Trump Tweet Semantics Report

Luke Ireland

1. Table of Contents

2. Introduction

This project revolves around analysing Donald Trump's twitter in various ways to provide interesting insights to his narrative.

Analysis methods include:

- Frequency Distribution
- Word Clouds
- Sentiment Analysis
- Tweet Generation

These methods will be presented as a web page via PonyORM

2.1. Background

I was interested analyzing Trump's tweets for semantics due to his controversial nature. It would be interesting to see, given what the media often say about him, if he truly is a bad/negative person.

2.2. Aim

The point of this project will be to perform various types of analysis on the language used in Trump's tweet to see if any interesting trends arise.

2.3. Objectives

1. Perform frequency distribution on variable length phrases
2. Render word clouds of phrases
3. Analyse whole tweets for semantics

3. Literature Review

3.1. Guide (To-Do)

- Technology Justifications
- Background Reading
- Justify using anything
- Be critical of decisions/guides/opinions

I decided to use Python as it's my strongest language, plus it's flexibility across platforms and level/variety of API support makes it an obvious choice.

I originally planned to use Twitter's API via Twitter Search[1], but I couldn't use it due to being unable to apply for a Twitter Developer Account. I instead opted for someone else's collected tweets at Trump Twitter Archive[2]. The export format wasn't great, as you had to wait a while for the page to compile all the tweets into the correct format (When it would be useful to have it precompiled) and the page doesn't actually give you a JSON file, just a text output in JSON format, that you have to slowly copy and paste into a file and use programs to format the JSON into readable format.

I saw guides such as Basic Binary Sentiment Analysis using NLTK[3], Text Classification using NLTK[4] and Creating a Twtter Sentiment Analysis program using NLTK's Naive Bayes Classifier[5] using NLTK's Naive Bayes Classifier, but they

[1] https://github.com/ckoepp/TwitterSearch

[2] http://www.trumptwitterarchive.com/archive

[3] https://towardsdatascience.com/basic-binary-sentiment-analysis-using-nltk-c94ba17ae386

[4] https://pythonprogramming.net/text-classification-nltk-tutorial/

[5] https://towardsdatascience.com/creating-the-twitter-sentiment-analysis-program-in-python-with-naive-bayes-classification-672e5589a7ed

used pre-processed data meaning I can't use them for my tweets. This guide[6] used Google's Natural Language API to perform Sentiment Analysis but this method required internet access, and wasn't particularly fast.

Eventually, I fell upon this article[7] which used TextBlob to perform sentiment analysis instead. TextBlob is a simplified text processing library for Python, and provides a simple API for performing Natural Language Processing tasks, such as speech tagging, noun extraction, classification, translation and, most importantly, sentiment analysis.

4. Parsing

I used Python's JSON library to load the .json file into the program as a dict.

5. Tweet Cleaner

I removed non-alphabetical characters from the tweet.

6. Sentiment Scoring

I used TextBlob to retrieve a sentiment score

7. Sentiment Classification

I decided to choose rather narrow ranges for each sentiment class.

- Less than -0.25 = Negative
- Between +0.25 and -0.25 = Neutral
- More than +0.25 = Positive

[6] https://www.freecodecamp.org/news/how-to-make-your-own-sentiment-analyzer-using-python-and-googles-natural-language-api-9e91e1c493e/

[7] https://www.geeksforgeeks.org/twitter-sentiment-analysis-using-python/

8. Word Cloud

Here is a word cloud I created using the Python library wordcloud.

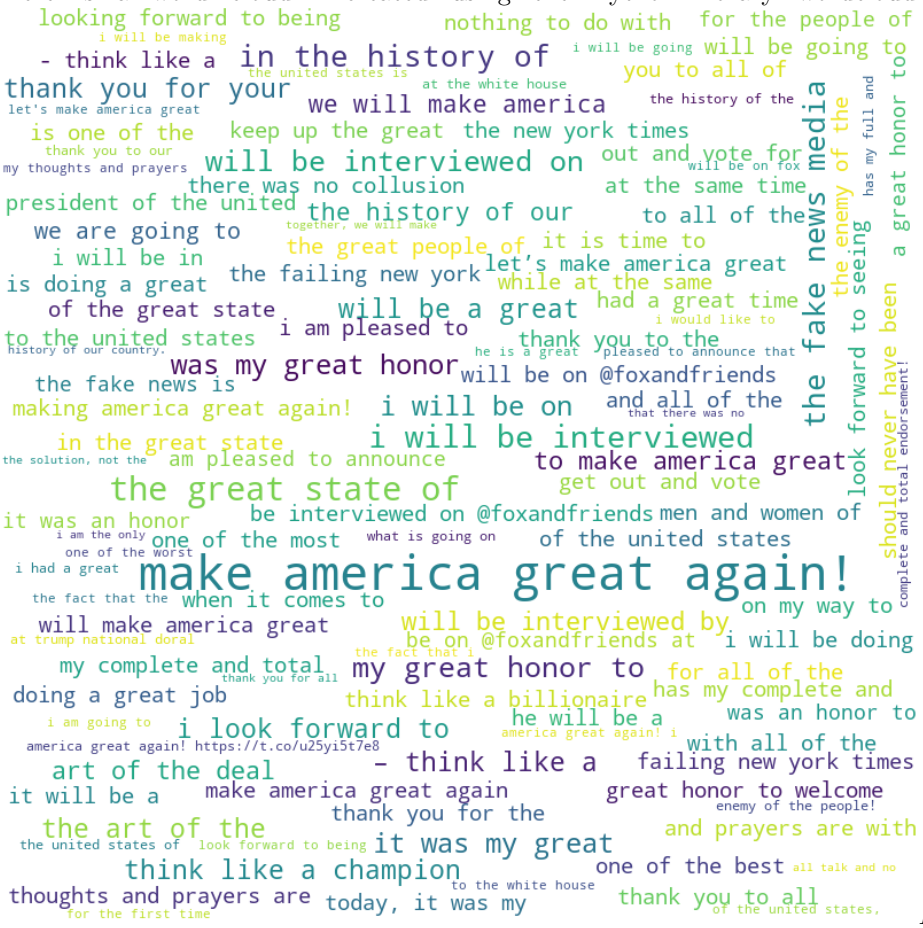

Figure 1 - WordCloud of phrases of length 4.

9. Frequency Distribution

I used NLTK to look at the most common words and phrases of different lengths.

10. Web Application Presentation

I used PonyORM to...

11. Evaluation

Results and findings

12. Conclusion

Footnotes