FIT5147 Data Exploration Project

Isobel Rowe, 30042585

Table of Contents

1. Introduction

Twitter, a microblogging platform that has over 321 million monthly active users around the world (Shaban, 2019), has played an incredibly important role in Donald Trump's election campaign, and beyond. Since 2009, Donald Trump's Twitter account, @realDonaldTrump, has 'tweeted' more than 41,000 times, and racked up a follower count of almost 60 million people. Yet, despite calls that his Twitter-use is inflammatory or even dangerous, the fast-paced, 140-character (now 280) limit platform quickly became his preferred method of communication throughout his Presidency. It has given him a platform to instantly express his views, opinions and beliefs and is a window into not only his thoughts and psyche, but into the kind of messages he wants to communicate to his supporters, and the world at large.

This report aims to answer the questions:

<u>Q1) How do favourites/retweets/replies correlate to Donald Trump's popularity?</u>

An important political tool, opinion polling aims to gain insight on voters' preferences, views, and voting intentions. Through the use of exploratory visualisation, this question will investigate if using Twitter analytics has the potential to be a less time-intensive alternative, or complement, to traditional opinion polling.

<u>Q2) What did Donald Trump's Twitter look like during his Presidency?</u>

Donald Trump's Twitter feed is a treasure trove of sentiment; he has tweeted thousands of opinions and reactions on a multitude of subjects, from all around the world, almost every single day. This question will provide an analysis of the rich trail of textual data left behind by Donald Trump through both frequency and sentiment analysis.

On the second of July, 2017, Donald Trump tweeted "My use of social media is not Presidential - it's MODERN DAY PRESIDENTIAL." (Trump, 2017). This is the crux of the motivation behind this report – gaining insight into, and understanding, how the nature of communication among the world's leaders and their followers is transforming, having far-reaching impacts and implications beyond just politics.

2. Data Wrangling

Donald Trump's Approval Ratings

FiveThirtyEight (https://fivethirtyeight.com) is a website that focuses on the statistical analysis of a wide variety of topics, including opinion polls and politics, among others. Fortunately, they provide open access to their data through their GitHub. I downloaded the Donald Trump approval ratings, which is an aggregated set of data sourced from the research/polling companies Gallup, Morning Consult, and Ipsos. The particular dataset that I downloaded included the dates 23/01/17 to 12/03/19, had 10 columns and 2338 rows.

In terms of wrangling, this dataset was quite simple to clean with the help of Python and Pandas. The steps of this process are as follows:

1. Open the csv file with pd.read_csv(…) and check with df.head().
2. Isolate 'all polls' under the column 'subgroup', and remove unwanted rows.

3. Drop unwanted columns using df.drop() as they are computationally inefficient. I decided that the relevant columns here were the approval and disapproval estimates, as well as ‘modeldate’.
4. Save the data frame to a new CSV file for further analysis with df.to_csv().

Donald Trump’s Twitter

Next, the Twitter data. The acquisition of this dataset was somewhat arduous as I was unable to simply use Twitter’s REST API to download all of the relevant tweets as there is a search index limit of about 7 days, and the dataset I needed spans over a period of over two years.

In ruling out the Twitter API as a means of dataset collection, I was left with two options: the Trump Twitter Archive or web scraping. The Trump Twitter Archive (www.trumptwitterarchive.com) is a website that collects and documents Donald Trump’s Twitter activity. It provides full access to its archive, available to download as either a CSV or JSON. Hence, I downloaded a CSV file. However, upon preliminary visual inspection, there were evidently a lot of problems with this dataset, including but not limited to:

1850	Twitter for iF	https://t.co/	10/10/18 1:18	8122	34233	FALSE	1049831428115062790\	
1851	Twitter for iF	Beautiful evening in Iowa. GOD BLESS THE U.S.A.! #MAGA						
1852	\f1 \uc0\u55356 \u56826 \u55356 \u56824							
1853	\f0 https://t	###########	18514	78472	FALSE	1049829285677203456\		
1854	Twitter for iF	RT @fema: F	10/9/18 22:18	4856	0	TRUE	1049786137500217345\	
1855	Twitter for iF	RT @FLOTUS: Thank you Egypt						
1856	\f1 \uc0\u55356 \u56810 \u55356 \u56812							
1857	\f0							
1858	\f1 \uc0\u55356 \u56826 \u55356 \u56824							
1859	\f0 https://t	###########	18218	0	TRUE	1049781519424712704\		
1860	Twitter for iF	RT @FLOTUS: Thank you Kenya						

Table 1

- Bad formatting, with the inclusion of many empty rows (see Table 1.)
- The inclusion of deleted tweets with favourite, reply, and retweet counts of zero. This would skew the data and create and inaccurate depiction of Trump’s Twitter activity.
- The favourite, retweet, and reply counts reflect the numbers at the time the tweet was collected (not the current day) creating vastly inaccurate numbers. This would also skew the outputs made in any further exploration

The next option was web scraping. Fortunately, Trump Twitter Archive provides details on how to do this. There are very few downsides to this method; but the main disadvantage is that it doesn’t collect every single tweet, but the large majority. Ultimately, the decision was made that scraping would be the best route to go down as the quality of the data was superior to the Trump Twitter Archive, even if the quantity was smaller. So, the process for procuring the data is as follows:

1. Use Twitter’s advanced search to bring up Donald Trump’s tweets for the time period 23/01/17 to 12/03/19 – I did this in three rounds so as not to overwhelm my computer.
2. Open the JavaScript console in Google Chrome with command + option + j
3. Paste in the full setInterval line below and wait for the page to automatically scroll
    a. setInterval(function(){ scrollTo(0, document.body.scrollHeight) }, 2500)
4. Paste in the javascript as specified in the instructions and press enter to automatically collect the tweets and their associated metadata. This automatically copies the data to the computer's clipboard.
5. Paste the JSON data into a TextEdit file and save.

The result of this process is a JSON file comprising 5633 records, with 8 values in each. It must be noted that the numbers of the favourites, retweets, and replies in these records are accurate as of time downloaded (25/03/19). An example of each JSON record is shown below:


```
{
  "id": "1105110383227035654",
```

```
    "timestamp": "1:17 AM - 12 Mar 2019",
    "text": "Making Daylight Saving Time permanent is O.K. with me!",
    "link": "https://twitter.com/realDonaldTrump/status/1105110383227035654",
    "is_retweet": false,
    "retweets": "32K",
    "favorites": "186K",
    "replies": "23K"
  },
```


This dataset required significant wrangling efforts to convert it to a clean, usable dataset that can be used for exploration. First, Python was used for initial cleaning. The steps are as follows:

1. Import the 'json' module to open and convert the data to a nested list using json.load() and check with a print() statement.
2. Normalize the data into a data frame using json_normalize() from the pandas.io.json module.
3. Remove unwanted columns 'is_retweet', 'link, and 'id' using df.drop().
4. For the favourites, replies, and retweets columns, use str.replace() alongside various regular expressions to convert numbers from shorthand versions such as '150K' to the standard version of '150,000'. To do this I used the following str.replace():
    a. [..].str.replace(r'(?<=\d{3})K', ',000')
    b. [..].str.replace(r'(?<=\d{2})K', ',000')
    c. [..].str.replace(r'(?<=\d\.\d)K', '00')
    d. [..].str.replace(r'(?<=\d)\.', ',')
5. The timestamp was in a format that was unrecognisable by Python such as '2:27 AM - 12 Mar 2019'. To fix this, I used another str.replace() with the regular expression: \d{1,2}:\d{1,2}\s\D{2}\s-\s to remove the time as only the date is needed.
6. Save to CSV file for further wrangling in R.

Next, R was used as there is some text pre-processing needed in order take the tweet content text from an unstructured to structured form. Once the CSV file saved in Python is opened using read_csv(), the following was executed:

Figure 1

```r
tidy_tweets <- tweets %>%
  mutate(text = str_replace_all(text, "https://t.co/[A-Za-z\\d]+|http://[A-Za-z\\d]+|&amp;|&lt;|&gt;|RT|https", "")) %>%
  unnest_tokens(word, text, token = "tweets") %>%
  filter(!word %in% stop_words$word,
         str_detect(word, "[a-z]"))
tidy_tweets
```

This snippet of code removes URLs or links in the tweet content using str_replace_all() and a regular expression. Then, the content is *tokenised*, which is the process of breaking the text into tokens by splitting on spaces, converting to lower-case and removing any punctuation. To do this, unnest_tokens() from tidytext is used, with the built-in option 'tweets', which uses a regular expression to parse the textual data in a Twitter-specific way, preserving usernames and hashtags. Each of these tokens are split into new rows in the data frame and the metadata from the original tweet is reapplied; for instance, a tweet containing 5 words in the original data frame would result in 5 new rows, each with a different word, but identical metadata. After tokenisation, stop words are filtered out. Stop words are words that are extremely common and carry little lexical content, such as 'a' or 'and'. The tidytext module has a built-in table called 'stop_words' which is used in a filter() function. The data has, at this point, been transformed into a much leaner body of text that is much cleaner for feature extraction. There are additional normalization techniques such as stemming and lemmatizing that I could have tried on the data, but Twitter messages are short by design (280 characters or less) and as such these methods don't work well because they essentially shorten words to their base words. e.g. *helping* to *help*.

3. Data Checking

Approval Ratings

In order to check this data for any errors, inaccuracies, or missing values, I used various graphical and non-graphical techniques from the Pandas and Matplotlib packages in Python. Firstly, I checked the data types that the data frame objects were stored as using df.dtypes. I found that the 'model date' column was saved as an object. So, I converted the timestamp to a datetime object using pd.to_datetime(), which saved the object as a datetime64. Then, I used df.info() and df.isnull().sum() to check for any missing values, of which there were none. Next, I used df.describe() to check for any outliers in the data that were incorrect. As a final check to ensure data accuracy, I plot the data onto two line graphs using Matplotlib. I set the x-axis to the model date and the y-axis set to the approval and disapproval rates, which are evidenced by Figure 2 and 3.

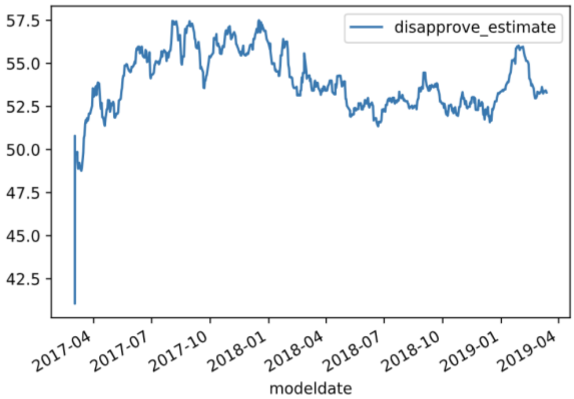

Figure 2

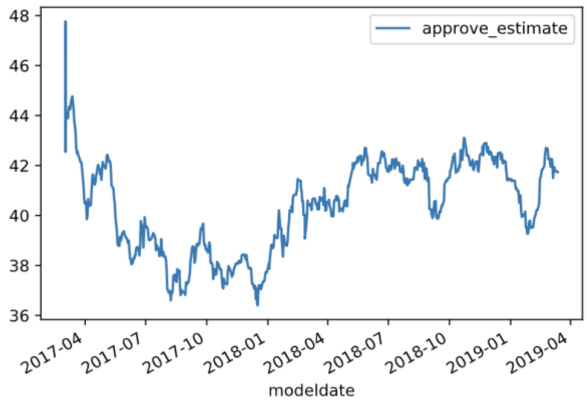

Figure 3

Twitter Data

Next, the Twitter data set. In terms of data checking, the process for this data was much the same as the approval ratings data. I used various methods such as df.isnull().sum() and df.info() to check for any missing values, and df.describe() to check for any outliers. Using df.dtypes, I found again that the timestamp needed converting to a datetime object using pd.to_datetime(). I also found that the favourites, replies, and retweets columns need to be converted to integers using df.astype(int) from objects. Finally, I plot the data onto three line graphs using matplotlib to ensure data accuracy. I set the x-axis to the timestamp and the y-axis to the favourites, retweets, and replies, which can be seen in Figures 4, 5, and 6.

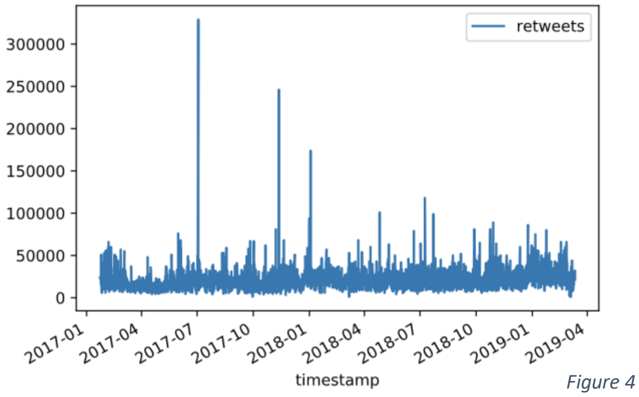

Figure 4

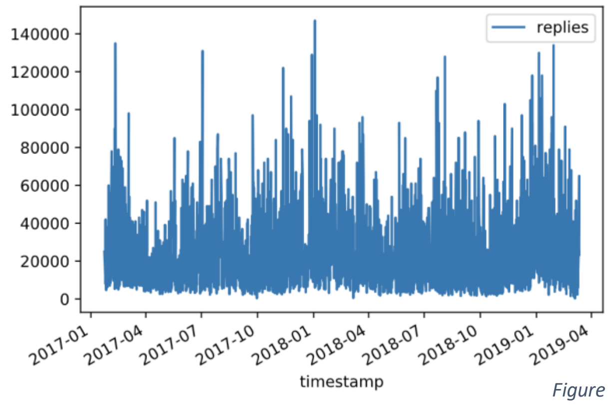

Figure 5

Figure 6

Through this process, I found that
there were a number of outliers in the data - for instance, where the number of favourites on a single tweet was over 500,000. I honed in on these tweets using df.loc[df['favorites'] >= 500000] and decided to manually check these using the link to the actual tweet in the 'link' column. I found that all of these were, in fact, accurate numbers, and simply represented viral tweets made by Donald Trump, for instance, the infamous: "Why would Kim Jong-un insult me by calling me "old," when I would NEVER call him "short and fat?" Oh well, I try so hard to be his friend - and maybe someday that will happen!" (Trump, 2017), which racked up over 570,000 'likes'.

4. Data Exploration

The first section will answer Question 1 and the final two sections will answer Question 2.

Popularity vs. Twitter Stats

To begin with, I decided to show the relationship between the approval ratings and various Twitter metadata. To do this, I used Tableau. However, I ran into some trouble, as the scales for the approval ratings and the quantitative Twitter data were so vastly different that it became difficult to plot. I first tried a dual-axis chart (see Figure 7), to compare the two trends with one another, and illustrate the information in a limited space. Hadley Wickham (2013), developer of the Tidyverse collection in R, explained why dual-axis plots are not available in ggplot2: "I believe plots with separate y scales (not y-scales that are transformations of each other) are fundamentally flawed." I'm not as dogmatically against dual-axis plots as some may be, but I did quickly realise that this chart could mislead viewers about the relationship between the two data series as the scales became arbitrary.

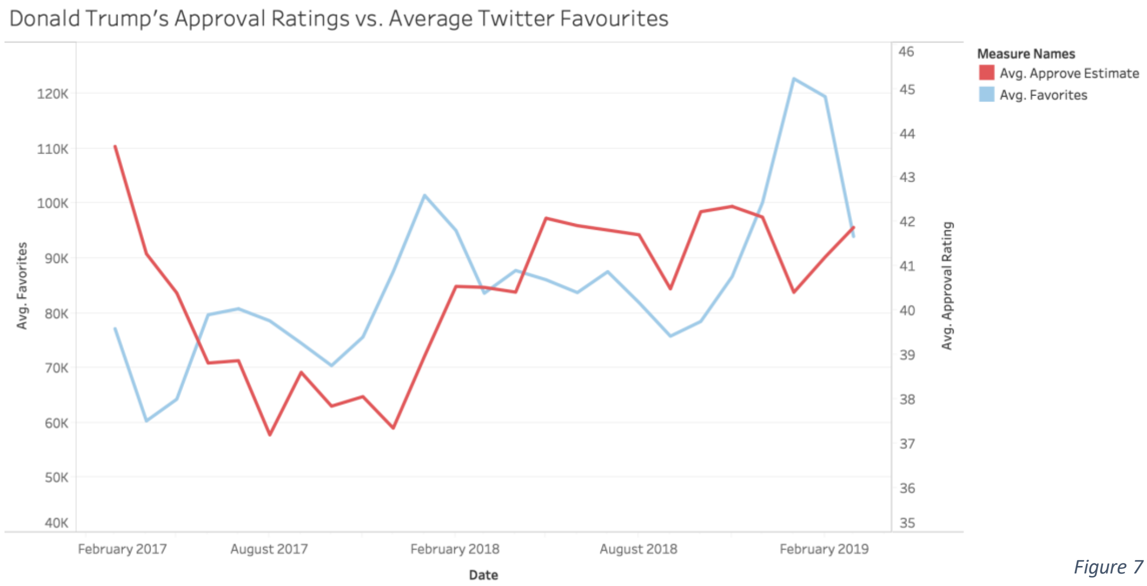

Figure 7

The options I was left with were either an indexed chart or a side-by-side chart. The indexed chart ended up not working as they only work for data series that have a similar rate of change; and the approval ratings, month on month, experienced anywhere between a +2 or -2 percentage change, whereas the Twitter favourites experienced anything between a -22% to +22% change. Therefore, I tried two side-by-side charts, the outcome of which is shown below in Figures 8 and 9.

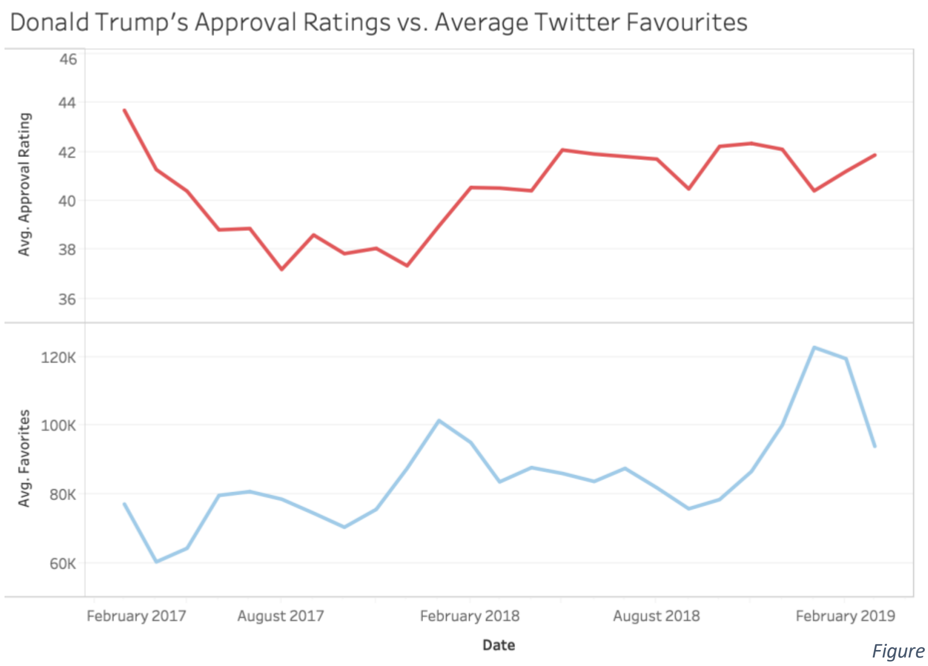

Figure 8

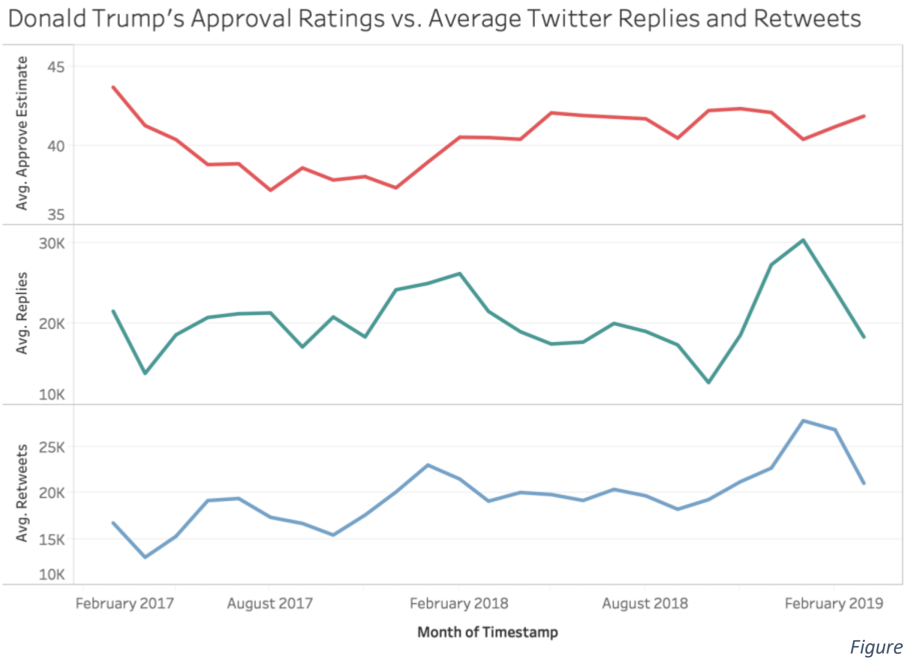

Figure 9

Furthermore, to use these graphs to answer the question: How do favourites/retweets/replies correlate to Donald Trump's popularity? It's difficult to say with a beyond-doubt level of certainty. However, although the match is not perfect, they do appear correlated, and there are some conclusions we can draw:

- There exists a stronger relationship between approval ratings and favourites rather than approval ratings and retweets/replies.
- The approval ratings and favourites followed a similar trajectory in 2017, but at the start of 2018, the two variables seem to start a trend of inverse relationship.
- Approval dropped to its lowest point at a 40.412% average in a year in January 2019, whereas favourites had an all-time peak of 122,720 average in the same month.

A recent poll by Gallup (2018) found that only 4% of American adults "have a Twitter account, follow Trump's account and read all or most of his tweets." This means that Trump's Twitter is not reaching the American electorate at large; few Americans see or read his tweets directly on the platform itself, but ultimately hear about them via media coverage due to the media hype over his frequent tirades and ramblings. The interaction with Trump's twitter comes from any age group, anywhere around the world – the grand majority of which are not included in the approval ratings polling. One viral tweet can be 'liked' by hundreds of thousands around the world, skewing any analysis. It's for these reasons that there may be a disconnect between approval ratings and quantitative Twitter metadata.

Frequency analysis

For the frequency analysis, I decided to firstly find the most common words used in Trump's Twitter corpus. To do this, count() is used to count the number of occurrences of each unique word, then filtered to remove hashtags and twitter handles, and finally ordered from most to least frequent. Then, these findings are visualised with ggplot2, which is a part of the TidyVerse meta-package. Geom_bar() is used to create an ordered bar graph, with stat= 'identity' so that the heights of the bars represent the values in the data. The colours and labels are added, and finally coord_flip() to rearrange so that the bars are horizontal instead of vertical. This code can be seen in Figure 10, and the output in Figure 11.

```
tidy_tweets %>%
  count(word, sort=TRUE) %>%
  filter(substr(word, 1, 1) != '#', # omit hashtags
         substr(word, 1, 1) != '@', # omit Twitter handles
         n > 250) %>% # include only most common words
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = word)) +
  geom_bar(stat = 'identity', color="#adadad") +
  scale_fill_manual(values=c("#f1f6fa", "#e3ecf6", "#d4e2f1", "#c5d8ec",
                             "#b6cee7", "#a8c4e2", "#99badd", "#8ab0d8",
                             "#7ca6d3", "#6d9cce", "#5e92c9", "#4f88c4", "#417ec0")) +
  xlab(NULL) +
  ylab('Word Count') +
  ggtitle('Most Common Words in Donald Trump\'s Twitter 2017-2019') +
  theme(plot.title=element_text(size=14, hjust = 0.5),
        axis.text=element_text(size=12),
        axis.title=element_text(size=12)) +
  theme(legend.position="none") +
  coord_flip()
```

Figure 10

From this plot, we can see that the frequency of words reflects some of Donald Trump's major beliefs and policies. The words 'news', 'fake' and 'media' referring to his negative rhetoric about the press, and 'border' and 'wall' reflecting a campaign policy of the construction of a larger and fortified wall at the southern border of the United States.

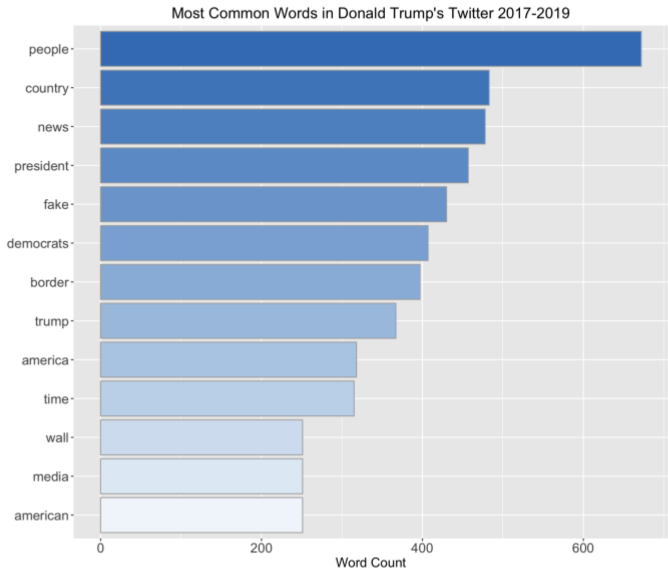

Figure 11

The same process can be implemented for finding the number of occurrences of particular hashtags and mentions (twitter handles). However, instead of filtering out the hashtags and mentions, we include only these using 'filter(substr(word, 1, 1) == '#'' and 'filter(substr(word, 1, 1) == '@'', respectively. In addition, the colours and labels are changed to fit the new plot, as seen in Figures 12 and 13.

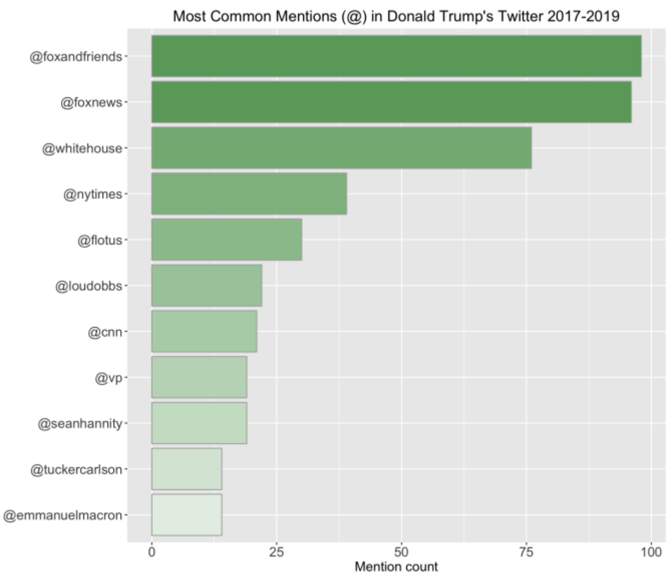

Figure 12

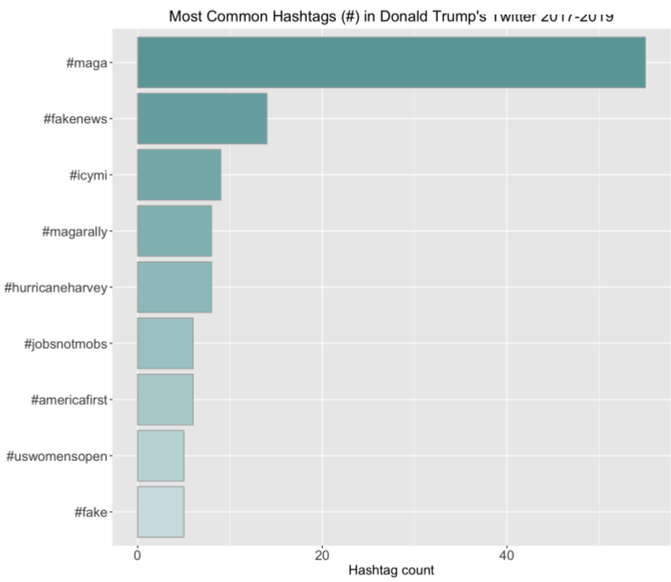

Figure 13

Once again, these bar charts are reflective of his anti-media rhetoric. @foxandfriends and @foxnews are branches of the American television channel 'Fox News', which is favoured by Donald Trump and known for its Republican Party and more general conservative bias in their news coverage.

The last frequency analysis explored was bigrams. Bigrams are common pairs of consecutive words, often which provide more detail or information than individual words (unigrams). Using the previously tokenised tibble, dplyr's lead() function is used to append the next word to each word with mutate(). Then, any unwanted tokens, such as raw numbers and hashtags, are removed with a filter function and substr() and str_detect(). Finally, unite() is used to transform the tokens into a single bigram.

```
tidy_bigrams <- tidy_tweets %>%
  mutate(next_word = lead(word)) %>%
  filter(substr(word, 1, 1) != '@', # remove
         substr(next_word, 1, 1) != '@',
         substr(word, 1, 1) != '#', # remove hashtags
         substr(next_word, 1, 1) != '#',
         str_detect(word, "[a-z]"), # remove words containing ony numbers or symbols
         str_detect(next_word, "[a-z]")) %>%
  unite(bigram, word, next_word, sep = ' ') %>%
  select(bigram)
```

Figure 14

Next, the bigrams are visualised in a bar graph, which took the same route as in the previous section – the only thing that changed was that the bigrams were counted with count() and sorted with mutate(). This visualisation is shown in Figure 15.

We can see from this plot that the top bigrams are all extremely topical. This is to be expected, as his tweets are so often stream of consciousness thoughts; if something is on his mind – he tweets it for the world to see. 'Fake news' took out the top spot, with over 300 mentions in the first two years of his Presidency – double that of the next most common bigram, 'witch hunt'. 'Crooked Hilary' and 'Hilary Clinton' both make the top-ten, too.

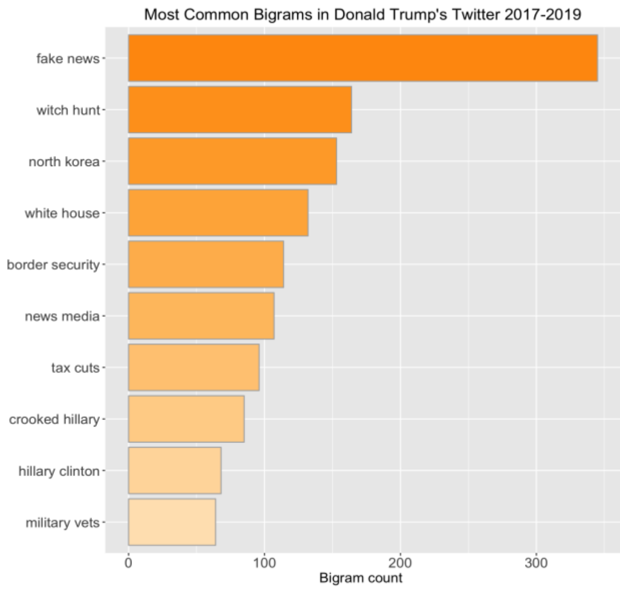

Figure 15

Sentiment Analysis

This sentiment analysis aims to track the attitude or emotion of Donald Trump's Twitter. In order to find the sentiment of the tweets, I used inner_join(get_sentiments("nrc"), by = "word"). This makes use of the tidytext function get_sentiments(), which makes it easy to match words against different lexicons or vocabularies. To begin with, I used the NRC lexicon for sentiment analysis, which has the categories: 'trust', 'surprise', 'sadness', 'positive', 'negative', 'joy', 'fear', 'disgust', 'anticipation', and 'anger'. Using ggridges and ggplot2, a ridgeline plot was made (see Figure 16). To do this I used geom_density_ridges(), which is a convenient way of visualising changes in distribution of sentiment over time.

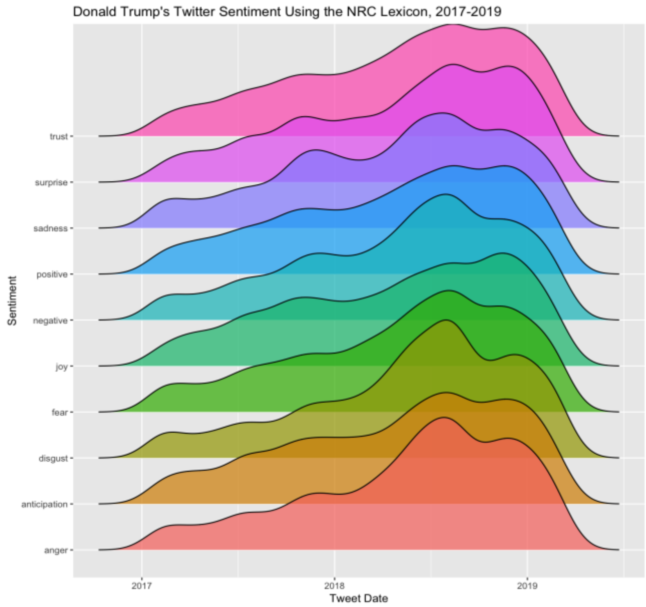

Figure 16

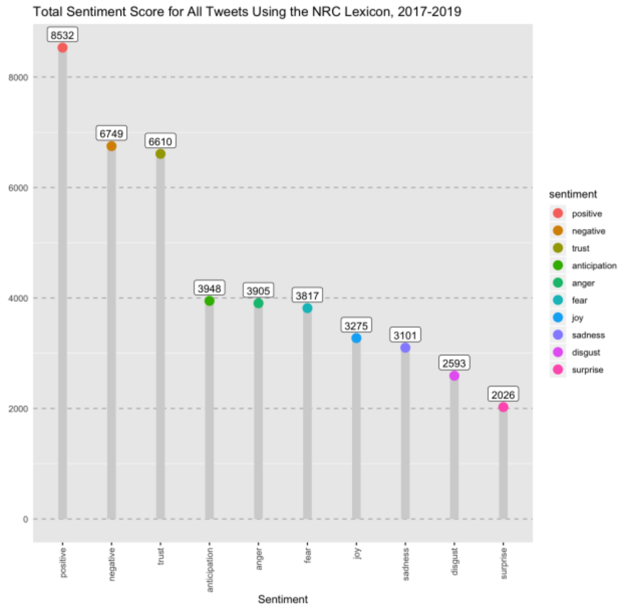

Figure 17

As is evidenced in Figure 16, the frequency of all sentiments was on an upward trend during this time period, peaking in either mid-to-late 2018. Perhaps this is because the frequency of tweets themselves peaked in this period of time, or perhaps the content of the tweets began moving away from the middle-ground, and started to oscillate between extremes. This being said, the ridgeline plot has obscured some data points due to overlap, so I also made a lollipop chart using ggdotchart() from ggpubr that plots total sentiment numbers over the period (see Figure 17). This shows that Trump's twitter content is mostly 'positive', followed by 'negative' and 'trust[ing]' far outweighing all other emotions.

I also decided to plot the sentiments using the Loughran lexicon, which was designed specifically for the analysis of shareholder reports, according to (van der Laken, 2017). This is because I thought it would be interesting to see the different categories – 'uncertainty', 'superfluous', 'positive', 'negative', and 'constraining' - had to offer. In addition, another lollipop graph was made with this lexicon. These plots can be seen in Figures 18 and 19.

We can see that, according to the Loughran lexicon, Trump's tweets are overwhelmingly negative, with the next sentiment, 'positive', only equalling less than half of the count for negative. Third is 'litigious', which is defined as 'too often taking arguments to a court of law for decision' by (Cambridge dictionary). I found this really interesting, so looked for the words that fell under this category - 'collusion', 'crime', 'lawyer', 'testimony', and 'suing' among others. Perhaps many of these litigious words are found in tweets relating to the various investigations and lawsuits plaguing his Presidency – of which he frequently refers to in tweets. For instance, "NO COLLUSION - RIGGED WITCH HUNT!" which he tweeted on the 23/9/18 in reference to the investigation into Russian meddling in the 2016 election that came to a head in late 2018/early 2019.

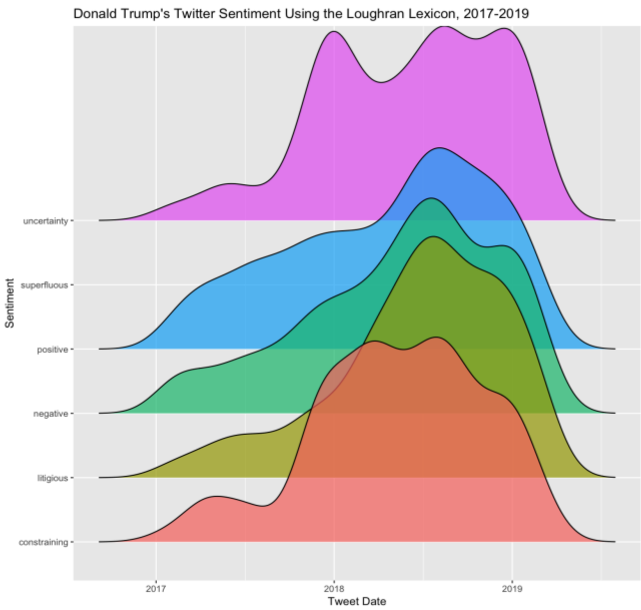

Figure 18

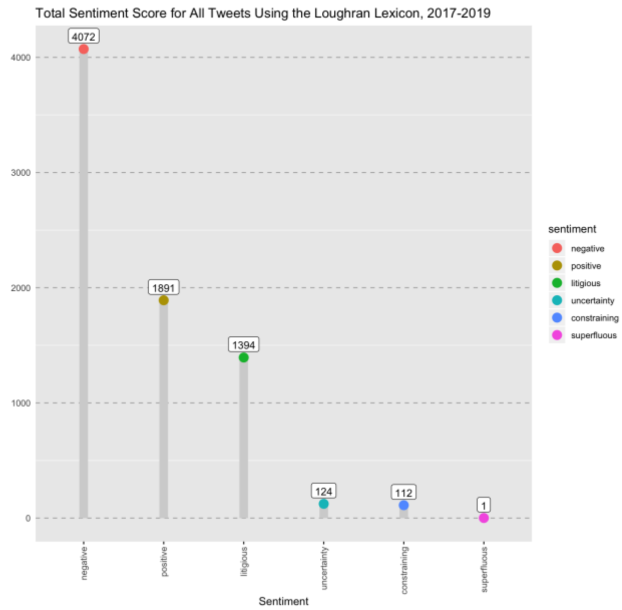

Figure 19

In addition to these ridgeline plots, two word clouds were made to help identify the most frequent 'anger' and 'joy' words in the corpus from the NRC lexicon – the more a specific word appears in tweets, the bigger and bolder it appears in the word cloud. To do this, the anger and joy words were filtered into separate tibble, grouped by word, and arranged in descending order. Then, I downloaded a picture of Donald Trump's silhouette to use as a mask (https://freepik.com), and used wordcloud2() to create the visualisation with figPath set to the downloaded picture, and the size set to 2. This is shown below in figures 20 and 21, with the blue word cloud being the joy words, and the red the angry words.

I think this was a really effective way of visualising the

Figure 20

Figure 31

words. The word clouds provide a really quick overall sense of the tweets and the silhouette lends context. The words displayed in the word cloud, once again, reflect the rhetoric of Trump's presidency – including the aforementioned vendetta against media outlets, and the various legal troubles plaguing him and his team. It is not all negative, though, and the 'joy' word cloud shows many positive words, like 'love', 'proud', and 'success'.

5. Conclusion

This report focused on analysing Donald Trump's Twitter to establish whether his popularity (or approval ratings) could be linked to his Twitter metadata throughout his presidency, and what exactly his Twitter looked like during this period.

Through exploratory visualisation, I found that the metadata and popularity share a tenuous connection at best due to various factors, including that his twitter is varied and wide in its reach, and doesn't represent a strong cross-section of actual American voters. Using frequency and sentiment analysis, I established an overall sense of what Trump's Twitter looked like over the past couple of years. I found a few surprising things, including that, despite what may be the general rhetoric surrounding his Twitter, it is more positive in sentiment than anything else.

In conclusion, there's a plethora of avenues to explore with Twitter data, and Trump's Twitter is certainly a goldmine of information – and an exciting future exists for the text analysis, exploration and visualisation of his Twitter, and perhaps others within the 321-million-personstrong Twitter community.

6. Reflection

I learnt a lot during this project, and in particular some technical skills, including:

- Exploratory data analysis techniques in the tidyverse;
- Various visualisation techniques in R, including ridgeline plots, lollipop charts, and masked word clouds;
- How to use numerous packages in R, such as ggridges, and ggpubr and wordcloud2;
- Web scraping methods using JavaScript; and
- Wrangling and checking tools in Python, especially Pandas.

I also learnt a lot about how to use visualisation techniques to best represent the dataset on hand. I discovered that not all visualisation techniques are created equally, and found limitations in various visualisations, nevertheless, I managed to produced some meaningful and interesting visualisations.

In hindsight, I would have abandoned efforts to find a correlation between popularity and Twitter metadata because it was unsubstantial and ended up producing dull visualisations. And perhaps in doing so, I would have been given more time to focus on the sentiment analysis, to explore it deeper and further. Having said this, in the future, it would be interesting to complete the same analysis for other politicians with less international notoriety and infamy.

7. References

Shaban, H. (2019, February 7). Twitter reveals its daily active user numbers for the first time. The Washington Post, Retrieved from: https://www.washingtonpost.com/technology/2019/02/07/twitter-reveals-its-daily-active-user-numbers-first-time/?utm_term=.0ac6e521989a

Donald Trump (2017, July 2). My use of social media is not Presidential [Tweet]. Retrieved from https://twitter.com/realdonaldtrump/status/881281755017355264?lang=en

Donald Trump (2017, November 12). Why would Kim Jong-un insult me [Tweet]. Retrieved from https://twitter.com/realDonaldTrump?lang=en

Hadley Wickham. (2013, November 5). Re: Plot with 2 y axes, one y axis on the left, and another y axis on the right [Online forum comment]. Retrieved from https://stackoverflow.com/questions/3099219/plot-with-2-y-axes-one-y-axis-on-the-left-and-another-y-axis-on-the-right/3101876#3101876

Van der Laken, P. (2017 December 17). Sentiment Analysis: Analyzing Lexicon Quality and Estimation Error [Blog post]. Retrieved from https://paulvanderlaken.com/2017/12/27/sentiment-analysis-lexicon-quality/

Newport, S. (2018). Deconstructing Trump's Use of Twitter, Retrieved from https://news.gallup.com/poll/234509/deconstructing-trump-twitter.aspx

Litigious. (n.d.) In Cambridge Dictionary. Retrieved from https://dictionary.cambridge.org/dictionary/english/litigious