Comparing The Social Media Writing Complexity of Political Leaders, or, Does a World Leader Write Better than a Fifth Grader?

CMPT 318 - Data Science Project

Chris Norris-Jones
301047765

Abstract

With the ubiquity of social media, and the near-requirement of every public figure to make use of it, it seems like every major world leader now has an active Twitter account. With the current United States president often making headlines based on his use of the Twitter platform, parsing complex geopolitical topics into 140 (or now 280) characters, it felt like the general political discourse has been simplified. Hence the core idea to this data science project: is the Twitter output of major world leaders more or less complex than the average writing of a fifth grader?

This project intends to find an answer to this question. Through the data gathered in order to answer this first question, it will also attempt to try and answer two other questions, the first being anecdotal and the second perhaps having an actual application: whom amongst the current important political figures of the United States and the UK (specifically Hillary Clinton, Donald Trump, Barack Obama, Bernie Sanders, Jeremy Corbyn, and David Cameron) whose tweets on average have the most linguistic complexity, and is it possible to develop a machine learning model that can accurately guess a tweet's author, based on the sentence's complexity?

Required Files and Setup

Attached in the git repository are five files, and accompanying folder containing the pulled data in various formats. In addition it will be required to install the textstat python library, which contains the linguistic complexity tests performed on the pulled Twitter data, and the tweepy python library, which provides assistance in connecting with Twitter's API.

The five python files contained in the git repository are:

twitter_pull.py
Prerequisite Files Needed: None (Twitter API access keys required)

A script which makes a call to the Twitter API, and extracts up to the most recent 3,240 tweets of a provided Twitter user, specifically their tweet ID, the date of the tweet's creation, and the raw text in UTF-8 format, which is returned in a csv format. Original script was found from a git repository online (source can be found in file itself), with some changes made to make the output easier to use for future data cleanup steps.

tweet_cleaning.py
Prerequisite Files Needed: Raw output csv file extracted from *twitter_pull.py*

A script which takes the raw UTF-8 strings provided from the results of *twitter_pull.py*, and cleans up the data to make them as approximate to regular sentences as possible. After which, it also runs through the five chosen textstat linguistic tests on each tweet, determining and saving the various test results, appending a column containing the author of the tweet to be later used in the machine learning classifier, and returning the results in a csv file.

tweet_textstat.py
Prerequisite Files Needed: A csv file containing the corpus of text from fifth-grade writing samples

This script takes a csv file that's the listing of samples of fifth grade writing, upon which the five chosen textstat linguistic complexity tests are run against, with the results for each sample saved, and the final results returned in a csv file

tweet_stat_finder.py
Prerequisite Files Needed: The cleaned up csv data files of each world leader, as well as the tested data file of fifth grade samples

This file will run the various statistical tests on the provided data, first running a t-test comparing the fifth grade corpus to each individual political leader, then transforming the data into the necessary form to run two Tukey tests, one for the group just containing all world leaders, and one of the group containing all world leaders and the fifth grade sample results

tweet_classifier.py
Prerequisite Files Needed: The cleaned-up csv data files of each world leader

This file will take in the data for each provided political figure, and attempt to create a machine learning classifier that will attempt to discern a tweet's author based on a tweet's linguistic complexity.

In addition to the above five python files, the git repository also contains a file folder containing all the necessary extracted data, including the original raw data csv files for each world leader extracted from the Twitter API, the cleaned-up data csv files for each leader, as well as cleaned-up files for each leader that's been normalized to have the same number of rows (specifically 800), as Tukey statistical tests require datasets of equivalent size. Lastly, is also contained the original text file of fifth grade sample writing, as well as the csv file containing that same data, the data samples found by extracting samples from the internet, and then manually placing them into a csv file.

Data Gathering and Preparation

The ETL (Extract-Transform-Load) method for this project began through the installation of the tweepy python library, which provides a simple interface to connecting with Twitter's API. As well, I found a python github which allows the extraction of the important parts of a provided Twitter user's 3,240 most recent tweets (the maximum number of tweets Twitter's API allows to pull at a time), those important parts specifically being the tweet ID, the date and time of the tweet's creation, and the tweet text itself, in UTF-8 format. I made some slight changes to the original script, allowing me to extract a csv via the use of pandas as opposed to the built-in csv library.

I also found a python library called Textstat, which contains various tests that, given a string input outputs a ranking of that string's linguistic complexity. The details of the five tests I chose to implement on my data can be found in the following section.

Upon the extraction of a user's tweets, I then wrote a script that would take the raw csv data, turn it into a pandas dataframe, and clean up each raw tweet string, which contained things like emoji encodings, urls, hashtags, tweet signatures, and various Twitter language simplifications like "w/" instead of "with", which needed to be removed or replaced in order to get the tweets into a closer approximation of normal writing. Upon cleaning up each text string for a data file, I would then run a series of vectorized textstat tests on the tweet text column of the dataframe, saving the results of each into a new column. At the culmination of each test being run on all tweet strings, the cleaned-up dataframe would be return in a csv file.

I also prepared a corpus of text that contained numerous samples of average fifth-grade student's writing. This dataset was prepared somewhat more low-key than the political figure datasets, in that I manually trawled the internet for repositories teachers and schools maintain that contain "example" writing samples for different grades, copying those samples into a text file, splitting them so that they are roughly of the same length, while still remaining groupings of complete sentences, and then finally moving that corpus of data to a csv file.

Due to requirements for the Tukey statistical test, in that each provided dataset must contain the same number of samples, I also manually ensured that the number of elements within each dataset was the same, choosing 800 samples (the amount available within my least-populated political figure dataset).

Textstat Tests Detailed

Upon investigation of the available tests in the textstat library, I chose five tests to perform which all gave back a simple result, which corresponded to the approximate grade level required to understand the text. For example, if the result of a test on a string was "7.6", then the complexity of the sentence would be of that of a student between the 7th- to 8th-grade.

The five tests used and their details are:

Flesch-Kincaid Readability Formula: A relatively simple formula used to determine text complexity, which is the standard for most US and Canadian government agencies. The formula for which is:

RE = 0.39(total words/total sentence) + 11.8(total syllables/total words) - 15.59
RE = Flesch-Kincaid Readability Grade

Automated Readability Index: A formula similar to Flesch-Kincaid, but that relies on characters per word, as opposed to syllables-per-word. The formula is:

AR = 4.71(characters/words) + 0.5(words/sentences) - 21.43
AR = Automated Readability Grade

Coleman-Liau Index: A formula which, like the Automated Readability Index, relies on characters per words instead of syllables. The formula is:

CLI = 0.0588(Average number of letters per set of words) - 0.296(average number of sentences per set) -15.8
CLI = Coleman-Liau Index

Linsear-Write Formula: A readability index originally created by the US Air Force to determine the complexity of their technical manuals. The formula is:

Find a word sampling, calculate the easy words (defined as two syllables or less) and place a number "1" over each word, even including a, an, the, and other simple words. Calculate the hard words (defined as three syllables or more) and place a number "3" over each word as pronounced by the dictionary. Multiply the number of easy words times "1." Multiply the number of hard words times "3." Add the two previous numbers together. Divide that total by the number of sentences. If your answer is >20, divide by "2," and that is your answer. If your answer is <20 or equal to 20, subtract "2," and then divide by "2." That is your answer.

Dale-Chall Readability Score: A readability formula which uses an index of the 3,000 most common "simple" words in english, considering any word within the index "simple" and any not "hard". The formula is:

DC = 0.1579(difficult words/words * 100) + 0.0496(words/sentences)
DC = Dale-Chall Readability Score

Data Analysis

Upon cleaning up the data extracted from twitter, I ran a number of statistical tests, both comparing the datasets for each political figure against each-other, as well as the datasets compared against the fifth-grade sample dataset.
I initially ran a t-test for each political figure dataset, comparing them against the fifth-grade dataset, with the null hypothesis being that the average rating for both the political figure as well as for the fifth grade set were the

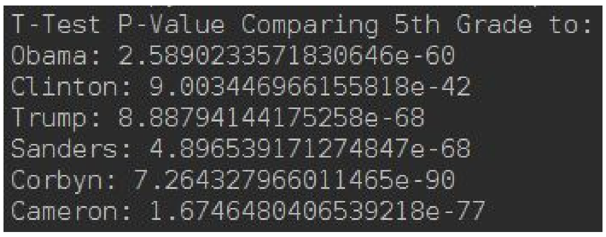
same. As the datasets for the political figures contained 800 samples, and the fifth-grade dataset contained 431 samples, I assumed normality of the datasets due to the Central Limit Theorem.

As I wanted a statistical test that allowed for the simultaneous comparison of multiple datasets, which would also provide a visualization of the different datasets, I utilized two Tukey Honest Significant Difference tests, one used on the melted concatenation of datasets of just the different political figures, and another using the melted concatenation of datasets amongst the political figures as well as the fifth grade sample set. In order to perform the later test, I was required to again shrink the sample sizes of the political figures to be equivalent to the fifth grade set, ensuring all data sets contained exactly 431 elements. I ran the two Tukey tests once for each textstat statistical test, and recording visualizations of each result for later analysis.

Lastly, I also created a number of machine learning classifier models, using the techniques we'd learned in class, to see if it would be possible to accurately predict the author of a tweet based on the results of the various computational complexity tests. The three models created utilized a pipeline that ensured the X inputs were all standardized using StandardScaler(), and then one of the three major classifier methods was used, either Naive Bayes, K-Nearest Neighbour, or SVC, with the results of all three being printed on the screen.

Results

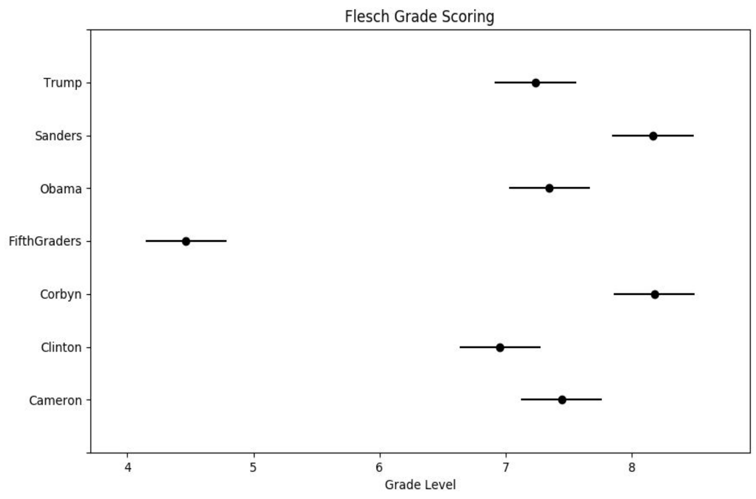
The results for each test performed were as follows.



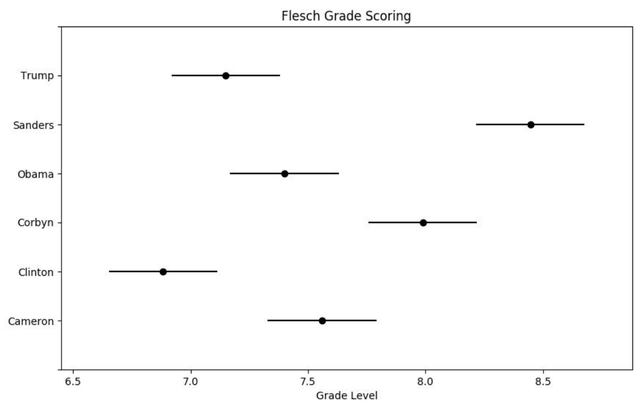
```
T-Test P-Value Comparing 5th Grade to:
Obama: 2.5890233571830646e-60
Clinton: 9.003446966155818e-42
Trump: 8.88794144175258e-68
Sanders: 4.896539171274847e-68
Corbyn: 7.264327966011465e-90
Cameron: 1.6746480406539218e-77
```

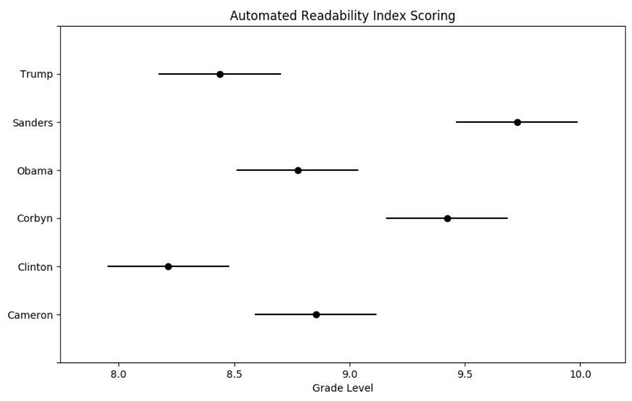
For the t-test ran between the fifth grade sample set against each political figure, the p-values were in the realm of approximately zero, and thus the null hypothesis could be reliably discarded for each comparison. Therefore, the averages of political leaders datasets were not the same as the fifth grade dataset. See the attached screenshot of the results to the right. This result was in line with my initial expectations, as I initially believed that the Twitter writing style would result in linguistic complexity less than that of an average fifth grade sample.

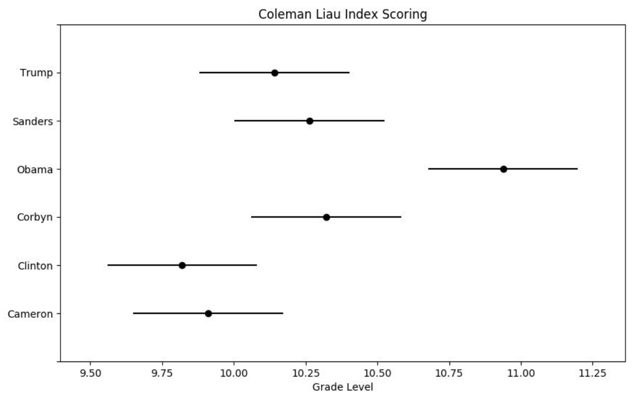
Upon running the Tukey tests that used the concatenated set containing both the political figures as well as the fifth grade dataset, I found results that defied my initial expectations. It could reliably be stated that for each test the fifth-grade dataset performed at a lower level than all politicians, with the resulting confidence interval visualizations showing the fifth grade set coming in last for 4 of the 5 tests, and second-last for one test (the Linsear-Write Score test). I have attached the confidence interval visualization for one of these tests, specifically when tested using the Flesch-Kincaid Grade test, to the right. As you can see the average tweet complexity for the political figures was around the 7th to 8th grade while the fifth grade set (logically enough) was around the 4th to 5th grade.

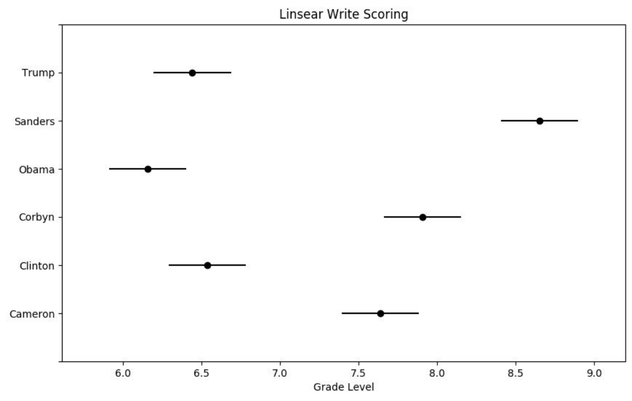


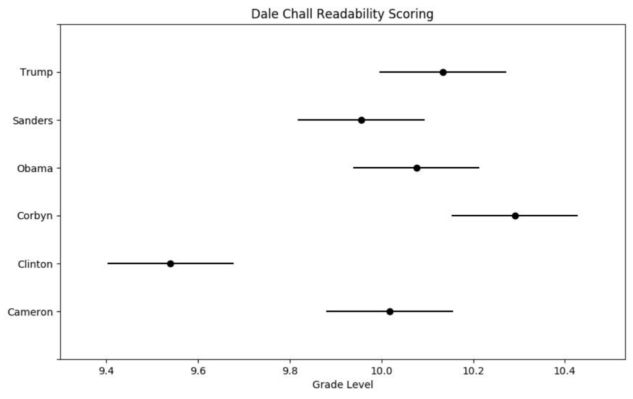
When comparing specifically the different political figures, the results remain relatively similar throughout all the tests, with some slight deviations between most tests. Below are the visualized confidence interval results for the five Tukey tests comparing specifically political figures:







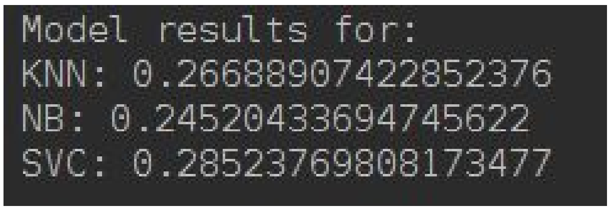




The reliable winners look to be Bernie Sanders and Jeremy Corbyn, with Barack Obama and David Cameron reliably holding the middle of the pack, with Hillary Clinton and Donald Trump holding the last place. This was somewhat in line with my expectations before seeing the results, in that my (admittedly somewhat biased) opinion expected to see Jeremy Corbyn or Bernie Sanders at or near the front of the pack.

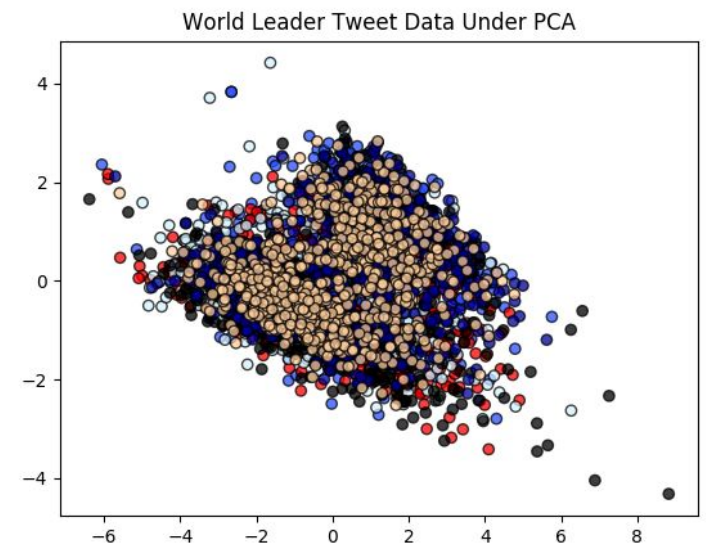
What truly surprised me, however, was that in nearly every case Hillary Clinton came in last, and with the results of the Dale Chall Readability score coming decisively in last place. Perhaps Clinton's twitter writing style doesn't cleanly translate when it comes to linguistic complexity?

Lastly, upon developing and testing the three machine learning classifiers which would attempt to predict a tweet's author based on the results of the five complexity tests, it was found that the model scores, while statistically better than the method of "guessing at random", were not particularly strong. After various tweaks and tests, the Naive Bayes result would reliably



```
Model results for:
KNN: 0.26688907422852376
NB: 0.24520433694745622
SVC: 0.28523769808173477
```

come in third, the K-Nearest Neighbour model (with a K-value of 15-40 seemingly producing the optimal model results) coming second, and the SVC model, with an rbf kernel and a C value of 1-10, coming in first. That being said, the score for any of the models never went past 30%. See an example result to the above-right.



Upon further examination of the clustering of results, it becomes pretty evident why a machine learning model, at least given these specific features, would have a hard time being particularly accurate. The picture to the right shows the visualization of the various tweet clusters, decomposed using principal component analysis into two dimensions. It seems enough to say that I initially attempted trying to find a method for drawing a legend on the visualization to the right, but quickly determined there would be little point. The clustering is much too compacted together, leading me to believe that trying to predict a Twitter author based on a tweet text's complexity may not be the best approach.

Limitations

There are plenty of limitations to this project that likely resulted in the results being more obscure or less accurate than truly possible. To start, the extraction method used to interface and pull data from the Twitter API was capped at 3,240 tweets, limiting me from pulling the entire tweet history of most of the political figures selected. If given more time, I also would have preferred to have found a larger sample corpus for the fifth grade writing sample set, and to have found them from more varied sources.

The current fifth grade sample set came from taking around a dozen fifth-grade essays, writing samples, or writing assignments, and splitting them up so that they were of roughly the same length as an average tweet. This, one could easily argue, somewhat skews the data away from a truly representative sample of a fifth grader, as it's more likely than not that those fifth graders whose writing samples are being saved for posterity are likely somewhat better than the average fifth grader.

Finally, the methodologies behind many of the linguistic complexity tests expect to take in a large corpus of text, usually around 100 or so words, which is not just impossible but antithetical to the output expected from a Twitter tweet, as it's bounded by 140/280 characters. As well, with the inherent size limitations of a tweet in Twitter, which thus leads to creative attempts by users to fill as much context, meaning, and creativity into just 140/280 characters, this can lead to constantly shifting slang or word mutations outside of the normal english language. As the textstat tests used in this project are generally designed to work with traditional english, using them on tweets may not always be a perfect fit.

Conclusions

Coming back to the three underlying topics I wanted to investigate when starting this project, I believe I have a satisfiable answer for each of them. Based on the results of the t-tests and Tukey tests comparing the fifth grade sample set to the political leaders, I'm comfortable stating that, at least amongst the political leaders chosen for this project, they do all write reliably more complex sentences on twitter than the average writing of a fifth grader.

I was also able to create some form of generalized ranking amongst the six political leaders chosen, with Bernie Sanders and Jeremy Corbyn taking the lead on most tests, Barack Obama and David Cameron in the middle, Donald Trump sitting near the back of the pack, and Hillary Clinton reliably in last. Lastly, I can say with some certainty that, while a machine-learning classifier model's predictions of a tweet's author based on the tweet's complexity will give you *somewhat* better results than just simple guessing, it isn't what you would call a strong prediction model.

At least, it's hard to predict given the knowledge of classifier models taken from CMPT 318, and the linguistic complexity tests used in this project. Perhaps there's a most complicated model possible, using better attuned tests for determining the linguistic complexity of short text phrases.

Project Experience Summary

To complete this project, I had to utilize my statistical and machine learning skills, as well as my python programming knowledge. I did so by setting up and utilizing an interface with an outside API, transforming and cleaning up various large sets of string data, entering them into datasets and performing various vectorized tests, and then utilizing a handful of statistical tests to check my hypothesis regarding comparative complexity of political figures' Twitter samples and the writing of an average fifth grader.

I also created a number of machine learning models to see if the data provided could allow predictions of Twitter authors. To that end, I was successful in coming to a conclusion regarding initial fifth grader vs politician question, and was also able to put together that a model that performed better than average.