Twitter Sentiment Analysis

Mounika Narayanan

March 14, 2018

1. Introduction

This project aims to gather tweets using Twitter's API and then perform sentiment analysis on the given topic(s) to garner interesting findings. I became interested in this topic because I really wanted to learn how Twitter's API works and found it fascinating and fun to create my own dataset. I also am very passionate about the intersection of technology and other disciplines. Therefore, I wanted to use my tech skills (scraping and sentiment analysis) to see if I could draw any interesting conclusions about political and social leanings about controversial issues.

The process entailed gathering a dataset of tweets using Twitter's API, cleaning the data and removing duplicates, applying a bag of words algorithm to assign each tweet a score, and then creating visualizations to convey any findings (i.e. histograms of scores plotted against various factors).

The topic I initially chose to study was James Damore's controversial Google Memo. For those unfamiliar with this, it was a dissertation writtten by a former Google employee, James Damore, which explained his perspective about women in technology and in the workplace. After his memo was released and received a lot of backlash, Google CEO decided to fire him. I thought that this topic would be really interesting to study because I was curious as to what sort of leanings people had about this issue. However, towards the middle to end of my research during the quarter, gun violence became a hot controversial topic that trended after the unfortunate Florida shooting. Thus, I decided to conduct my analysis on both topics.

2. Webscrape Using Twitter's API

I used the package twitteR to interact with Twitter (get tweets, investigate users, convert tweet lists into dataframes, etc.). To get my data, I had to create a Twitter app online, which then gave me my own consumer key, consumer secret, access token, and access secret; I was then able to connect with Twitter and then read in tweets using function searchTwitter(). I understood that there was a rate limit which prevented me from reading in an infinite amout of tweets. Thus, in order to get a larger sample, every time I worked on the project over the quarter, I pulled more tweets and wrote them to my local drive. Afterwards I read in all the files I had written to my local drive into R and used rbind() to combine them all and conduct my cleaning and analysis.

3. Cleaning the Data

To clean my data, I used gsub and regex expressions to remove emojis, unprintable characters, URLs, truncated text, special characters, the hashtag symbol, @___ handles, punctuation, decimal numbers, new line characters, and trailing spaces at the beginning and ending of the text. I then converted my text to lowercase and removed duplicated rows for analysis.

4. Bag of Words Score Algorithm

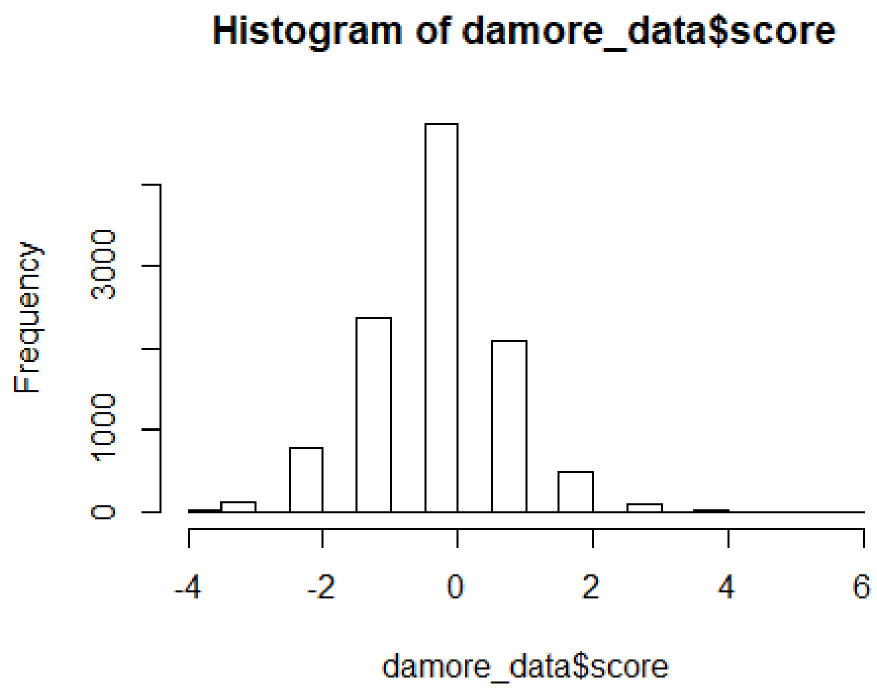
The bag of words algorithm is actually quite simple. Its based off of the idea that there are classically “negative” and “positive” meaning words in the English language. The more “negative” words there are in a sentence or tweet, the more “negative” the sentiment. The same is true for positive sentiments.

The process I used is outlined below: 1. I read in a moderately comprehensive positive and negative words list into R. 2. I used strsplit() to split each tweet into its individual words. 3. I then used an lapply() to check if a word in the tweet was found in the positive or negative word list. 4. If a positive word is found, a point is added to the positive count for that tweet. If a negative word is found, a point is added to the negative count for that tweet. 5. After creating both positive sum and negative sum columns in my dataframe, I created a column called “score” which takes he positive sum minus the negative sum.

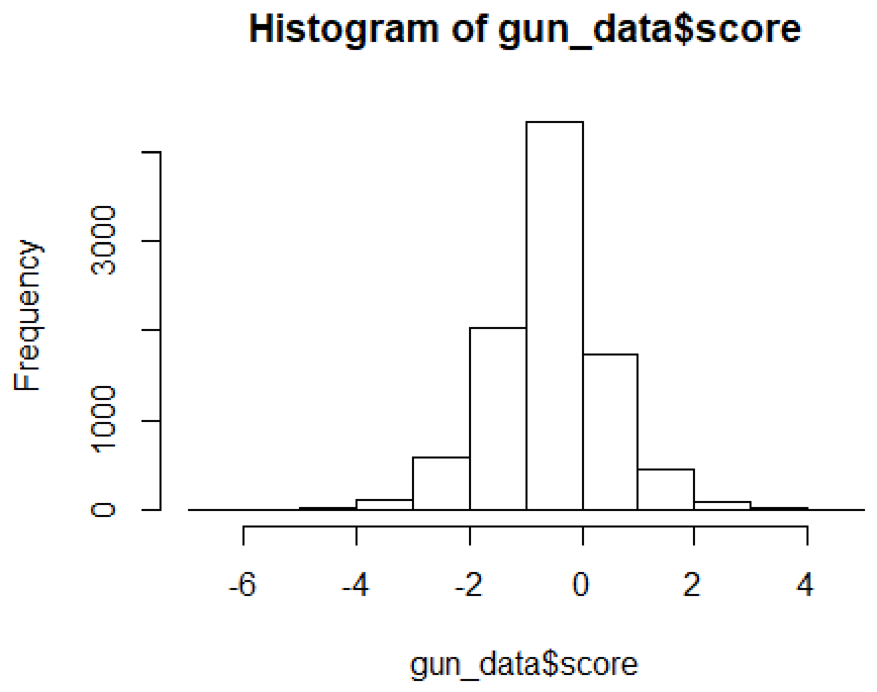
5. Analyses

5.1 Distribution of Scores

```r
setwd("C:/Users/Mounika/Documents/RESEARCH")
damore_data <- read.csv("memo_final.csv", stringsAsFactors = F)
gun_data <- read.csv("gun_data.csv", stringsAsFactors = F)

hist(damore_data$score)
```



```
hist(gun_data$score)
```



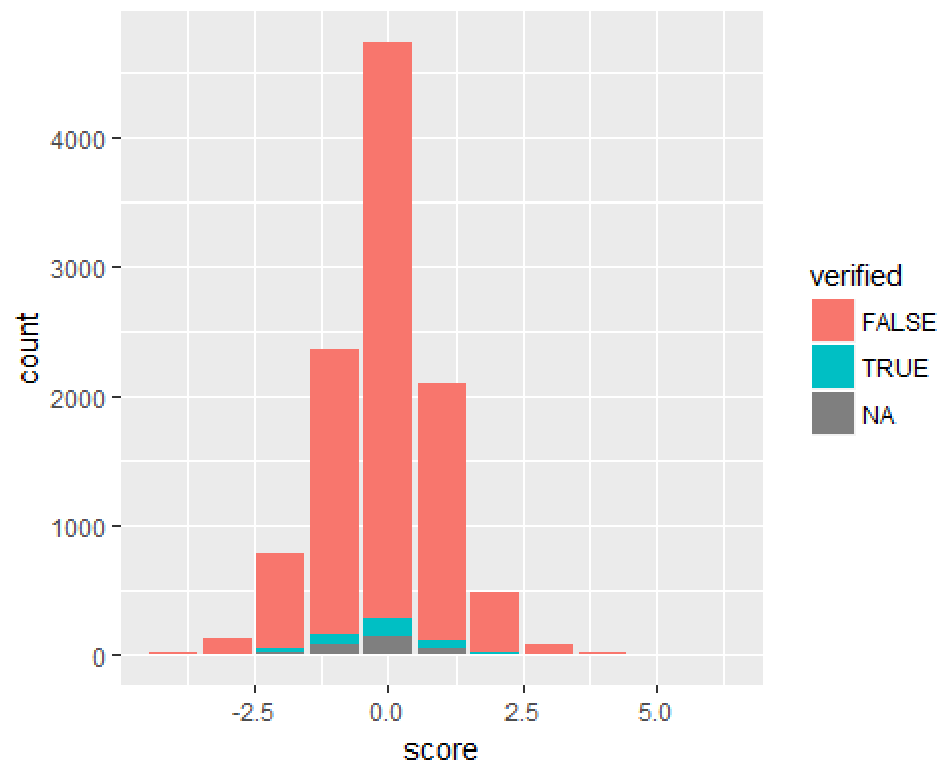
5.2 Scores by Other Factors

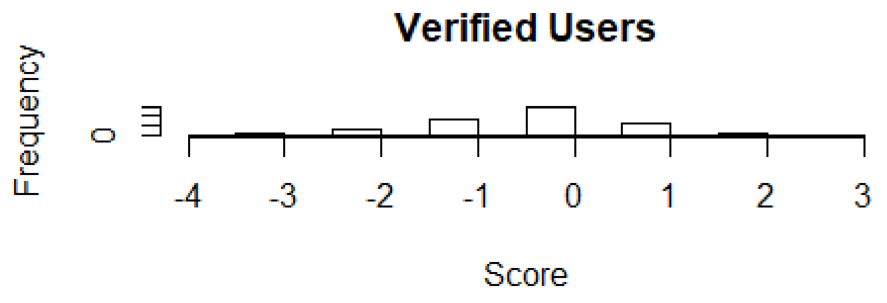
I was able to get user information in the form of user objects using the function lookupUsers() in package twitteR which returns a list of user objects. I iterated through the list and extracted the information I was interested in and merged it with my original dataframe.

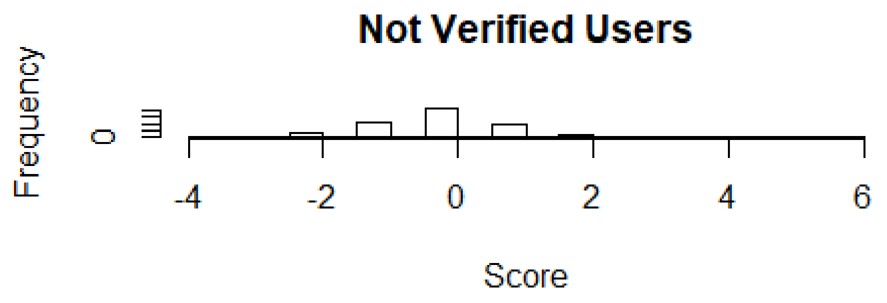
i. Verified

I was curious to see if there was a difference in scores between verified vs not verified users.

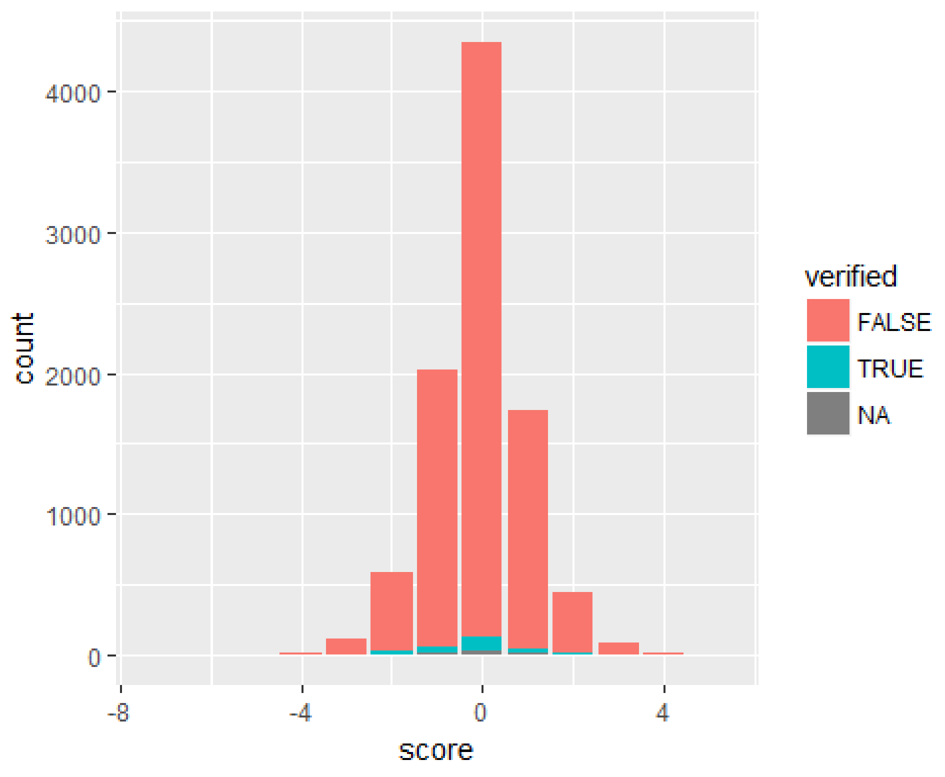
```
library(ggplot2)

verifiedfct <- function(df)
{
  verified <- as.data.frame(cbind(df$score, df$verified))
  names(verified) <- c("score", "verified")
  verified$verified <- as.logical(verified$verified)

  graph <- ggplot() + geom_bar(aes(x=score, fill=verified), data=verified,
stat='count')
  print(graph)

  par(mfrow=c(2,1))
  hist(verified[verified$verified==T,]$score, main = "Verified Users",
xlab="Score")
  hist(verified[verified$verified==F,]$score, main="Not Verified Users",
xlab="Score")

  t.test(verified$score~verified$verified, na.rm=T)
}

verifiedfct(damore_data)
```

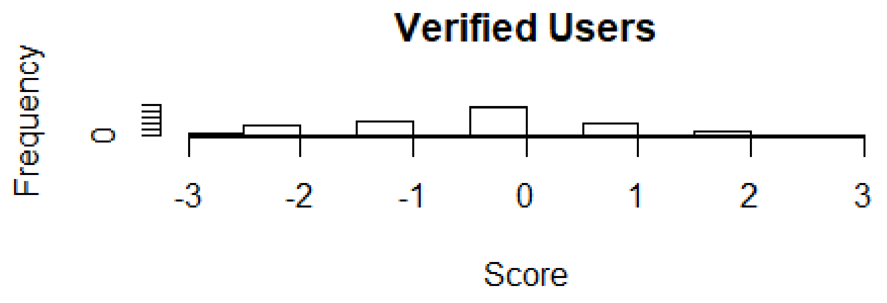


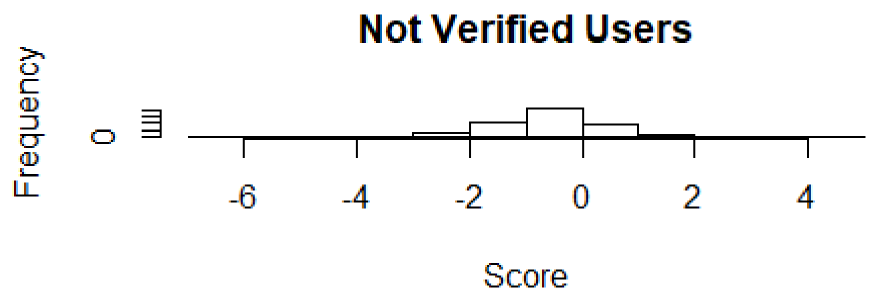




```
##
##  Welch Two Sample t-test
##
## data:  verified$score by verified$verified
## t = 2.2794, df = 371.39, p-value = 0.02321
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.01841385 0.24977746
## sample estimates:
## mean in group FALSE  mean in group TRUE
##         -0.08080406         -0.21489971

verifiedfct(gun_data)
```







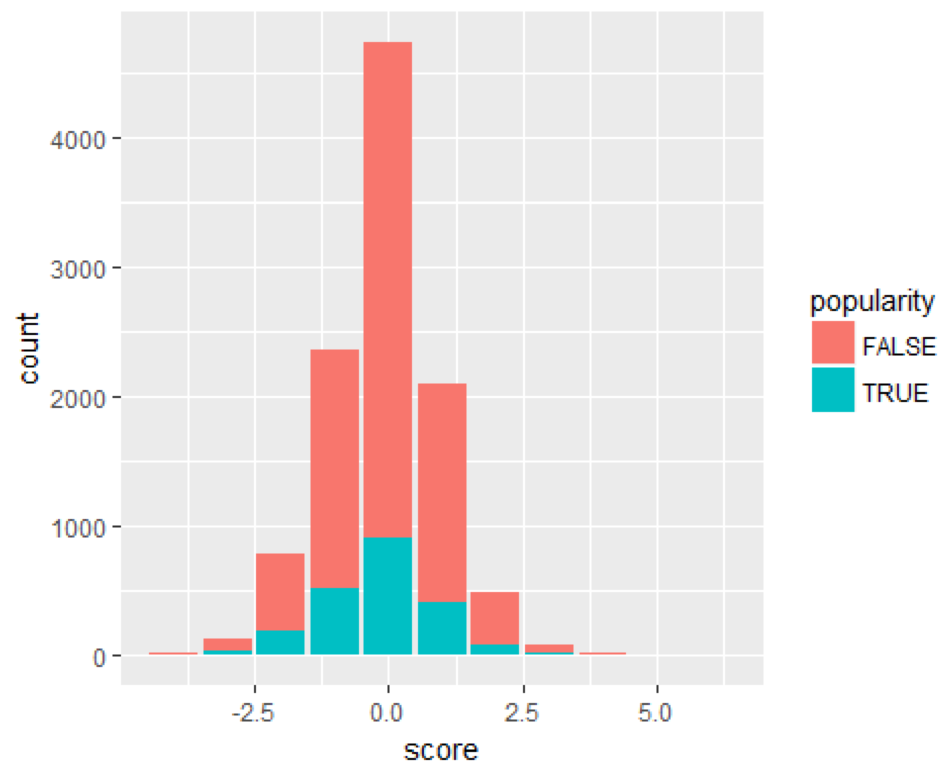
```
##
##  Welch Two Sample t-test
##
## data:  verified$score by verified$verified
## t = 2.142, df = 243.26, p-value = 0.03318
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.0130473 0.3114156
## sample estimates:
## mean in group FALSE  mean in group TRUE
##         -0.07281126         -0.23504274
```

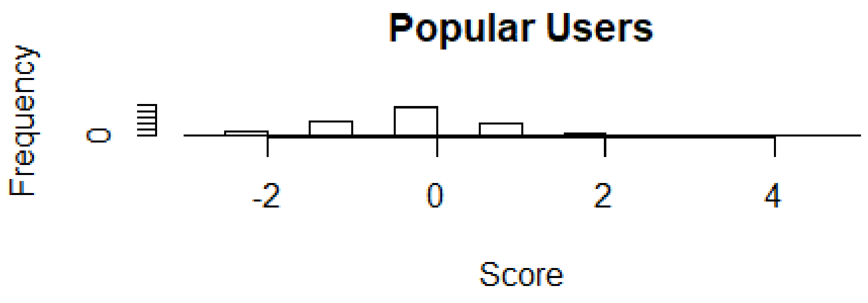
The graphs do not seem to show a significant difference in scores. However, the t-test shows that the verified users had significantly lower scores than not verified users for both the damore and gun data sets.

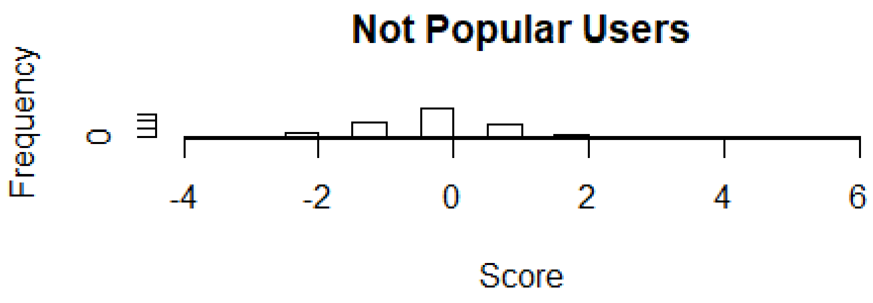
ii. Popularity

Because, not a significant amount of my sample is "verified," I decided to take a more mechanical approach to determining fame/popularity of users. I assigned the top 20% users with the most followers "popular," and the rest were assigned "not popular." I then plotted the scores by popularity to see if there was any meaningful difference.

```r
followersfct <- function(df)
{
  followers <- as.data.frame(cbind(df$score, df$followersCount))
  names(followers) <- c("score", "followersCount")
  followers <- followers[order(followers$followersCount),]
  followers$popularity <- T
  followers[1:((nrow(followers) - round(nrow(followers)*0.2))), 3] <- F

  graph <- ggplot() + geom_bar(aes(x=score, fill=popularity), data=followers,
stat='count')
  print(graph)

  par(mfrow=c(2,1))
  hist(df[followers$popularity==T,]$score, main = "Popular Users",
xlab="Score")
  hist(df[followers$popularity==F,]$score, main = "Not Popular Users",
xlab="Score")

  t.test(followers$score~followers$popularity, na.rm=T)
}

followersfct(damore_data)
```

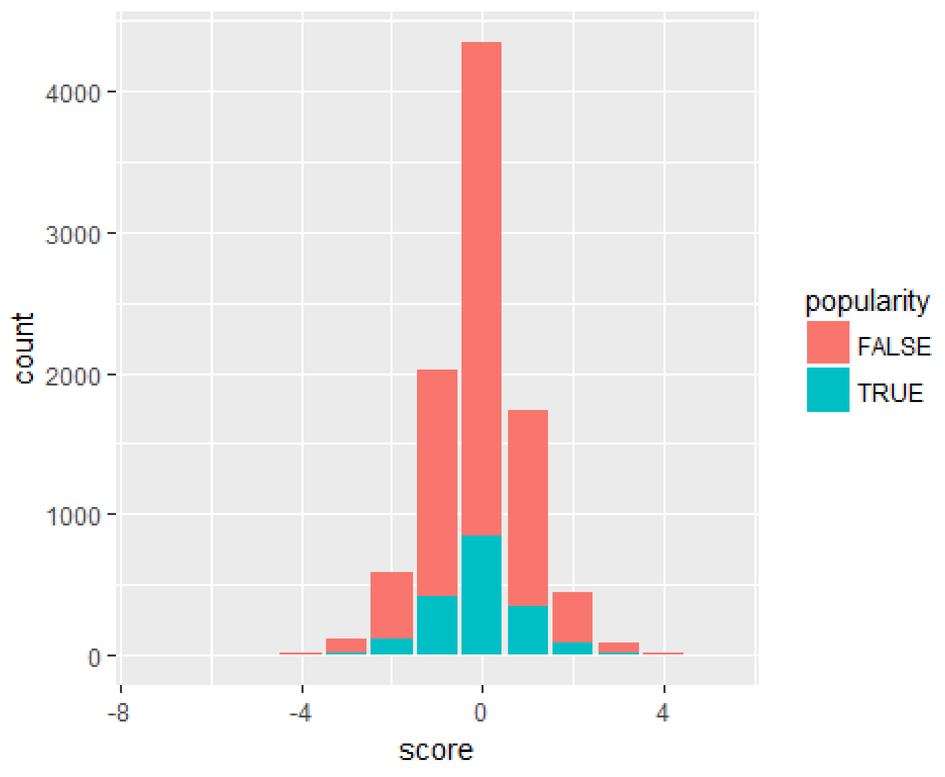


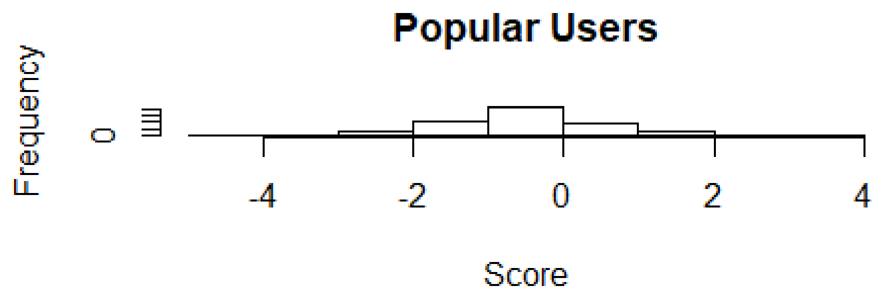


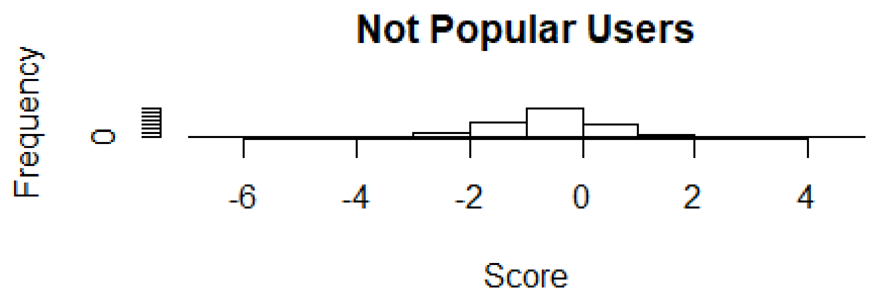


```
##
##  Welch Two Sample t-test
##
## data:  followers$score by followers$popularity
## t = 4.7438, df = 3296.3, p-value = 2.186e-06
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.07062716 0.17013956
## sample estimates:
## mean in group FALSE  mean in group TRUE 
##         -0.06521739         -0.18560075
```

```
followersfct(gun_data)
```







```
##
##  Welch Two Sample t-test
##
## data:  followers$score by followers$popularity
## t = -0.65408, df = 2910.7, p-value = 0.5131
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.07081298  0.03538656
## sample estimates:
## mean in group FALSE  mean in group TRUE
##         -0.08007995         -0.06236674
```

The graphs do not seem to show a significant difference in scores. However, the t-test for the damore data shows that the popular users had significantly lower scores than not popular users. There was no significant difference between popular and not popular scores for the gun data.

iii. Gender and Location

I thought it might be interesting to plot the scores based on gender to see if there was a significant difference between them. However, the only information available using the API is the "name" of a given user, which is not always the user's actual name. My initial idea was to read in a comprehensive list of typical female and male names, see if any name in the list of names (male or female) matched, and assign genders to each user. With the unreliability of this data, this analysis was not feasible.

I also thought it might be interesting to see if different locations had different opinions about the memo and gun violence. The API is able to give location information, but that too is unreliable and is determined by the user.

6. Validation and Limitations Faced

6.1 API Limitations

I wanted to validate my bag of words model, so I sampled 50 tweets from my dataframe with tweets dealing with James Damore's Memo. I assigned my own scores to these 50 tweets, and applied a paired t-test. There showed no significant difference in scores, which was a good sign, I had thought. However, as I was reading through a good chunk of my tweets, I had realized that the tweets themselves were not conveying very many sentiments to begin with. This I realized was a weakness of the API rate limit.

The API only returns the n most recent tweets and the rate limit makes it even more restrictive. Thus, even though I did do several different pulls of data over the course of several weeks, my data is still not as comprehensive as I would have liked it to have been. The tweets that show up in my data are also dependent on what time of day I pulled the data, which is another confounding variable that may affect what tweets I am pulling.

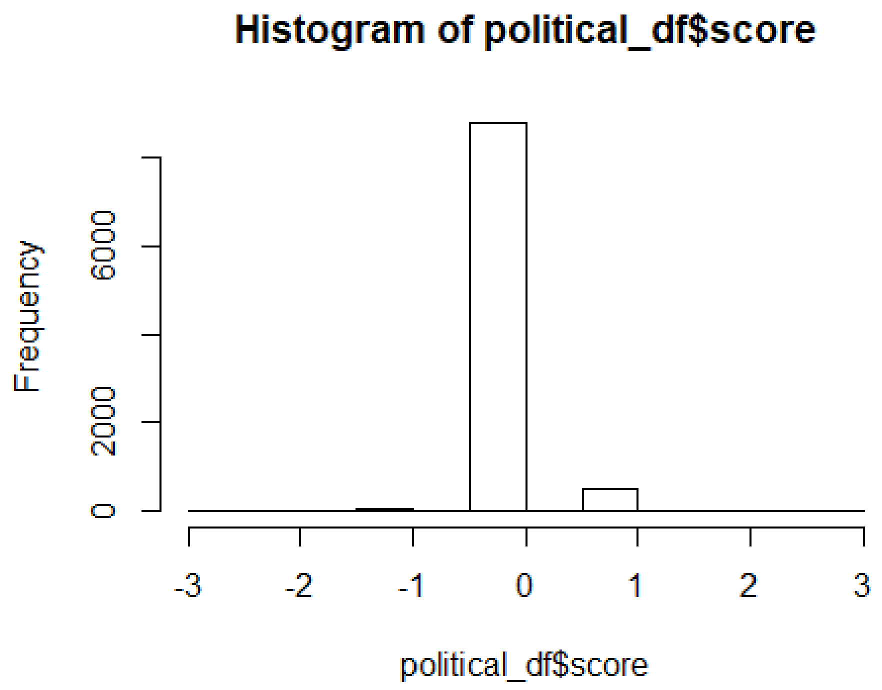
I also realized that majority of the tweets are retweets of news posts and events that are happening, not actual opinions of these events from users themselves. Thus, most of the data I pull gets deleted anyways because I remove duplicates to ensure that I do not account for a score coming from a given tweet more than once in my analysis.

6.2 Algorithm Limitations

As I had stated in an earlier section, the bag of words algorithm is used to assess sentiment on general words in the English language and fails to take into consideration context. Certain analyses (especially political ones) require a more robust bag of words because political leanings are not necessarily "positive" or "negative"; perhaps they are more liberal or conservative. This was my motivation behind creating my own bag of words but for "left" and "right" political leanings dealing with gun violence. Below are the results of my attempted contextual bag of words. Unfortunately, the tweets did not match my words frequently enough, rendering this analysis not that insightful. I had realized that my bag of words was not comprehensive enough, and a more robust algorithm is necessary to better determine sentiment.

```r
# my own bag of words
right<-c("antigun control", "2a", "defendthesecond", "nra", "maga",
"altleft", "libs", "leftist")
left<-c("gunreformnow", "guncontrol", "guncontrolnow", "gunviolence",
"altright", "rightist")

# sentiment function
score.sentiment <- function(tweet_text, pos.words, neg.words,
.progress='none')
{
  require(plyr)
  require(stringr)

  word.list <- str_split(tweet_text, "\\s+")
  pos_matches <- lapply(word.list, function(x) match(x, pos.words))
  neg_matches <- lapply(word.list, function(x) match(x, neg.words))

  pospoints <- lapply(pos_matches, function(x) !is.na(x))
  negpoints <- lapply(neg_matches, function(x) !is.na(x))

  possum <- unlist(lapply(pospoints, sum))
  negsum <- unlist(lapply(negpoints, sum))

  score <- possum - negsum

  vals <- as.data.frame(cbind(score, possum, negsum))

  return(vals)
}

politicalscores <- score.sentiment(gun_data$text, right, left, .progress =
"text")

## Loading required package: plyr

## Loading required package: stringr
```

```r
political_df <- as.data.frame(cbind(gun_data$text, gun_data$screenName,
politicalscores))
political_df$score <- as.numeric(as.character(political_df$score))

hist(political_df$score) # not conclusive
```



7. Conclusion and Future Research

7.1 No Significant Skew of Scores

In conclusion, I do not think I can conclude that there is a significant positive or negative trend regarding the Damore memo or gun violence. However, there were some interesting findings regarding sentiments about these topics for verified/not verified and popular/not popular users.

7.2 Future Research

There are three avenues in which one could improve this project in the future: 1. In order to increase the breadth of my sample and combat the API limitations, I would like to implement a bash script that runs in background that reads tweets and writes files to my local drive. Therefore, when I am ready to begin my analysis, I can combine the tweets on all the files and complete a more comprehensive analysis. 2. I would use a different algorithm to calculate scores by using a clustering method. For example, package word2vec converts words, phrases, and sentences to vectors and applies a clustering algorithm (unsupervised learning) to these vectorized components.

3. I would explore other controversial topics, depending on what is popular in politics, news, and pop-culture at a future time.

Thank you for reading my writeup. I hope you find my project useful for you!