MA615 Final Project

Matthew D. Ciaramitaro

May 7, 2018

Project Summary

The goal of this project was to analyze popular politcs related twitter queries to determine the qualities that make them popular. We examine factors such as the length of a tweet, the most common words in popular tweets, and the most common sentiments of popular tweets.

Dataset

What is the popularity of a tweet? In this case we consider the unweighted sum of the number of retweets and the number of favorites to that tweet. We could weigh the augend and addend to favor favorites or retweets, but we have no justification to do so. We sum them because it records the amount of user interactions for that tweet.

We choose the following search terms for our query

```
search_terms = 'politics OR democrats OR republicans OR socialist OR healthcare OR Gun+Control
               OR Syria OR Mueller OR blacklivesmatter OR NSA OR drone+strikes OR Obama OR Sanders
               OR Clinton OR Trump OR progressive OR conservative OR liberal OR immigration
               OR right+wing OR left+wing OR war OR United+States OR US OR education
               OR congress OR senate OR president OR VP OR POTUS OR SCOTUS OR law
               OR bill OR hor OR vote
               '
print(search_terms)
```

```
## [1] "politics OR democrats OR republicans OR socialist OR healthcare OR Gun+Control\n                (
```

We are limited from including more terms by the rtweet API, which will throw an error with too many terms.

We then define our stop words, which are words which we ignore during analysis, due to their commonality and lack of contribution to the interesting properties of the data in most of NLP. This mostly includes pronouns, very common verbs and their conjugations, and contractions.

```
### Run this code once if stopwords.csv does not exist
#words to ignore
#stop_words <-  data.frame(stopwords(kind="en"))
#colnames(stop_words) <- c("word")
#write.csv(stop_words, "stopwords.csv")
###

stop_words <- read.csv("stopwords.csv")
```

We have modified R's default stop words to include https and t.co, which are common sequences of characters in tweets which contain urls. t.co represents most of the compressed urls. There are sequences of characters after those involved in URLs but those are too uncommon to affect our analysis.

Now we verify our account with twitter

```
###run this code once if twitter_token.rds file does not exist
#key <- "DhEKno4O4jyDE7H4tgxw2KCJv"
#This secret is not real for privacy purposes; This code will not run unless the key and secret are cor
```

```
#secret <- "dR581H8j6PMAWbaDKu4ncLEsxDBE5BB2cxpgtY4hnzFFenFB9"
#
# twitter_token <- create_token(
#   app = "MA615Final",
#   consumer_key = key,
#   consumer_secret = secret,
#   set_renv=F
# )
# saveRDS(twitter_token, "./twitter_token.rds")
###
twitter_token <- readRDS("./twitter_token.rds")
cat(paste0("TWITTER_PAT=", "./twitter_token.rds"),
     file = file.path("./", ".Renviron"),
     append = TRUE)
```

And finally we pull the data into an rds file for the user

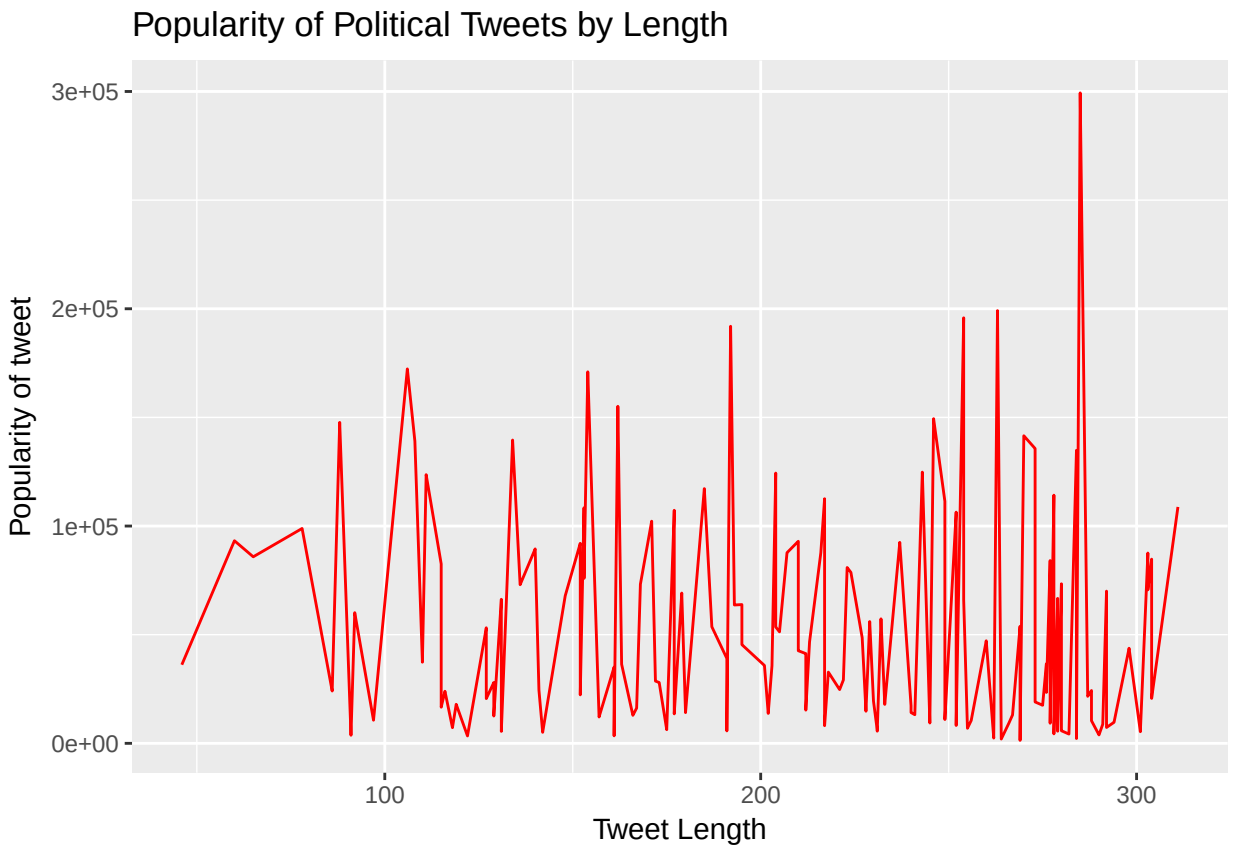
```
get_tweets <- function(search_terms){
  #Download tweets with search terms into a df
  tw <-  distinct(rtweet::search_tweets(search_terms, n = 1e6, type="popular", retryonratelimit = 1e4,
  #Collect column for text and column for the amount of shares and favorites
  tw <-  tw %>% mutate(popularity = retweet_count + favorite_count) %>% select(text, popularity)
  #Remove duplicate text
  textpop <- tw[!duplicated(tw[,c('text')]),]
  #Now we have the popularity of each tweet
  print("Saving file")
  saveRDS(textpop, file="tweets.rds")
  print("File saved")
  return(textpop)
}

###run the below to generate tweets.rds
#get_tweets(search_terms)
###

#arranged by popularity in descending order for print
textpop <- arrange(readRDS("tweets.rds"), desc(popularity))
print(textpop)
```

```
## # A tibble: 174 x 2
##    text                                                          popularity
##    <chr>                                                              <int>
##  1 "Dear @kanyewest,\n\nTrump asked if there were any "Hispani~     299348
##  2 I attended the WHCD last night. Donald Trump has so poisone~     199142
##  3 You can show me all the individuals you want. I'm sure they~     195828
##  4 "A white guy with an AR-15 killed 4 blacks at a Waffle Hous~     191929
##  5 Just got recent Poll - much higher than President O at same~     172323
##  6 Very fortunate to be adding another award in my first seaso~     170973
##  7 infinity war cemented tom holland as the best spiderman. I'~     155109
##  8 This Army soldier's flight was delayed. He had to watch the~     149404
##  9 BREAKING: North Korea has released all U.S. detainees at th~     147705
## 10 House Intelligence Committee rules that there was NO COLLUS~     141536
## # ... with 164 more rows
```

Length vs Popularity

We now determine the relationship between length and the popularity of a tweet.

```
lenpop <- textpop %>% mutate(length = nchar(text))
ggplot(lenpop %>% select(length,popularity), aes(x=length, y=popularity))+
  geom_line(color="red")+
  xlab("Tweet Length")+
  ylab("Popularity of tweet")+
  ggtitle("Popularity of Political Tweets by Length")
```



According to this plot, there is no obvious relationship between length and height, until we approach the maximum tweet length, which is 280 characters (not including urls). At this length we see a large spike in the popularity of tweets, as the most popular tweets tend to use all the characters that they are allotted for their tweets.

Sentiment Analysis

Now we look at the sentiments in each tweet, and determine what percentage a sentiment appears over the whole set of tweets.

```
#Now we have the number of occurences of each word in the popular tweets
#Sentiment Analysis
get_sentimentss <- function(df, sentiments, na.rm=F){
  #given a list of sentiments, cross references words in the data frame with list, may remove NA rows
  #NA rows are ones where the df contains a word which is not in the list of words/sentiments nrc
  nrc <- filter(nrc, sentiment %in% sentiments)
  ds <- df %>% left_join(nrc, by="word")
```

```
  if(na.rm){
    ds <- ds[complete.cases(ds), ]
  }
  return(ds)
}
get_pct_occur <- function(df, sentiments){
  #get df of pct occurence for each sentiment in a column dataframe
  occ <- as.data.frame(sentiments)
  n <-  nrow(df)
  l <- length(sentiments)
  occ$occurrences <- foreach(s= sentiments) %do% (nrow(df %>% select("sentiment") %>% filter(sentiment =
  return(occ)
}
text <- textpop %>% select(text)
text <- text %>% # Word COUNT
  unnest_tokens(word, text) %>%
  anti_join(stop_words, by=c("word"="word")) %>%
  inner_join(nrc) %>%
  count(word, sort = TRUE)
```

```
## Warning: Column `word` joining character vector and factor, coercing into
## character vector
```

```
## Joining, by = "word"
```

```
senttable <- get_sentimentss(text, nrc$sentiment)
pctsent <- get_pct_occur(senttable, nrc$sentiment)
pctsent <- pctsent[!duplicated(pctsent[,c('sentiments')]),]
pctsent <- as.data.frame(lapply(pctsent, unlist))
pctsent <- arrange(pctsent, desc(occurrences))
print(pctsent)
```

```
##       sentiments occurrences
## 1       negative   18.448439
## 2       positive   17.502365
## 3          trust   13.339640
## 4           fear   10.217597
## 5          anger    8.798486
## 6   anticipation    7.663198
## 7        sadness    7.190161
## 8        disgust    6.433302
## 9            joy    6.433302
## 10      surprise    3.973510
```

As we can see the most common sentiments are positive (17.5%) and negative (18.4%), meaning that most popular political tweets contain strongly positive or strongly negative words. The next most common were trust (13.3%) and fear (10.2%). Because the word president has a sentiment of trust, as do other words associated with leaders, the percentage this appears is high because the president is a very common topic of political discussion. What is interesting is that the political tweets with fearful language were more likely to receive retweets or favorites. The fifth most popular sentiment was anger at 8.7%, showing that political tweets often take an angery tone. These all indicate that our political discourse is often focused on strong negative (fear, anger, negative) and positive (trust, anticipation, postive) emotions, possibly to encourage people to react more emotionally in favor or against the tweet and spread the message to others.

We also want to determine which sentiments were most popular among this list of popular tweets.

```
get_sum_pop <- function(df, sentiments){
  #get df of pct occurence for each sentiment in a column dataframe
  occ <- as.data.frame(sentiments)
  n = nrow(df)
  l <- length(sentiments)
  occ$popularity <- foreach(s=sentiments) %do% sum(df %>% filter(sentiment == s) %>% select(popularity))
  return(occ)
}



tidypop <- textpop %>% # Word COUNT
  unnest_tokens(word, text) %>%
  anti_join(stop_words, by=c("word"="word")) %>%
  inner_join(nrc) %>% select(sentiment, popularity)
```

```
## Warning: Column `word` joining character vector and factor, coercing into
## character vector
```

```
## Joining, by = "word"
```

```
out <- get_sum_pop(tidypop, nrc$sentiment)
sentimentpop <- as.data.frame(lapply(out, unlist)) %>% distinct()
sentimentpop <- arrange(sentimentpop, desc(popularity))

print(sentimentpop)
```

```
##      sentiments popularity
## 1      positive   19403285
## 2      negative   18033061
## 3         trust   17076014
## 4          fear   10375874
## 5  anticipation    9551013
## 6         anger    9086309
## 7       sadness    8215041
## 8      surprise    8033919
## 9           joy    6683846
## 10      disgust    6630905
```

The top 6 haven't changed at all other than their order in this secondary analysis. This means that the distribution of the sentiments is fairly uniform amongst the popular tweets, regardless of how large the popularity is. There is a remarkable difference with the sentiment "surprise" though, as this sentiment appears no where near the percentage of the others, but tweets containing it tend to be the most popular of the popular tweets. This is likely due to the last name of the president having a sentiment of "surprise", and this reasoning is consistent with the increase in the word "president"'s sentiment of "anticipation".

Word Commonality

Here we determine which words appear most frequently in popular tweets

```
text <- arrange(text, desc(n))
print(text)
```

```
## # A tibble: 448 x 2
##    word          n
```

```
##    <chr>     <int>
##  1 vote        128
##  2 trump        65
##  3 president    56
##  4 lie          32
##  5 united       32
##  6 cash         30
##  7 collusion    30
##  8 white        28
##  9 court        24
## 10 attorney     20
## # ... with 438 more rows
```

As we can see the most common word is vote, second to president and trump, which often appear together. Next we see lie, which is often associated with lawyers and politicians. The next two words are associated with political corruption, with the court cased Citizens vs United, or United as in the United States. Cash is also associated with jobs and other important issues. We then see 3 words associated with the legal system, likely due to the current investigation into Donald Trump's campaign's collusion with the Russian Government in election fraud, and due to general high profile court cases. We also see the word "white" a significant number of times. This is indicative of how much race is still relevant in our current political discussion. It also appears higher than black or other races because it appears in the term "White House". Other very relevant words that occur at least 15 times are "shooting", "hoax", "war", "hero", and "deal". Which describe things like propaganda, the gun control debate, military action, and foreign policy.

Conclusion

A popular political tweet should be near the top of the character limit without a link. The subject of the tweet should be the previous election, and collusion with Russia, specifically naming Trump. The tweet should be angry, negative, and use fearful language. This is the profile of the tweets from the first week of May 2018.

This may be very volatile data, so my shiny application allows the user to compare the current data to this data from May 1st to May 7th 2018 to determine if any changes have occurred.

My methodology may be biased by the search terms I chose. Future experiments may be improved by a larger list that gets the tweets in batches to deal with the maximum queries possible with the API.