Milestone 3 - Twitter Sensitivity Analysis

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, I also need to use sentiment data. Initially in milestone 2, I decided to use the *get_sentiment* along with some downloaded lexicon packages. However, I have since found a better package, namely `sentimentr`, which calculates the sentiment of text in a smarter way. This will be further discussed in the analytical plans section.

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.

```
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.

```
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"           "wordcount"
## [16] "av_sentiment"
```

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 (main code in Appendix 3.1) by removing unnecessary elements, such as http strings (weblinks), new line characters and I convert to lower case for simplicity. Note that I originally (in milestone 2) removed punctuation as well to tidy the text. I have decided to keep punctuation in for now, the reasons for

which will become clear in the analysis section as this impacts the sentiment analysis.
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).

```
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))
```

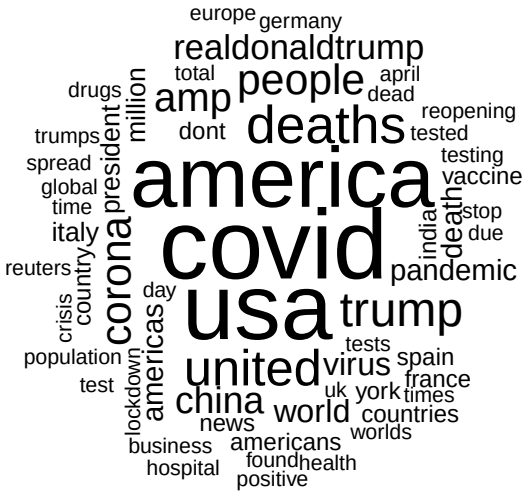
word	n
coronavirus	993
covid	727
usa	677
america	613
deaths	320
united	312
trump	273

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))
```

```
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)
```



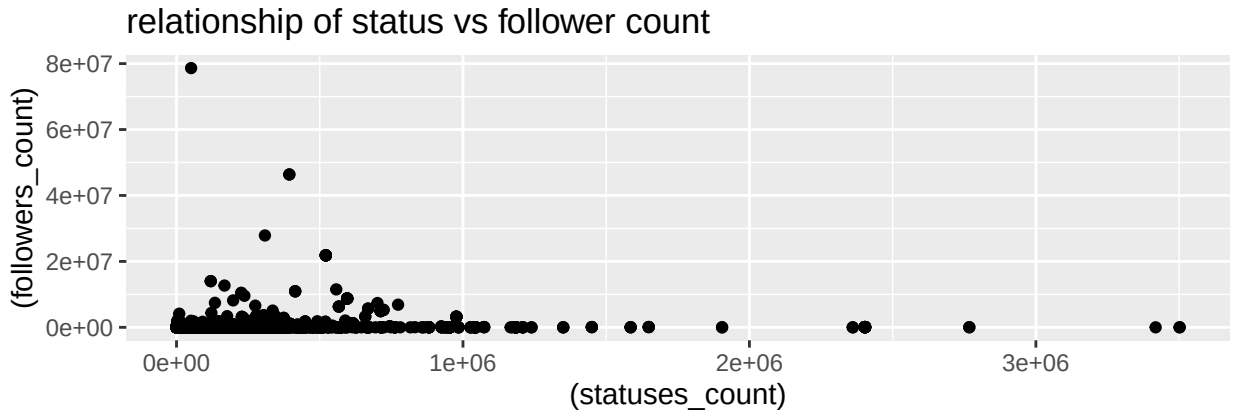
I also decided to look at some other attributes and how they interact. Below is the code and plots of *statuses_count* (the number of statuses a user has made) against *followers_count* (the number of followers a user has).

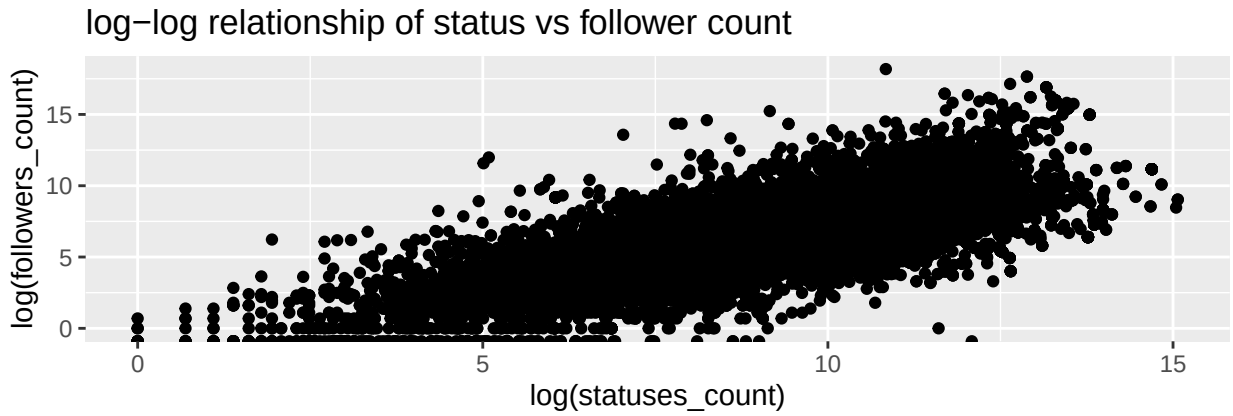
```
library(ggplot2)
library(gridExtra)

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)
```





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. As mentioned above, I will be using the `sentimentr` package and the corresponding function `sentiment_by` to observe the average sentiment of each tweet.
The `sentiment_by` function calculates the average sentiment, word count and some other attributes of each text input. I have chosen this analytical tool for the sentiment because it uses a standard dictionary lookup (where positive and negative words are scored +1 and -1 repectively), but also takes *valence shifters* into account. Valence shifters are words in text that may alter the overall sentiment of a word - e.g. the word "really" in "I **really** like it" amplifies the sentiment of the word "like". Alternatively, they can flip the sentiment entirely - e.g. "I do **not** like it". This makes calculating the sentiment much more accurate compared to treating every word independently, and avoids misinterpretation.

The function breaks the text into sentences/clauses and can also deal with punctuation. Furthermore, removing punctuation manually beforehand could lead to the function miscalculating the sentiment. For example consider the text: "Not really. Good thing I don't care". The function calculates the sentiment of each sentence seperately and gives an average sentiment. However, if I remove punctuation, it will read "Not really Good..." all at once, which we can see is an example of a valence shifter (as "not really" changes the sentiment of "good"). This is why my tidy text has kept its punctuation.

In this part of the analysis, I add columns to each tweet object including the text word count and average sentiment (as outputs from `sentiment_by`) and I also add a discretised sentiment score myself which is positive, neutral or negative based on the average sentiment score.
An output table of positivity/negativity can be seen below the following code for each country:

```
library(sentimentr)

sentiments <- coronatweets.df$tidytext %>%
  get_sentences() %>% sentiment_by()
```

```r
coronatweets.df <- coronatweets.df %>%
  mutate(wordcount = sentiments$word_count) %>% # word count in tweet
  mutate(av_sentiment = sentiments$ave_sentiment) %>%  # sentiment av value
  mutate(dscrt_sentiment = ifelse(av_sentiment>0, "Positive", # discrete sentiment
                                  ifelse(av_sentiment<0, "Negative", "Neutral")))

tab <- table(coronatweets.df$dscrt_sentiment, coronatweets.df$specified_country)
tab <- rbind(tab, Total=colSums(tab))
tab <- cbind(tab, Total=rowSums(tab))
tab
```

```
##           Aus Canada China   NZ   UK  USA Total
## Negative  982    898  1163  673  845 1010  5571
## Neutral   167    180   130  109  266  146   998
## Positive  849    815   704 1217  889  842  5316
## Total    1998   1893  1997 1999 2000 1998 11885
```

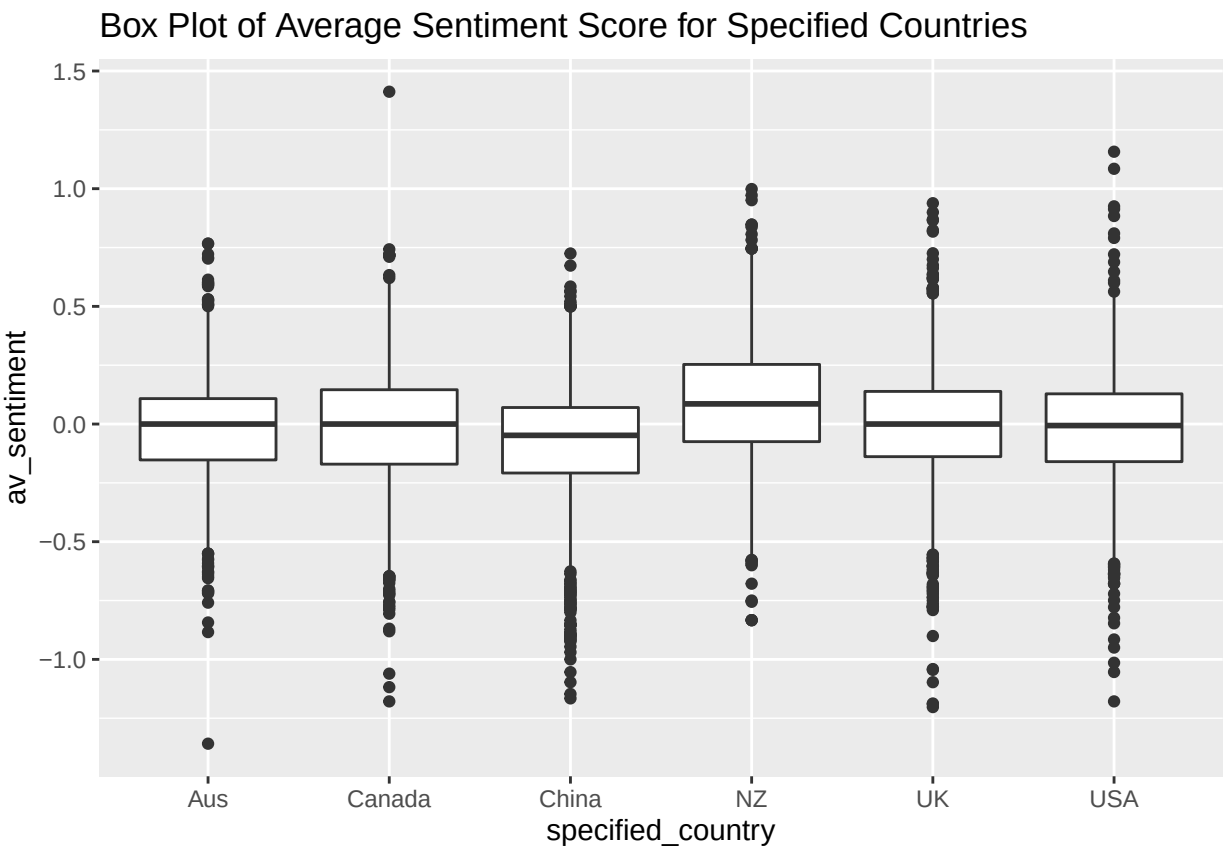
It is interesting to see from the table that NZ has the most amount of positive tweets, and the least amount of negative tweets (based on the calculations), while China and the USA both have a considerably larger number of negative tweets. I personally think that this seems plausible, given how NZ has been praised for their response, while the virus began in China, and USA has the highest death toll.

To add to the analysis between countries, the box plot below illustrates the differences in sentiments.

```r
library(ggplot2)

# Load data and convert specified_country to a factor variable
coronatweets.df$specified_country <- as.factor(coronatweets.df$specified_country)

# Box plot
plot <- ggplot(coronatweets.df, aes(x = specified_country, y = av_sentiment)) +
  geom_boxplot() +
  ggtitle("Box Plot of Average Sentiment Score for Specified Countries")
plot
```



NZ related tweets can be seen to have the most positive (highest average sentiment) tweets on average, while China related tweets appear the most negative on average. There is a larger spread of sentiment scores seen for USA related tweets, relative to other countries like NZ. This could imply more controversy in this country than in others.

For more future analysis, 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.

I also think the use of trees and/or random forests could be a valuable analysis method to try next as well. In particular I could use these to predict the discrete sentiment value (positive, neutral or negative), to see what attributes are important when predicting the sentiment.

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.

Discussion

The data I have available has some limitations, which could impact my results. Firstly, the text attribute of a tweet is limited to a relatively small amount of characters. This makes it harder to perform analysis on because the average sentiment calculation does not have much input text to work with, so any errors or misinterpreted sentiments impact the average sentiment more. Furthermore, if there were more words available in the text, I could count the frequency of words used in a tweet, similarly to the message spam example used in class. This would allow me to see what words play an important role in the overall sentiment and use these for predictions.

Another limitation when calculating the sentiment of a tweet is the (current) inability to analyse emojis/emoticons. These can often strongly indicate, or add to, the sentiment of a message - especially when there are limited characters already. At this stage, I have been unable to make use of any emoticons in the text, however in the future the emojis could be a strength of the text data by contributing to the sentiment score.

The data also contains a mixture of different types/structures of attributes such as discrete and continuous, as well as string and numeric type elements. This may make it more difficult to create the model (perhaps just due to my level of

understanding in R).

Despite the mix of attribute types, a plus is that there is a large number of attributes to observe. Although I am yet to analyse which attributes are important, it gives me a range of different variables that I can analyse with respect to the various tweet sentiments.

Another strength of my data that I have fonud very useful so far is that I have a (roughly) equal amount of data for each of the 6 countries chosen. This is obviously because I searched for data using the relevant key words, nonetheless, it means that I can easily compare results or summaries between countries.

Appendices

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

```
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"
)
```

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) # http elements
# coronatweets.df$tidytext <- gsub("[^[:alpha:][:space:]]*", "", coronatweets.df$tidytext) # punctuation
coronatweets.df$tidytext <- gsub("[\r\n]", "", coronatweets.df$tidytext) # newline chars
coronatweets.df$tidytext <- tolower(coronatweets.df$tidytext) # lower case

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)
}

nz_words <- gen_wordcloud(nz_corona)
us_words <- gen_wordcloud(us_corona)
uk_words <- gen_wordcloud(uk_corona)
```

```r
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