HW2: Text Analytics: Twitter Sentiment Analysis of #COVID_19

Marshall Tuck

11/14/2020

Background and Introduction

This paper represents work in the space of text analytics, sentiment analysis, and topic modeling to better understand the feelings of Twitter users towards the topic of COVID. I perform this by scraping Twitter by use of the hashtag #COVID_19, and then executing analysis to better understand this conversation.

Overall Research Process

The process of analyzing tweets in this assignment can be summarized into the following steps:

```
+ Twitter Provisioning
+ Data Gathering and Web Scraping via Twitter API
+ Data Cleansing
+ Sentiment Analysis
+ Text Analytics
+ Topic Modeling
```

Twitter Provisioning and API use

In order to gather the full Tweet corpus needed, access to the Twitter development API was needed. An application had to be submitted to developer.twitter.com explaining the request, and after a 3 day review access was provisioned. This included the provision of API keys and Tokens in order to authenticate via Oauth to the Twitter API.

Once authentication is established, the twitteR package is used to search for tweets using specific parameters, in our case, those with #COVID_19. In the analysis within this assignment, 5000 tweets are returned (excluding retweets to avoid double counting) with a parameter max date of 01-01-2020. The returned tweets are then vectorized for easier processing.

Basic Data Cleansing

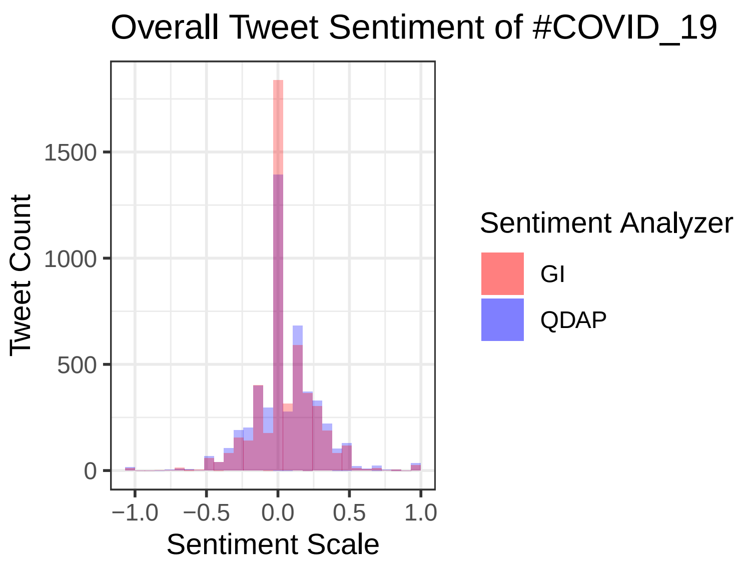
Our initial cleansing includes the following steps - convert all tweets to lowercase, remove emojis, remove Twitter Handles ("@JohnDoe"), remove all hashtags ("#xyz"), and remove links embedded within tweets.

Sentiment Analysis

Next, sentiment analysis is applied using the SentimentAnalysis package. This function returns the overall sentiment of each document in the corpus (ie. tweet in the data).

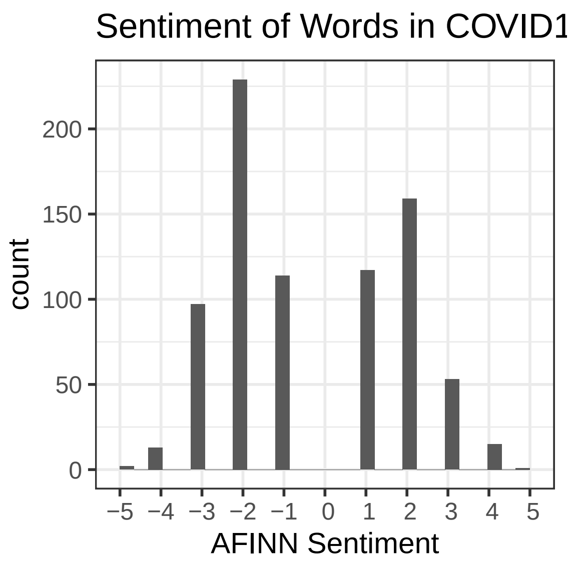
In the presented histogram below, we show the overlapped QDAP and GI sentiments, which each show fairly similar results.

Sentiment on the #COVID_19 hashtag trends fairly neutral, which may be best interpreted as being informative and news-based, and not fear- or emotion-based.



In the graph below, we show the sentiment of the words in our corpus, found by returning the sentiments within the “afinn” package. We see a positive shift of the returned words in our corpus, which is interesting compared to the average neutral sentiment of the tweets.

This is an interesting finding - that the words used in the tweets are more positive and negative than the tweets themselves. One explanation, neutral sentiment tweets may have a mix of positive sentiment and negative sentiment words.

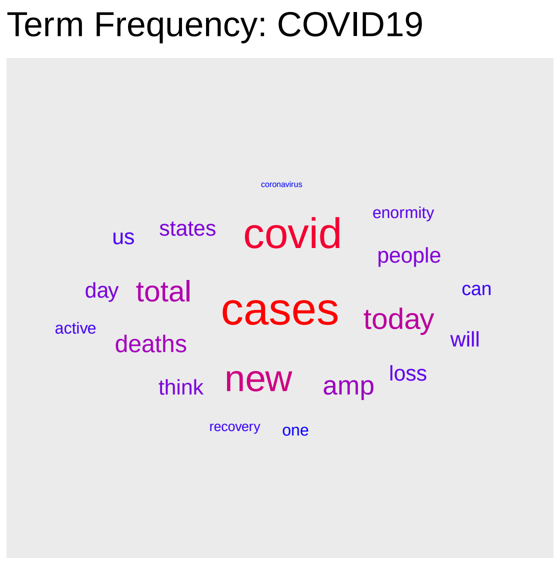


Text Analytics

Term Frequencies

Our first approach in text analytics will be to understand which words in our corpus are used the most. The word cloud is a visual representation of the 1 grams (ie. term frequencies) within our corpus. Common words are “covid”, “cases”, and “new”.

Word Cloud of 1 grams



2 grams and 3 grams

Next we present our most common 2 grams and 3 grams.

ngrams	freq	prop
new cases	211	0.0054359
think enormity	175	0.0045085
enormity loss	175	0.0045085
loss states	175	0.0045085
active cases	146	0.0037613

ngrams	freq	prop
think enormity loss	175	0.0045086
enormity loss states	175	0.0045086
new cases today	125	0.0032204
total active cases	121	0.0031174
today recovery today	120	0.0030916

Each of the ngram presentations show a bit of a different story - 1 and 2grams grams show “Cases” being the most common focus word. 3grams appear show a higher focus on “enormous loss” combinations and tragedy-type words.

TF-IDF

TF-IDF is a way to identify words within the corpus that are a combination of 1) important within the tweet and 2) rare within the document; TF stands for term frequency, and IDF is inverse document-frequency.

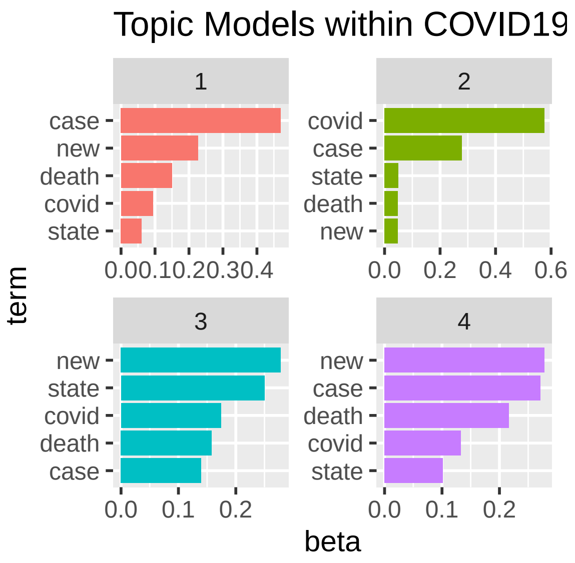
This offers insight into which words are used most powerfully - if a word is used in a single tweet once or twice, but it is not used in very many tweets in the corpus, it receives a high TF-IDF score. The words with the highest TF-IDF scores are below - note that a gibberish tweet only containing a single rare word would receive a high TF-IDF score. In our sense, "grid", "resolution", and "gripe" can be thought of as the Most Valuable Words in a tweet.

tweet_num	word	count	total_words	tf	idf	tf_idf
2	terminate	1	1	1	8.517193	8.517193
236	grid	1	1	1	8.517193	8.517193
456	resolution	1	1	1	8.517193	8.517193
473	gripe	1	1	1	8.517193	8.517193
621	awesomest	1	1	1	8.517193	8.517193

Topic Modeling

Topic modeling is our next analysis, and we a priori select a topic number = 4. This appears to be reasonable, because with a minimum sparsity of 95% (words must be in at least 5% of all tweets), our doc-term matrix is significantly reduced. Of note, there is slight overlap of words within each topic.

Next, we show the word topic matrix. This is an indicator of the overall probability of a word belonging to a topic. This is called the word-topic matrix, and is represented in the colorful bar charts below. The probabilities are conditional probabilities - the probability of the appearance of a given word, conditional on that topic.



Summary

In summary, our process to understand the sentiment and feelings of Twitter users to COVID19 highlighted some interesting results:

One, the sentiment of individual tweets in the corpus, as shown in a histogram, was neutral - however, the sentiment of the words within those tweets was much more extreme, either positive or negative.

Two, by reducing sparseness in our DTM and by also selecting 4 topics within our topic model, we reveal conditional probabilities of 20% or higher within each topic. This probability is meaningful in our sense, and is an indicator of true behavior of words belong to topics. In comparison, a conditional probability of 1% of a word to topic may indicate noise instead of true signal.

Third and last, the four topics found in the topic model can be thought of as:

```
+ Topic 1 - State Summaries of Covid Cases
+ Topic 2 - New Covid Cases
+ Topic 3 - State Covid Cases and Deaths
+ Topic 4 - New Deaths from Cases
```

Code Appendix

```r
##########Libraries
library(twitteR)
library(rtweet)
library(purrr)
library(dplyr)
library(stringr)
library(textstem)
library(ngram)
library(hunspell)
library(tm)
library(sentimentr)
library(SentimentAnalysis)
library(tidytext)
library(textdata)
library(dplyr)
library(XML)
library(xml2)
library(rvest)
library(topicmodels)
library(ggplot2)
library(base64enc)
library(openssl)
library(dplyr)
library(ggplot2)
library(kableExtra)
library(Rmpfr)
library(digest)
library(ldatuning)
library(ggwordcloud)

############Twitter Setup
#Values found from developer page within Twitter
# App Name within Twitter
appname <- "covid_hw2"

## api key
key <- "veCnbnKzSzhmoeZfsZLLsgsRw"

## api secret key
secret <- "Mp7SGHqZlxzrJiqmzNe5Yvv3BvYlIa5IeO2LvIabfjD0pLFjc2"

#Access token
access_token<- "1326954916729004036-ZcnGYsZrUbqHcVymBywtQlbheXhjzf"

#Access Secret Token
access_secret<- "HcVK1EjHOD5UieJ1ib0KS4mxKe1m1lilPiTfuwTSkCvgK"

#Authenticate using OAuth handshake
options(httr_oauth_cache=TRUE)
setup_twitter_oauth(consumer_key = key, consumer_secret = secret,
                    access_token = access_token, access_secret = access_secret)
```

```r
#Return tweets marked with #COVID_19 hashtag
covid<- searchTwitter("#COVID_19 -filter:retweets",n=10, retryOnRateLimit=120, lang="en",
              since="2020-01-01", resultType="recent")

#######Data Wrangling
#Get only the text
covid<- sapply(covid, FUN=function(x) gettext(x$text))

#Make lower case
covid.processed<- tolower(covid)

#Remove emojis
covid.processed<- gsub("[^\x01-\x7F]", "", covid.processed)

#Remove @handles
covid.processed<- gsub("@\\w+", "", covid.processed)

#Remove #hashtags
covid.processed<- gsub("#\\w+", "", covid.processed)

#Remove twitter links
covid.processed<- gsub("https://t.co/\\w+", "", covid.processed)

#########Sentiment Analysis
#Conduct Sentiment Analysis on the corpus
covid.sa<- analyzeSentiment(covid.processed)

##Ggplot histogram of sentiment of each tweet within corpus
ggplot(covid.sa)+
  geom_histogram(aes(x=SentimentGI, fill="red"), alpha=.3)+
  geom_histogram(aes(x=SentimentQDAP, fill="blue"), alpha=.3)+
  labs(x="Sentiment Scale", y="Tweet Count", title="Overall Tweet Sentiment of #COVID_19")+
  theme_bw()+
  scale_fill_manual(name="Sentiment Analyzer",values=c("red","blue"),labels=c("GI","QDAP"))


#########Data Wrangling
#Remove single characters
covid.processed<- gsub(" *\\b[[:alpha:]]{1}\\b *", "", covid.processed)

#Remove non-word characters
covid.processed <- gsub("\\W", " ", covid.processed)

#Remove number characters
covid.processed<- gsub("\\d"," ",covid.processed)

#Remove non-english words
covid.processed <- gsub("[^a-zA-Z]", " ",covid.processed)

#Replace multiple spaces with single space
covid.processed<-stripWhitespace(covid.processed)

#Remove leading and trailing white spaces
```

```
covid.processed<-str_squish(covid.processed)

#Split each string of words into a list of separate words
covid.token<- str_split(covid.processed, " ")


###########Tokenize
#Get Unique tokens
unique.covid.token<- unique(unlist(covid.token))

#Count number of times each word appears in tweet corpus
sum_words <- sapply(1:length(unique.covid.token), function(i){sum(unlist(covid.token) == unique.covid.t

#Summarize freq of each word in corpus
word_summary <- data.frame("word" = unique.covid.token, "count" = sum_words)%>%
  arrange(-count)

############Sentiment of Tokens
# get AFINN sentiments
sentiment.df<- word_summary %>%
  merge(by="word", get_sentiments("afinn"))

#Return ggplot of count of words used by sentiment
ggplot(sentiment.df, aes(x=value))+
  geom_histogram()+
  theme_bw()+
  ggtitle("Sentiment of Words in COVID19 Tweet Corpus")+
  labs(x="AFINN sentiment")+
  scale_x_discrete(name ="AFINN Sentiment",
                   limits=c(-5:5))

#Remove stop Words
covid.processed<- removeWords(covid.processed, stopwords())

#Replace multiple spaces with single space
covid.processed<-stripWhitespace(covid.processed)

#Remove leading and trailing white spaces
covid.processed<-str_squish(covid.processed)

#Split each string of words into a list of separate words
covid.token<- str_split(covid.processed, " ")

#Combine all words from all strings into one vector
covid.words<- unlist(covid.token)

#Find unique words (minus stop words) in covid.tweets
covid.words.unique<- unique(covid.words)

#Create output vector where each of words is counted in all tweets
tf.covid.words<- sapply(1:length(covid.words.unique),
                            function (i){
                              sum(covid.words==covid.words.unique[i])
                              }
                            )

#Present table where each unique word is summarized with count and proportion
tf.covid.words.table <-
  data.frame("word" = covid.words.unique, "freq" = tf.covid.words)%>%
  as_tibble()%>%
  mutate(prop=freq/length(covid.words))%>%
  arrange(-freq)%>%
  head(20)%>%
  kable(caption="#COVID_19 Tweet Words")%>%
  kable_styling(latex_options = "HOLD_position")

#Output count and prop table of terms
tf.covid.words.table

#Create data to put into word cloud of 1gram
covid.tf.wordcloud<-
  data.frame("word" = covid.words.unique, "freq" = tf.covid.words)%>%
  as_tibble()%>%
  mutate(prop=freq/length(covid.words))%>%
  arrange(-freq)%>%
  head(20)

#Output word cloud of 1gram
ggplot(covid.tf.wordcloud,
       aes(label = word, size = freq, color = freq)) +
            geom_text_wordcloud_area() +
            scale_color_gradient(low = "blue", high = "red")+
            labs(title = "Term Frequency: COVID19 Word Cloud")

#Create one string of all covid words
all.covid.words<- concatenate(covid.words)

#Present in ngram of 2
two.ngram.covid <- ngram(all.covid.words, 2, sep=" ")

#Output count and proportion
kable(head(get.phrasetable(two.ngram.covid), 20), caption="#COVID_19 2 Grams")%>%
  kable_styling(latex_options = "HOLD_position")

#Present in ngram of 2
three.ngram.covid <- ngram(all.covid.words, 3, sep=" ")

#Output count and proportion
kable(head(get.phrasetable(three.ngram.covid), 20), caption="#COVID_19 3 Grams")%>%
  kable_styling(latex_options = "HOLD_position")

tf_document <- lapply(1:length(covid.token),
       function(i){
         unique_words <- unique(covid.token[[i]])

         tf_review <- sapply(1:length(unique_words),
```

```r
                              function(j){sum(covid.token[[i]] ==
                                                unique_words[j])})

        tf_single_review <- (data.frame("tweet_num" = rep(i, length(tf_review)),
                                        "word" = unique_words,
                                        "count" = tf_review,
                                        "total_words" =
                                          rep(length(covid.token[[i]]),
                                              length(tf_review))))
      })

#Bind the outputs of the function
tf_by_covid.tweet<- do.call("rbind", tf_document)

#Make table combining term freq, word, review number, and count
tfidf_covid <- bind_tf_idf(as_tibble(tf_by_covid.tweet), word, tweet_num, count)

#Present top 20 tf-idf scores
tfidf_covid%>%
  arrange(-tf_idf)%>%
  head(20)%>%
  kable(caption="COVID_19 Words Top 20 TF-IDF")%>%
  kable_styling(latex_options = "HOLD_position")

#Stem Words
covid.stem.token<- lapply(covid.token, stem_words)

#Check for blanks
which(covid.stem.token == c(""," "))

#Remove empty cell
#covid.stem.token<- covid.stem.token[-45]

#Create DTM
DTMcovid_stemmed <- DocumentTermMatrix(VCorpus(VectorSource(covid.stem.token)))

#Remove sparse terms where words must be in at least 5% of tweets
DTMcovid_stemmed<- removeSparseTerms(DTMcovid_stemmed, sparse = .95)

#Some empty tweets still exist, remove using the next two lines of code
raw.sum<- apply(DTMcovid_stemmed,1,FUN=sum)
DTMcovid_stemmed<- DTMcovid_stemmed[raw.sum!=0,]

#Create topic Model
LDcovid_stemmed <- LDA(DTMcovid_stemmed, k=4)

#Extract Word Topic Probabilities
topicmod_covid<- tidy(LDcovid_stemmed, matrix="beta")

#Present highest probability words within each topic
topic.mod.prob<- topicmod_covid %>%
  group_by(topic) %>%
  arrange(topic, -beta)%>%
  top_n(5)

topic.mod.prob%>%
  kable(caption="Top 5 Words by probability within each topic")%>%
  kable_styling(latex_options = "HOLD_position")

#Present Word Topic Matrix
topic.mod.prob %>%
  mutate(term = reorder_within(term, beta, topic)) %>%
  ggplot(aes(term, beta, fill = factor(topic))) +
  ggtitle("Topic Models within COVID19 Tweet Corpus")+
  geom_col(show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free") +
  coord_flip() +
  scale_x_reordered()
```