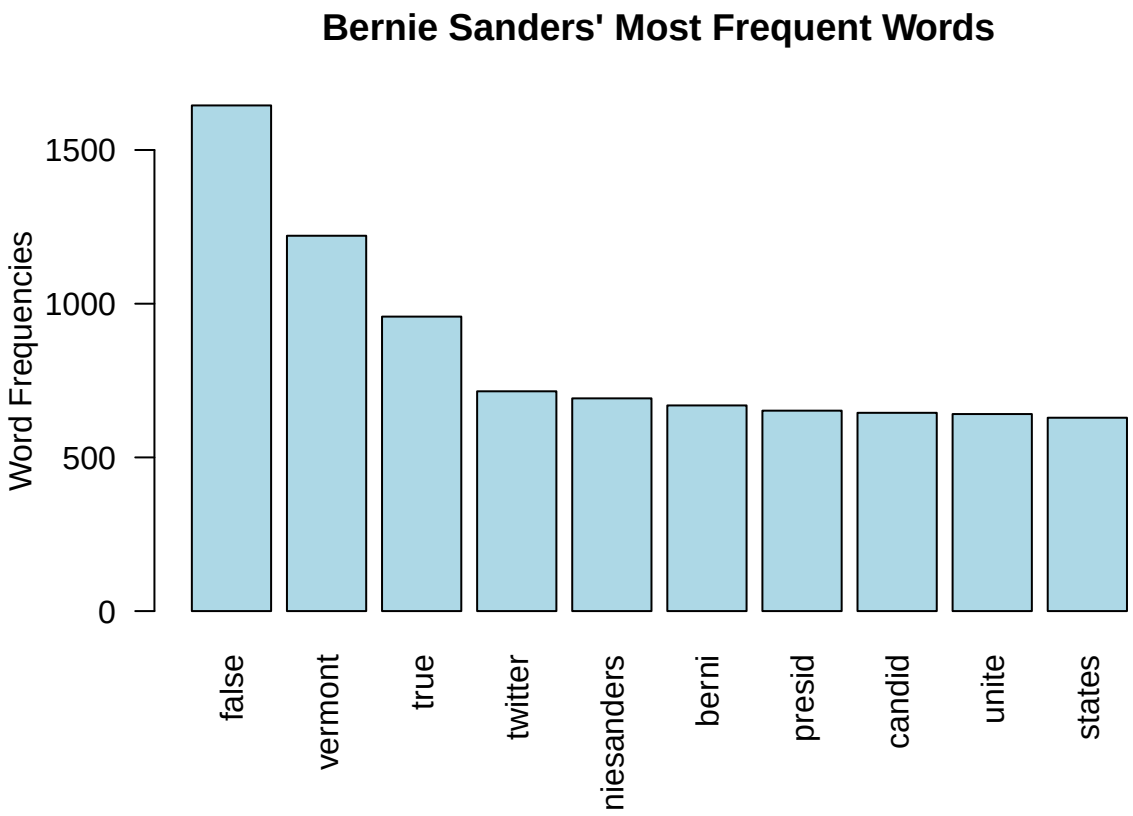
```
library(wordcloud)
library(SnowballC)
library(RColorBrewer)
library(RCurl)
library(XML)

wordcloud(words = bernie.freq$word, freq = bernie.freq$freq, min.freq = 1,
          max.words = 200, random.order = FALSE, colors = brewer.pal(8,"Dark2"),
          main = "Title")
title(main = "Bernie Sanders' Twitter Word Cloud")
```



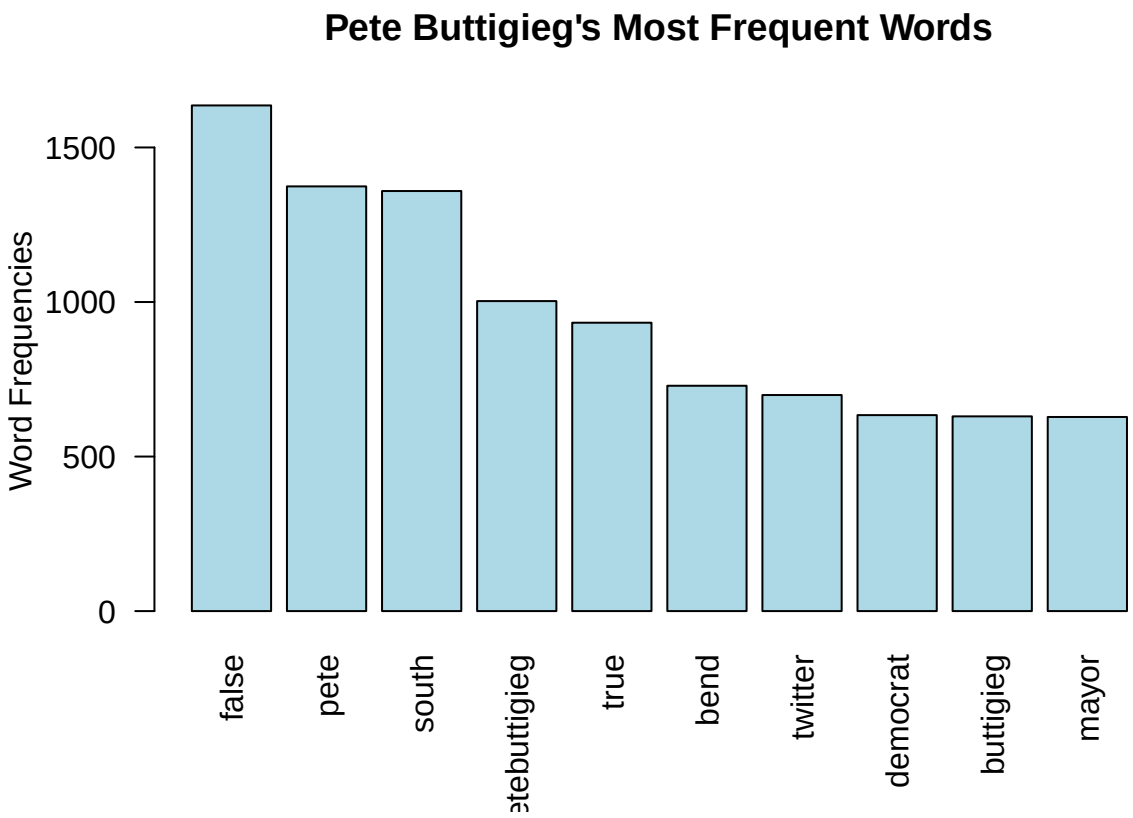

```
wordcloud(words = pete.freq$word, freq = pete.freq$freq, min.freq = 1,
          max.words = 200, random.order = FALSE, colors = brewer.pal(8,"Dark2"),
```

```
          main = "Title")
title(main = "Pete Buttigieg's Twitter Word Cloud")
```


```
barplot(bernie.freq[1:10,]$freq, las = 2, names.arg = bernie.freq[1:10,]$word,
        col = "lightblue", main = "Bernie Sanders' Most Frequent Words",
        ylab = "Word Frequencies")
```




```
barplot(pete.freq[1:10,]$freq, las = 2, names.arg = pete.freq[1:10,]$word,
        col = "lightblue", main = "Pete Buttigieg's Most Frequent Words",
        ylab = "Word Frequencies")
```



Logistic Regression Modeling

Pre-Processing Data

```
library(readr)
library(tm)
library(dplyr)

#need to make response variable
#0 for bernie, 1 for pete

#I am reading back in the csv file to make sure the regression portion is correct
bernie <- read_csv("bernie.csv")
pete <- read_csv("pete.csv")

reg_data <- rbind(bernie, pete)

reg_data$text <- tolower(reg_data$text)
reg_data$text <- removePunctuation(reg_data$text)
reg_data$text  <- removeNumbers(reg_data$text)
reg_data$text  <- removeWords(reg_data$text, stopwords("english"))
reg_data$text  <- stripWhitespace(reg_data$text)
reg_data$text  <- stemDocument(reg_data$text, language = "english")
```

```
reg_data <- bind_rows(bernie,pete)
reg_data <- VCorpus(VectorSource(reg_data$text))
reg_data <- DocumentTermMatrix(reg_data)
reg_data <- removeSparseTerms(x = reg_data, 0.95)
reg_data <- as.matrix(reg_data)
reg_data <- as.data.frame(reg_data)

#first 600 rows are bernie (0)
#second 600 rows are pete (1)

pol <- c(rep(0, 600), rep(1, 600))
pol <- as.data.frame(pol)

reg_data <- cbind(pol, reg_data)
```

Variable Selection

```
library(e1071)
library(MASS)
#Regression modeling with full dataset

pol.glm <- glm(pol~ ., family = "binomial", maxit = 100,
                  data = reg_data)

step_model <- stepAIC(pol.glm, direction = "both", trace = FALSE)
step_model <- model.matrix(step_model)
step_model <- as.data.frame(step_model)

step_model_glm <- step_model[,-(1:2)]

step_model_glm <- cbind(pol, step_model_glm)
```

Train Test Split

```
library(caTools)
set.seed(345)

id_train <- sample(nrow(step_model_glm), nrow(step_model_glm)*0.80)
train.dtm <- as.data.frame(as.matrix(step_model_glm[id_train,]))
test.dtm <- as.data.frame(as.matrix(step_model_glm[-id_train,]))
```

Modeling

```
pol.glm_red <- glm(pol ~. ,
                family = "binomial", maxit = 100, data = train.dtm)

summary(pol.glm_red)
```

```
##
## Call:
## glm(formula = pol ~ ., family = "binomial", data = train.dtm,
##     maxit = 100)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7435  -1.0132  -0.2278   1.0545   2.1582
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.29627    0.13108   2.260 0.023802 *
## american     0.96261    0.30712   3.134 0.001723 **
## americans    1.98637    0.40852   4.862 1.16e-06 ***
## are         -0.69881    0.16178  -4.319 1.56e-05 ***
## but          1.33442    0.30878   4.322 1.55e-05 ***
## care        -0.78914    0.27116  -2.910 0.003611 **
## chip         1.22844    0.40006   3.071 0.002136 **
## donald       0.19895    0.33144   0.600 0.548334
## `\\`for\\`` -0.37474    0.09854  -3.803 0.000143 ***
## get         -0.65302    0.26981  -2.420 0.015509 *
## have        -0.25644    0.15846  -1.618 0.105595
## help         0.66213    0.32631   2.029 0.042447 *
## more         0.55100    0.29254   1.883 0.059636 .
## must        -0.74277    0.25168  -2.951 0.003165 **
## need        -0.42614    0.21591  -1.974 0.048417 *
## new          1.00729    0.29873   3.372 0.000747 ***
## not         -0.26223    0.20912  -1.254 0.209863
## one         -0.50580    0.25997  -1.946 0.051699 .
## president    0.53366    0.31826   1.677 0.093580 .
## the         -0.09030    0.06207  -1.455 0.145696
## their        0.23396    0.22493   1.040 0.298271
## who          0.94905    0.19481   4.872 1.11e-06 ***
## will        -0.63407    0.18858  -3.362 0.000773 ***
## working     -1.04524    0.32641  -3.202 0.001363 **
## you         -0.03123    0.13250  -0.236 0.813651
## your         0.08743    0.17991   0.486 0.627000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1330.8  on 959  degrees of freedom
## Residual deviance: 1125.0  on 934  degrees of freedom
## AIC: 1177
##
## Number of Fisher Scoring iterations: 4
```

Prediction

```
pred.glm_pol <- as.numeric(predict(pol.glm_red, test.dtm, type = "response") > 0.5)
```

```r
table(test.dtm$pol, pred.glm_pol, dnn = c("Obs", "Pred"))
```

```
##    Pred
## Obs  0  1
##   0 66 52
##   1 36 86
```

```r
test_acc <- (mean(ifelse(test.dtm$pol != pred.glm_pol, 0, 1)))*100

cat("The test set accuracy is: ", test_acc, "%")
```

```
## The test set accuracy is:  63.33333 %
```

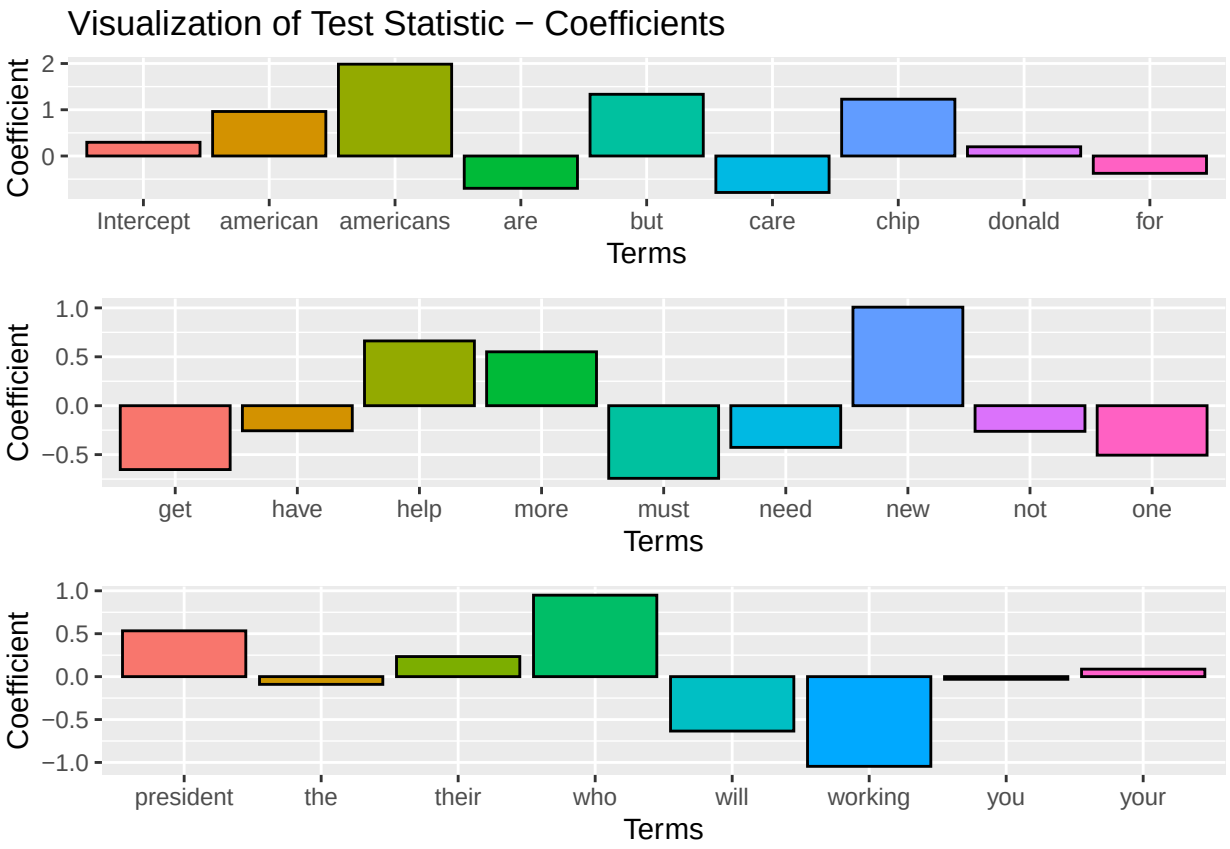
```r
cat("\n")
```

Visualization of Test Statistic

```r
library(ggplot2)
library(gridExtra)

test_stat <- as.data.frame(as.matrix(pol.glm_red$coefficients))

test_stat
```

```
##                       V1
## (Intercept)  0.29627031
## american     0.96260727
## americans    1.98637051
## are         -0.69881102
## but          1.33442000
## care        -0.78913891
## chip         1.22843899
## donald       0.19895066
## `\\`for\\`` -0.37473582
## get         -0.65302424
## have        -0.25643726
## help         0.66212870
## more         0.55099863
## must        -0.74276587
## need        -0.42613631
## new          1.00728806
## not         -0.26222935
## one         -0.50580471
## president    0.53365784
## the         -0.09030196
## their        0.23396048
## who          0.94905343
## will        -0.63407351
## working     -1.04524412
## you         -0.03123316
## your         0.08742918
```

```
dat <- data.frame(
  Terms = factor(c("Intercept", "american","americans","are", "but", "care",
                   "chip", "donald", "for"),
                 levels = c("Intercept", "american","americans","are", "but", "care",
                   "chip", "donald", "for")),
  Coefficient = test_stat$V1[1:9]
)

p1 <- ggplot(data = dat, aes(x = Terms, y = Coefficient, fill = Terms)) +
  geom_bar(colour = "black", stat = "identity") +
  guides(fill = FALSE) +
  ggtitle("Visualization of Test Statistic - Coefficients")


dat1 <- data.frame(
  Terms = factor(c("get", "have", "help",
                   "more","must", "need", "new", "not", "one"),
                 levels = c("get", "have", "help",
                   "more","must", "need", "new", "not", "one")),
  Coefficient = test_stat$V1[10:18]
)

p2 <- ggplot(data = dat1, aes(x = Terms, y = Coefficient, fill = Terms)) +
  geom_bar(colour = "black", stat = "identity") +
  guides(fill = FALSE)

dat2 <- data.frame(
  Terms = factor(c("president", "the", "their", "who", "will",
                   "working", "you", "your"),
                 levels = c("president", "the", "their", "who", "will",
                   "working", "you", "your")),
  Coefficient = test_stat$V1[19:26]
)

p3 <- ggplot(data = dat2, aes(x = Terms, y = Coefficient, fill = Terms)) +
  geom_bar(colour = "black", stat = "identity") +
  guides(fill = FALSE)


grid.arrange(p1,p2,p3)
```


Visualization of Test Statistic − Coefficients
Coefficient
2
1
0
Intercept
american
americans
are
but
care
chip
donald
for
Terms
Coefficient
1.0
0.5
0.0
−0.5
get
have
help
more
must
need
new
not
one
Terms
Coefficient
1.0
0.5
0.0
−0.5
−1.0
president
the
their
who
will
working
you
your
Terms