Milestone 2 - Twitter Sensitivity Analysis

Jordan Nottage

1. Goal

The main goal/objective of my project is to analyse the sentiment of various tweets from Twitter, and observe trends in the tweets' sentiments (by using more tweet attributes such as location and time). I decided to focus on tweets around the topic of Corona Virus.

2. Data Source

The main source of data came from Twitter using the package `rtweet`. This required me to create a developers account on Twitter and obtain a token key to access the data. The dataframe returned from Twitter is not very good for obtaining the location of a tweet - this is because very few people allow this feature, or they can put a custom location (e.g. "NZ" or "KiwiLand" etc) into their profile which is very messy.

As a result, I decided to search for tweets by including a country name as a keyword to obtain tweets on corona virus as well as the specified country. Overall, I obtained roughly 2000 tweets for each of the 6 countries: New Zealand, USA, Australia, Canada, UK/England and China. Note this does not mean the tweet came from the same country, but it will allow me to analyse the sentiment of a tweet involving that country. The data gethering code can be seen in Appendix 2.1.

Furthermore, the word sentiments are gathered from the *get_sentiment* function. I decided to use both the AFFIN (scores words from -5 to 5) and Bing (scores words as negative, neutral or positive) lexicons at this stage to compare as I go. This can be seen in Appendix 2.2.

3. Data Processing

The main form of data tidying required is to reduce the number of columuns. As seen below, there are initially 90 attributes, a lot of which are not going to be of any use to me. So the first step is to remove irrelevant columns.

```r
library(tidyverse)
library(rtweet)

load("sample.RData")
length(names(sample_data)) # number of col names of a twitter dataframe
```

```
## [1] 90
```

By selecting some useful parameters, I reduced the columns to relevant ones only and added a new column to each country's dataframe called 'specified_country' - this is the country I have specifically searched for tweets on. Then I join each country's dataset into one. The new column names can be seen below, and the code is in Appendix 3.1.

```r
load("allDFs.Rdata")
names(coronatweets.df) # potentially useful column names
```

```
##  [1] "screen_name"        "text"               "created_at"
##  [4] "source"             "specified_country"  "location"
##  [7] "favorite_count"     "retweet_count"      "followers_count"
## [10] "statuses_count"     "verified"           "account_created_at"
## [13] "description"        "tidytext"
```

Next, I combine each of the individual country datasets into one dataframe - although the individual sets may still be used for independent analysis. This can be seen in Appendix 3.1

Finally, I also tidy the text in the next section below (also in Appendix 3.1) by removing unnecessary elements, such as http strings (weblinks), punctuation and symbols, new line characters and I convert to lower case (in preparation for future annalysis/calculation of sentiment). An example of an original tweet text, and the cleaned text is output below the code segment.

```
coronatweets.df$text[14] # original text
```

```
## [1] "Coronavirus: taking Parliament onto the holodeck https://t.co/RUCI0aAT0J"
```

```
coronatweets.df$tidytext[14]
```

```
## [1] "coronavirus taking parliament onto the holodeck "
```

4. Data Exploration

I decided not to explore the time of the posted tweet because the dataset does not give the local time, and due to inadequate specific location data, I am unable to convert the universal time into local time.

Hence, my main exploration consists of glimpsing the top words wihtin each country to gain insight as to how the words and sentiments may vary between countries (note that calculating actual sentiment scores/values will be part of the analysis section in the next milestone).

```
knitr::kable(head(nz_words, 7))
```

word	n
coronavirus	930
covid	783
zealand	758
australia	339
nz	274
vaccine	240
lockdown	198

```
knitr::kable(head(us_words, 7))
```

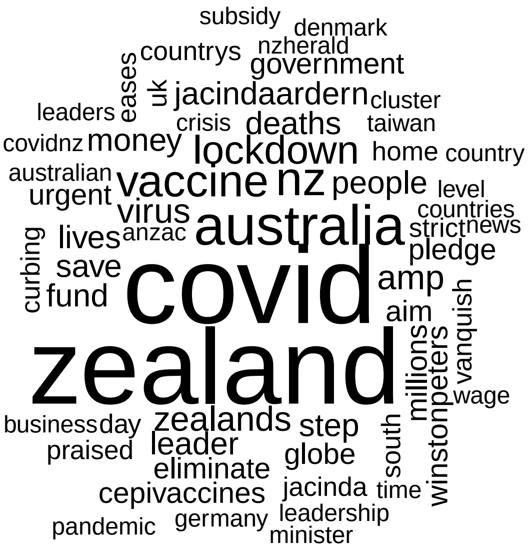
word	n
coronavirus	993
covid	727
usa	677
america	613
deaths	320
united	312
trump	273

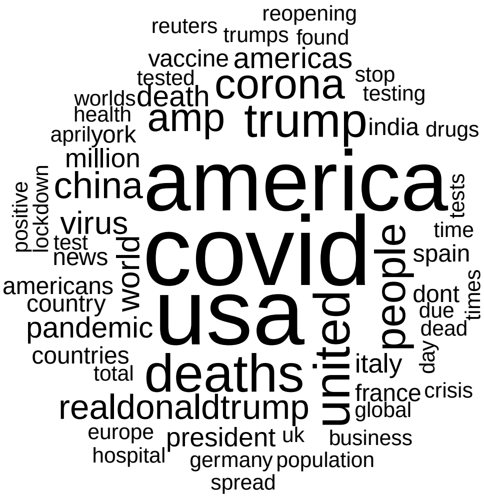
Above we can see the top 10 used words in the NZ and US set of tweets respectively. It is interesting to see words like "deaths" (giving negative conotations) as the 5th most used word in the Corona-based tweets for USA, and "vaccine" (possibly hopeful conotations) in the NZ tweets.

As seen below, I have used a word cloud to represent the words used in each country's set of tweets. Below are the word clouds generated for NZ and USA, for example. These word clouds give an idea of the most common words used across the tweets from a given country - bare in mind some of these popular words will likely be the keywords I used to generate the data (such as "corona" and the name of the country). This helps me explore different sentiments (just by reading the words) between countries, as well as on the given topic of coronavirus. In fact, we can already see some contrasting sentiments across NZ and USA words, such as "leader", "praised", "money", "save" in NZ versus "deaths" and "dead" appearing more frequently in USA - although NZ also contains "death" there still seems to be a contrasting sentiment.

```
par(mfrow=c(1,2))
par(mar=c(0.5, 0.5, 0.5, 0.5))
```

```r
wordcloud(words=nz_words$word, freq=nz_words$n, min.freq = 20, random.order = FALSE, max.words = 60)
wordcloud(words=us_words$word, freq=us_words$n, min.freq = 20, random.order = FALSE, max.words = 60)
```





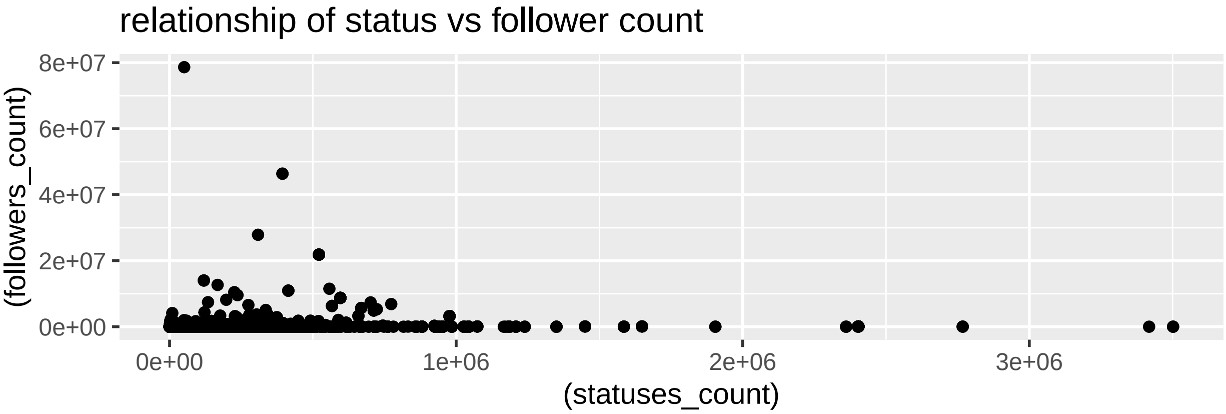
I also decided to look at some other attributes and how they interact. Below is the code and plots of *statuses_count* (the number of statuses a user has made) against *followers_count* (the number of followers a user has).

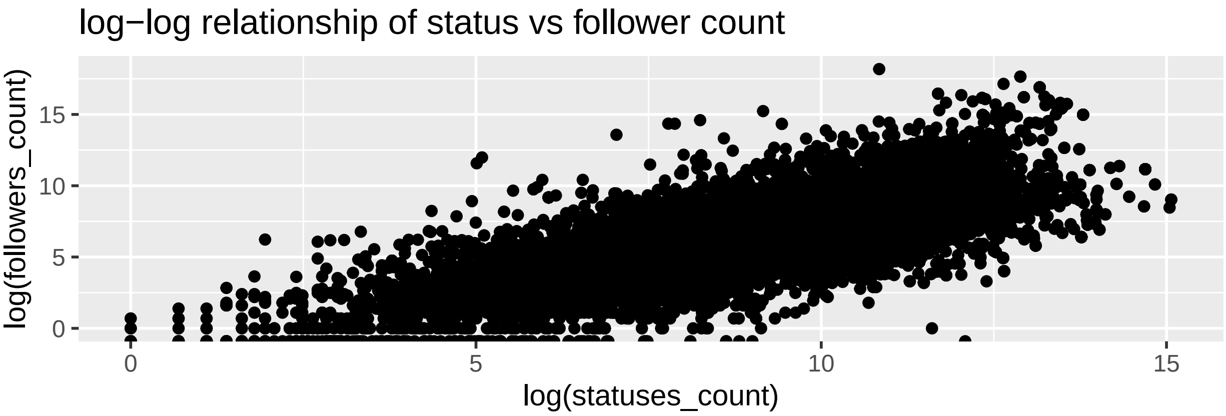
```r
library(ggplot2)
library(gridExtra)

plot1 <- coronatweets.df %>%
  ggplot(aes(x=(statuses_count), y=(followers_count))) +
  geom_point() + labs(title="relationship of status vs follower count")

plot2 <- coronatweets.df %>%
  ggplot(aes(x=log(statuses_count), y=log(followers_count))) +
  geom_point() + labs(title="log-log relationship of status vs follower count")

grid.arrange(plot1,plot2)
```


relationship of status vs follower count


log−log relationship of status vs follower count

From the graphs above we can see that it is initially difficult to see a relationship between these two attributes, however on the log-log model we can see the linearly increasing relationship. It will be interesting to see in future analysis if these attributes are important when determining the sentiment of a tweet.

5. Analytical Plan

The next step is to apply analytical methods. This will initially involve me calculating the sentiment of the tweets now that I have observed and explored the data. To do this I will need to use a sentiment package that I can compare the twitter text data to. This will score each word inside a tweet, and then calculate the total and/or average score of that tweet. The score will either be a number or a category of negative/positive (as outlined in Milestone 1) or even both for comparison.

Then I could perhaps build models to see what attributes may have an impact on the sentiment of a tweet. For example do the number of followers a user has play a part in the sentiment? This could be done graphically as well as by building models to see how the sentiment may change with varying attributes. I can achieve this with various regression models or I can even try to use a model selection tecnique such as cross validation or ridge/lasso model selection to observe what attributes are important in the analysis.

Potentially useful attributes to explore could be the number of followers and number of statuses a user has, as well as whether or not a user is verified, or perhaps how many likes they have. These are all attributes that I can use to model the sentiment or see if they affect the sentiment of a tweet.

Appendices

2.1 Obtaining the Twitter data based on Corona Virus for 6 countries:

```r
library(rtweet)
library(tidytext)
library(textdata)
library(tidyverse)

nz_corona <- search_tweets(
  q = "corona OR coronavirus OR covid19 nz OR zealand", # query (key words) - nz
  n = 2000, # number of tweets to return
  type = "mixed", # "recent" (default), "mixed", "popular" etc. tweets
  include_rts = FALSE, # include retweets
  parse = TRUE, # gets data as tidier df
  lang = "en" # limit tweets to english
)

us_corona <- search_tweets(
  q = "corona OR coronavirus OR covid19 usa OR \"united states\" OR america", # usa
  n = 2000,
  type = "mixed",
  include_rts = FALSE,
  parse = TRUE,
  lang = "en"
)

uk_corona <- search_tweets(
  q = "corona OR coronavirus OR covid19 uk OR \"united kingdom\" OR england", # uk
  n = 2000,
  type = "mixed",
  include_rts = FALSE,
  parse = TRUE,
  lang = "en",
)

aus_corona <- search_tweets(
  q = "corona OR coronavirus OR covid19 aus OR australia OR aussie", # australia
  n = 2000,
  type = "mixed",
  include_rts = FALSE,
  parse = TRUE,
  lang = "en"
)

ca_corona <- search_tweets(
  q = "corona OR coronavirus OR covid19 ca OR canada", # canada
  type = "mixed",
  include_rts = FALSE,
  parse = TRUE,
  lang = "en"
)

ch_corona <- search_tweets(
  q = "corona OR coronavirus OR covid19 china OR ch OR ROC", # china
  type = "mixed",
  include_rts = FALSE, s
  parse = TRUE,
  lang = "en"
)
```

2.2 Obtaining sentiment data:

```
sentiment_affin <- get_sentiments("afinn") # afinn scores words from -5 to 5 (negative to positive)
sentiment_bing <- get_sentiments("bing") # bing scores words as negative, neutral or positive
```

3.1 Reducing columns and tidying data and text:

```
keep_cols <- as.vector(c("screen_name","text","created_at","source", "specified_country",
                         "location","favorite_count","retweet_count",
                         "followers_count","statuses_count","verified",
                         "account_created_at",'description'))
nz_corona <- nz_corona %>% mutate(specified_country = "NZ") %>% select(keep_cols)
us_corona <- us_corona %>% mutate(specified_country = "USA") %>% select(keep_cols)
uk_corona <- uk_corona %>% mutate(specified_country = "UK") %>% select(keep_cols)
aus_corona <- aus_corona %>% mutate(specified_country = "Aus") %>% select(keep_cols)
ca_corona <- ca_corona %>% mutate(specified_country = "Canada") %>% select(keep_cols)
ch_corona <- ch_corona %>% mutate(specified_country = "China") %>% select(keep_cols)

# merge the data on the kept columns for all countries
corona.df <- full_join(nz_corona, us_corona, by=keep_cols)
corona.df <- full_join(corona.df, uk_corona, by=keep_cols)
corona.df <- full_join(corona.df, aus_corona, by=keep_cols)
corona.df <- full_join(corona.df, ch_corona, by=keep_cols)
coronatweets.df <- full_join(corona.df, ca_corona, by=keep_cols) # full dataset
```

Tidying the text attribute:

```
# remove http elements, punctuation and new lines etc, convert to lower case
coronatweets.df$tidytext <- gsub("https\\S+", "", coronatweets.df$text) # new col of tidy text
coronatweets.df$tidytext <- gsub("[^[:alpha:][:space:]]*", "", coronatweets.df$tidytext)
coronatweets.df$tidytext <- gsub("[\r\n]", "", coronatweets.df$tidytext)
coronatweets.df$tidytext <- tolower(coronatweets.df$tidytext)

coronatweets.df$text[1]
coronatweets.df$tidytext[1]

save(nz_corona, us_corona, uk_corona, aus_corona, ca_corona, ch_corona, coronatweets.df, file="allDFs.Rdata")
```

4.1 Generating the word clouds:

This was achieved by cleaning the text, and then creating a word frequency dataframe for each country set.

```
library(wordcloud)
library(stopwords)
library(tidytext)


gen_wordcloud <- function(tweets.df){
  # remove http elements, punctuation, and \n chars
  tweets.df$text <- gsub("https\\S+", "", tweets.df$text)
  tweets.df$text <- gsub("[^[:alpha:][:space:]]*", "", tweets.df$text)
  tweets.df$text <- gsub("[\r\n]", "", tweets.df$text)

  tweettext <- tweets.df %>%
    select(text) %>%
    unnest_tokens(word, text)# separate all words in the df

  wordcount <- tweettext %>% anti_join(stop_words) # remove common english words
  wordcount <- wordcount %>% count(word, sort=TRUE) # count and sort word freq in order

  return(wordcount)
```

```r
}

nz_words <- gen_wordcloud(nz_corona)
us_words <- gen_wordcloud(us_corona)
uk_words <- gen_wordcloud(uk_corona)
ca_words <- gen_wordcloud(ca_corona)
aus_words <- gen_wordcloud(aus_corona)
ch_words <- gen_wordcloud(ch_corona)

head(nz_words,10)
head(us_words,10)

wordcloud(words=nz_words$word, freq=nz_words$n, min.freq = 20, random.order = FALSE)
wordcloud(words=us_words$word, freq=us_words$n, min.freq = 20, random.order = FALSE)
```

EOF