# Predicting the virality of tweets related to the 2015 Canadian federal election

## Introduction

This project will use natural language processing in order to predict the likelihood that a tweet will be re-tweeted. This is in the context of the Canadian federal election that took place on October 19, 2015. This year, there was an unprecedented public interest in the election outcome. Citizens, news organizations and political actors took to twitter in order to express their views on a range of hot button topics related to the campaign platform of the Conservative, Liberal and New Democratic Parties. Which tweets are most likely to be re-tweeted? Is this correlated with the topics that are most important to voters? Which algorithms are most useful in analyzing text data for predictive purposes? Such analysis is useful because it allows those invested in the political process to have a real-time gauge of public response to political positions announced by politicians on various issues. Sentiment can be understood by geography (given geo-tagged data from Twitter), or other user identifiers. Based on such analysis, politicians can choose to adjust campaign positions to target certain segments of voters.

Key research questions include (i) Can twitter topic analysis be used in order to predict the retweet frequency of a given tweet? (ii) Which topics were most heavily retweeted? iii) Which algorithms are most useful in classifying text data?

## Literature Review

This literature review focused on two main themes. First, on the data collection processes used to organize research studies on twitter virality. This is a relevant topic for my study given the limits of the Twitter API. Research questions on twitter topic or content virality are limited to the types and amount of data that can be collected using the Twitter search API. As such, researchers require creativity as well as an understanding of what constitutes a quality dataset in order to stage a viable study using Twitter data. The second theme studied was that of text analysis. What are the most effective ways of organizing text data to extract meaning? Which algorithms are most helpful in doing so? How does one analyze text data for topic analysis?

Zaman et al. (2014) predicted the popularity of tweets using a time series analysis of its retweets. The team's data collection method is particularly relevant for my capstone study. The team identified 52 tweets in a range of subjects (music, art, sports, etc) and then used the twitter search API in order to track all retweets of those original tweets over the course of 1 week's time. This use of twitter to collect data is interesting as it collects meaningful data within the limits of what is publicly accessible via twitter. The team developed a probabilistic model for the evolution of the retweets using a Bayesian approach, and formed predictions using only observations on the retweet times and the local network

or “graph” structure of the retweeters. The team obtained good step ahead forecasts and predictions of the final total number of retweets even when only a small fraction (i.e. less than one tenth) of the retweet paths are observed.

In the article "Predicting Information Spreading in Twitter," Zaman et. al (2010) find that the most important features for prediction of a re-tweet are the identity of the source of the tweet and retweeter. The team's data collection method involved collection of every tweet from a one hour time window. Then the team looked for any retweets of these tweets for up to one hour after the time of the tweet. These retweets were the positive binary feedback. The team obtained the obtained negative feedback from all followers of the tweeter in the retweet network who did not retweet. This data contained over 99.8 % negative feedback because most tweets are not retweeted.

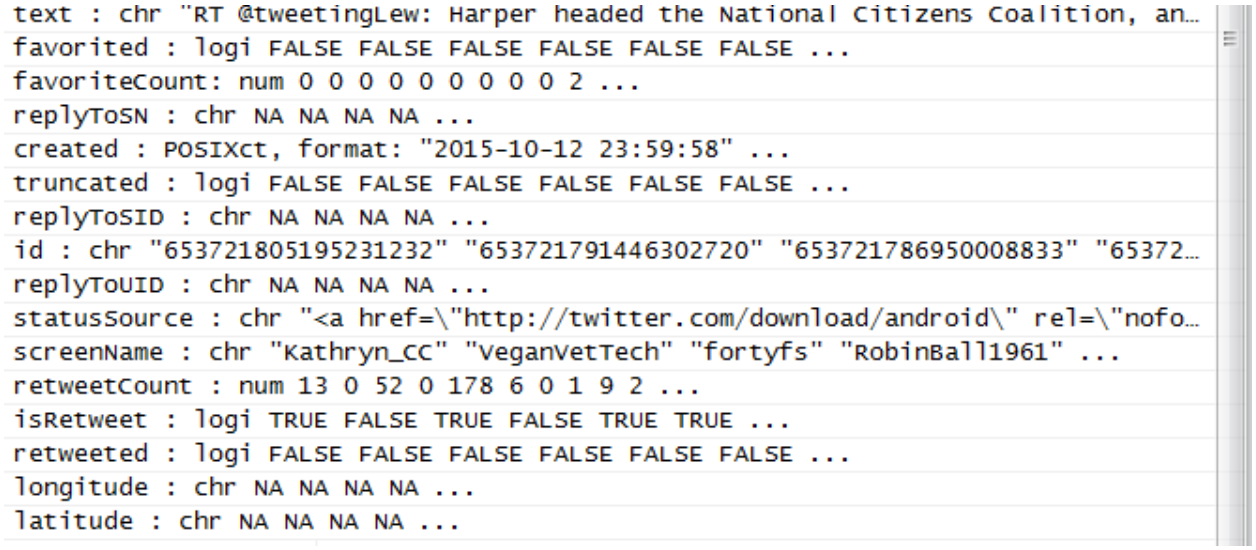
Hoang et. al (2011) studied the virality of socio-political tweet content in the Singapore’s 2011 general election (GE2011). The team collected tweet data generated by about 20K Singapore users from 1 April 2011 till 12 May 2011, and the follow relationships among them. The team identified a topic as a group of tweets of similar content. The team classified each tweet as containing only one topic and adopted a clustering approach
to construct topics. The team took a modularity based clustering algorithm [3] on a tweet graph so as to derive subgraphs representing topics. The tweet graph consists of original tweets as nodes, and pairs of tweets with overlapping terms as edges. Each edge is weighted by the similarity of the two corresponding tweets. To compute similarity between two tweets, we use the “bag-of-words” representation of each tweet after removing tweet-encoding terms, e.g. @, RT, and via, common stop words and internet slang words[7]. The remaining words form a word vocabulary W.

Related to the theme of accurately assessing text topics and sentiment, Bakliwal et al. (2012) emphasized the importance of preprocessing and proposed a set of features to extract maximum sentiment information from tweets. They used unigram and bigram features along with features which are more associated with tweets such as emoticons, hashtags, URLs, etc. and showed that combining linguistic and Twitter specific features can boost the classification accuracy.Davies and Ghahramani (2011), propose a probabilistic model for sentiment analysis of short, social-network statuses. This model is language agnostic and thus is useful in analyzing non-English data. The model is demonstrated on data from Twitter, modelling happy vs sad sentiment, and showed that in some circumstances this outperforms similar Naive Bayes models by more than 10%. Go et al. (2009) show that machine learning algorithms (Naive Bayes, Maximum Entropy, and Support Vector Machines) have accuracy above 80% when trained with emoticon data. This paper also describes the preprocessing steps needed in order to achieve high accuracy. The main contribution of this paper is the idea of using tweets with emoticons for distant supervised learning. What is clear from this review of literature is that pre-processing of data is critical to accurate twitter sentiment analysis.

## Dataset

The dataset for this project was downloaded via the Twitter API using R. The TwitteR package was used to search the API for 10,000 tweets containing the last name of each of the leaders of the political parties: Harper, Mulcair or Trudeau between the dates of Oct 1- 2015 to Oct 13- 2015.

This yielded a dataset that included the following fields:




From the list above, the text data was transformed using the TFIDF function. The following variables above were used in the final model: favorited, favoriteCount, truncated, isRetweet. The others were deleted either because they were not meaningful or there was no data in the vector.

The following additional features were added: Hour, Minute, Clusters and Bins.

```
hour : int 23 23 23 23 23 23 23 23 23 23 ...
minute : int 59 59 59 59 59 59 59 59 59 59 ...
bin : num 3 1 4 1 4 2 1 2 2 2 ...
```

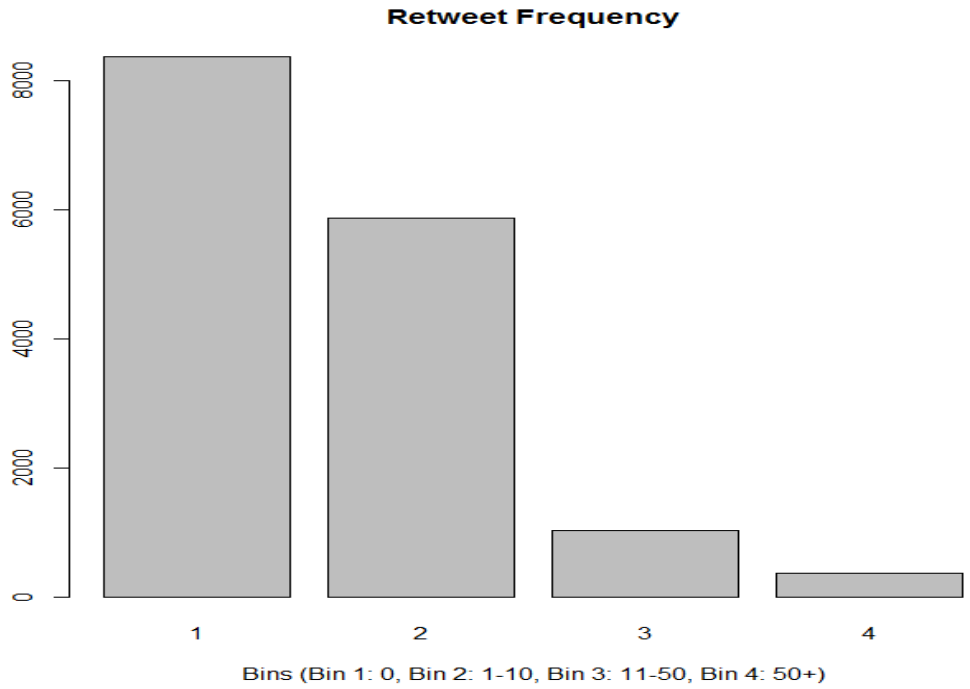
The variables hour and minute were extracted using a time stamp of the "created" variable originally downloaded from Twitter. The as.POSIXlt function was used in order to extract the hour and minute. Seconds variable was ignored as it was assumed that this would have little impact on the retweet count

The variable “clusters” was created using a text analysis bag of words with Kmeans clustering. Each tweet was thus assigned to one of 4 clusters. This variable was used only in the analysis specific to cluster data.

The variable "bin" was created by separating the "RetweetCount" variable into four bins. The reason for this transformation was to simplify the problem given that the variable had a very long tail with a range between 0 and 151,000 retweets, median value of 0 and mean of 39.5 retweets.

After making several visualizations of the RetweetCount variable, I decided on a "binning" where Bin 1: All tweets with zero retweets, Bin 2: Tweets with retweetCount between 1-10, Bin 3: Tweets with retweetCount between 11-50, Bin 4: Tweets with retweetCount over 50+ . This effectively transformed this project into a classification problem. This is the distribution of data after I binned it:

| 1 | 2 | 3 | 4 | Total |
|---|---|---|---|---|
| 8371 | 5871 | 1033 | 373 | 15648 |
| 53% | 38% | 7% | 2% | 100% |




Given the user id, it is possible to extract user objects from twitter which include information on the number of followers and friends an individual user has. Literature on this topic shows that follower counts are important predictors of retweet counts. In order to extract this data you can use the getUser, and lookupUsers function in the twitteR package. In the final dataset, I did not include these variables because I had difficulty downloading the appropriate data for all 10,000 tweets from twitter given API rate limits. There is a limit of 180 pieces of information per 15 minutes. Given the twitter API limits, I can try using a different algorithm using lookupUsers. The lookupUsers vector will return a list of user objects, so if I fill it with 180 screenames. However, again, this would require calling the Twitter API on 160 separate occassions with 15 minute intervals. Given my own time constraints I chose not to include this information.

## Approach

My original proposal aimed to predict voter preferences in the Canadian federal election based on geographic data.

However, there were several methodological issues with this. First, there was not enough data from twitter for a given geography (say Ontario). I tried to download all tweets with the last name "Harper",

"Mulcair" or "Trudeau" for the 1 month preceding the Canadian federal election. Furthermore, it was unclear on what would constitute a vote (i.e. - one tweet?) or would I look at scoring sentiment and then predict proportions? Originally, it was planned that the data would be gathered via Twitter API, with filtering by the names of prime ministerial candidates (Stephen Harper, Thomas Mulcair and Justin Trudeau) as well as the political party names (Conservative, New Democratic and Liberal). However, when the API was searched Canada-wide for these tags, a scarcity of tweets was available.[1]

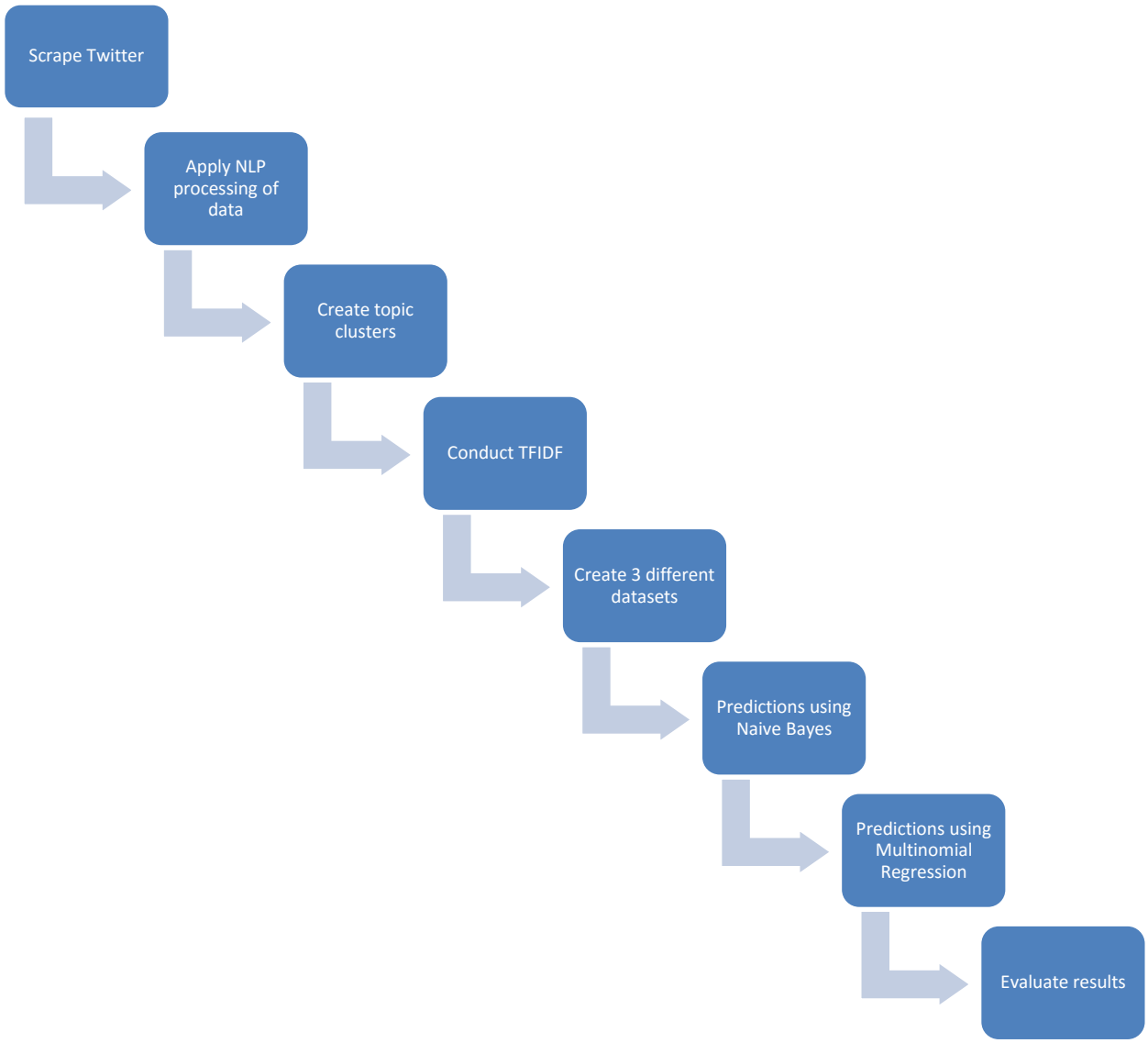
The algorithm for classifying tweets and scoring them to be used was the simple method proposed by Breen (2011). In this algorithm, a score was assigned to each tweet with the number of occurrences of negative words is subtracted from the number of positive. Larger negative scores correspond to more negative expressions of sentiment, neutral (or balanced) tweets net to zero, and very positive tweets score larger, positive numbers.

After creating a scoring algorithm for sentiment (and realizing that over 80% of sentiment for these given politician in twitter is negative), and after accepting the fact that geotagged data was unavailable in sufficient quantity to create predictions for individual ridings, I decided to change my topic slightly.

As a result, I re-visited my dataset with an eye to predicting retweetCounts. This transformed the question into one centered on topic analysis versus a strict sentiment analysis.

My new topic, takes the same dataset from above (thus 10K tweets for each of Harper, Mulcair and Trudeau) for Oct 1- 2015 to Oct 13- 2015. It then aims to predict the number of times a given tweet is retweeted. A topic analysis is conducted in order to assess whether certain topics were more likely to be retweeted.

[1] As an example, a search of "@pmharper" Canada-wide (using geo-tagging and a large radius) results in less than 20 tweets.




Once this is done, explain each of the steps in detail. What are you planning to do in each step or have already done. For example, in the above case you would create subheadings for each of the steps.

## Step 1: Scrape Twitter

The twitter API was scraped using a Twitter developer account. 10,000 tweets for each of Harper, Mulcair and Trudeau for Oct 1- 2015 to Oct 13- 2015 were downloaded. For the code on accessing the Twitter API, see my github account: github.com/SunitaKosaraju

## Step 2: Data cleaning, preparation and application of NLP

Remove duplicate tweets. Transform relevant variables. Apply standard text analytics pre-processing steps undertaken using the "tm" package. This includes stop words removal, stemming, corpus creation (using the tm package), sparse word removal, string within a string, and conversion to lower case.

## Step 3: Create topic clusters

Use Kmeans in order to create 4 topic clusters. A larger number of topic clusters was not possible given the limited RAM available for data analysis on my computer.

## Step 4: Analyze text using Term Frequency -Inverse Document Frequency

Use the TFIDF weighting function in the TM package. Add word features to the dataset. This added over 100 variables to the dataset.

## Step 5: Create 3 different datasets

Create train and test sets for 3 combinations of data:

Dataset 1: No variables representing tweet data. Bin prediction using only favorited, favCount,truncated, isRetweet, Retweeted,hour, min variables [total 7 predictor variables].

Dataset 2: Cluster variable is added to the dataset. Therefore, Bin prediction using favorited, favCount,truncated, isRetweet, Retweeted, hour, min and cluster variables [total 8 predictor variables].

Dataset 3: TFIDF variables added to the dataset. Cluster variable removed. Bin prediction using favorited, favCount, truncated, isRetweet, Retweeted, hour, min and 115 variables [total 123 predictor variables].

## Step 5: Predictions using Naïve Bayes

Use the naïve Bayes function in the "e1071" package to make predictions on the 3 datasets above.

## Step 6: Predictions using Multinomial Logistic Regression

Use the multinom function in the “nnet” package to make predictions on the 3 datasets above.

## Step 7: Compare results

# Results

Baseline prediction

A baseline prediction for this problem is created as follows. From the full dataset, we see the following breakdown by bin:

| 1 | 2 | 3 | 4 | Total |
|---|---|---|---|---|
| 8371 | 5871 | 1033 | 373 | 15648 |

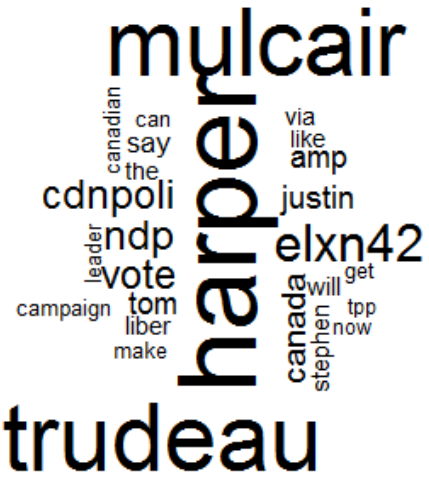
This baseline shows that 8371/ 15648 records fall into bin 1. Therefore if my default model predicts a tweet will not be retweeted, I will be correct 53% of the time

Clustering of data

The dataset included 15648 unique tweets. The wordcloud below shows the distribution of text data. It is clear that the politician names are the most prominent words given that the data collection method involved a Twitter API search of Trudeau, Harper and Mulcair as key words.

Word Cloud of overall dataset

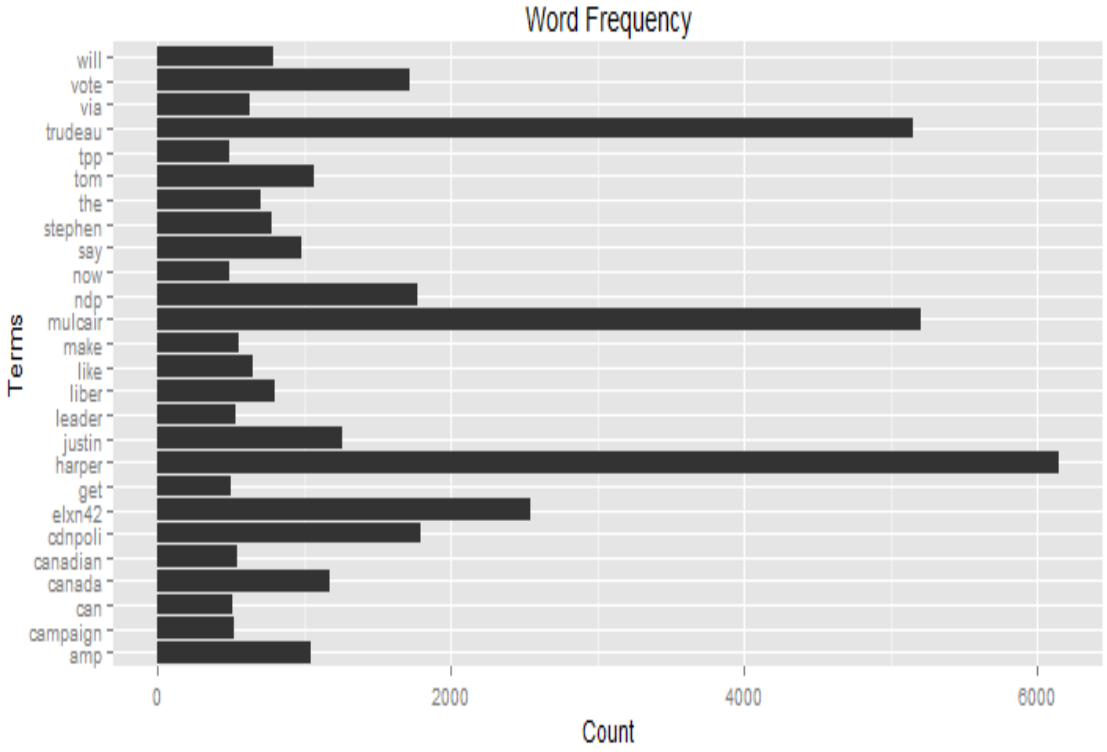
Here is the output




After conducting a bag of words analysis, below is a list of the top 26 most frequently appearing terms.

As is the case in the word cloud, most of these words are related to the party name or the leader name.

As you can see, the words are not very interesting or reflective of major campaign issue themes. Another type of analysis is thus, necessary.




Next, I clustered terms using Kmeans algorithm. Below is the list of clusters.

Here is the list of clusters:

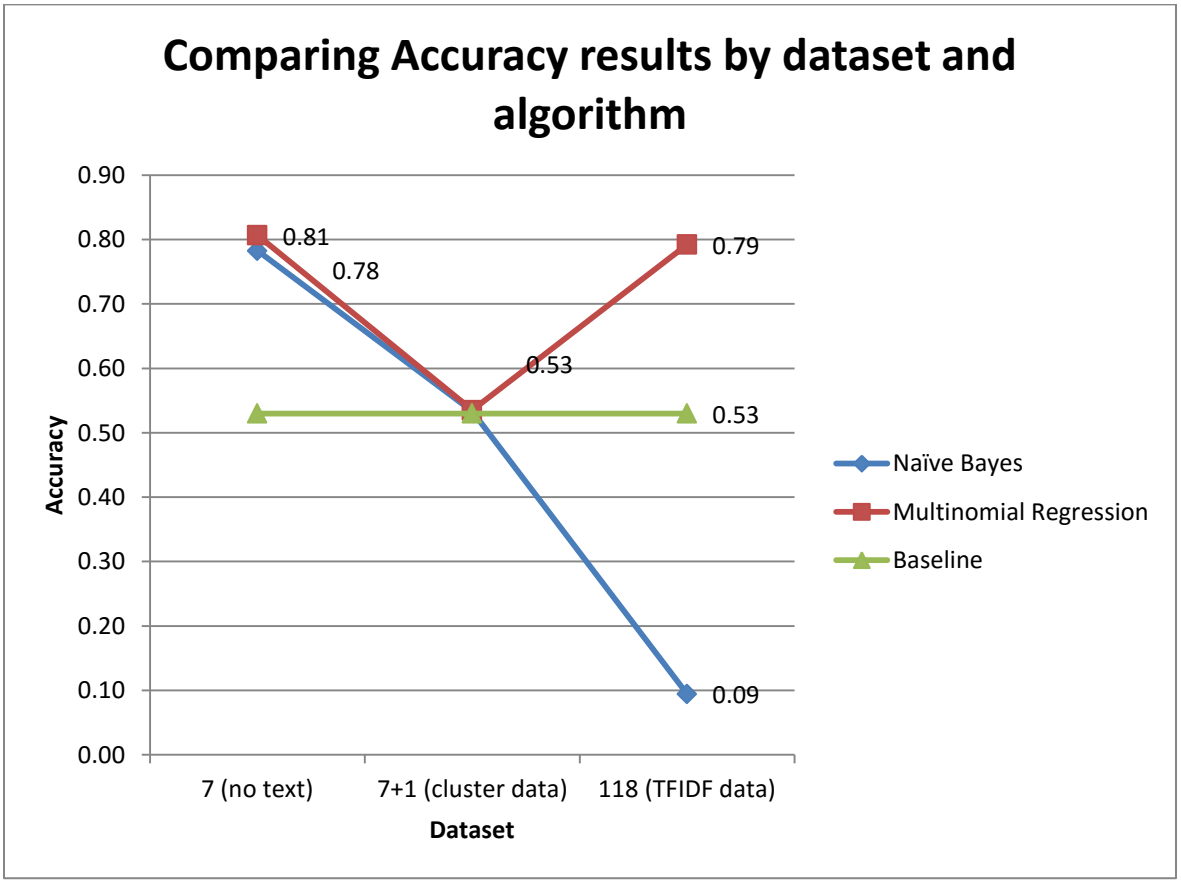
```
cluster 1: harper vote stephen elxn42 canada
cluster 2: mulcair ndp tom say elxn42
cluster 3: elxn42 cdnpoli trudeau harper mulcair
cluster 4: trudeau justin harper mulcair vote
```

As seen in the clusters, these are not very meaningful or reflective of specific campaign themes. It is unlikely that replacing text frequency data with cluster number will result in accurate predictions. I may be losing a lot of data unnecessarily. As a result, I also conducted TFIDF analysis. As a result, I kept over 100 variables instead of limiting my tweet value to just 1.

Comparing the results of 3 different datasets and 2 different algorithms

Overall, multinomial regression outperforms naïve bayes regardless of dataset. The most predictive model for retweets included the dataset with no text data and using the multinomial algorithm with 81% accuracy. This is surprising as you would expect that at least some text analysis would create a more predictive model. When text data was included multinomial regression analysis resulted in an accuracy of 79% while Naïve Bayes significantly underperformed at 9%. The least useful dataset (regardless of algorithm was that which reduced text data to clusters). The results of the analysis are displayed in the table below and in the graph below.

| | Accuracy | | |
|---|---|---|---|
| **Dataset** | **Naïve Bayes** | **Multinomial Regression** | **Baseline** |
| 7 (no text) | 0.78 | 0.81 | 0.53 |
| 7+1 (cluster data) | 0.53 | 0.54 | 0.53 |
| 118 (TFIDF data) | 0.09 | 0.79 | 0.53 |



**Confusion Matrices for each of the above applications**

**Dataset 1: 7 variables (no text analysis)**

**Naïve Bayes prediction**

```
NaiveBayesprediction
       1    2    3    4
  1 2060   33    0    0
  2  471  993    4    0
  3   39  211    8    0
  4   10   83    0    0
```

Accuracy: 78%

**Multinomial Logistic Regression prediction**

```
logregprediction
       1    2    3    4
  1 2024   68    1    0
  2  376 1086    6    0
  3    2  216   38    2
  4    0   83    3    7
```

Accuracy: 81%

**Dataset 2: 8 variables (7 + 1 cluster feature)**

**Naïve Bayes prediction**

```
   naiveBayesprediction
       1    2    3    4
  1 2065   28    0    0
  2 1448   20    0    0
  3  243   14    1    0
  4   87    6    0    0
```

Accuracy: 53%

**Multinomial logistic regression prediction**

```
   logregprediction
       1    2    3    4
  1 2093    0    0    0
  2 1468    0    0    0
  3  256    2    0    0
  4   84    9    0    0
```

Accuracy: 54%

**Dataset 3: 118 variables (TFIDF analysis)**

**Naïve Bayes prediction**

```
naiveBayesprediction
       1    2    3    4
  1  227   49   63 1754
  2   79   40   54 1295
  3   11    4   15  228
  4    3    0    2   88
```

Accuracy: 9.5%

**Multinomial logistic regression prediction**

```
logregprediction
       1    2    3    4
  1 2042   51    0    0
  2  437 1015   12    4
  3    5  216   36    1
  4    0   80    7    6
```

Accuracy: 79%

I see from above that just 9% of the data is predicted to the correct class. It appears that Naive Bayes is predicting the majority of the data to class 4.

## Conclusions

The conclusions from my study are centered around three topics:

1. Data collection via Twitter- Use of twitter as a source of original data for sentiment and topic analysis has a range of difficulties. As someone without a corporate account, your ability to collect twitter data is limited by daily rate limits, and expiry dates on the range of time accessible to you. Further, the very nature of twitter data means that often critical information such as geography, or other demographic information is inconsistently available. As such, when using twitter data in future studies, it is important to take an iterative approach to research design- first posing a question and envisioning the type of data needed to answer that question, then checking twitter to see what is available. In my case, I formulated my research question first, recognized the data was not available (at least not in the right quantities) and then reformulated my research question based on the data I had on hand.

2. Extracting meaning from text data - In this study, clustering of twitter data using the knn algorithm was not useful. There was too much overlap in the four topics and the most common terms were those

of the names of the politicians and political parties. This was reflect ed by the fact that the dataset that included text data had the lowest accuracy level of prediction in all trials. In the future, a clustering could be attempted after filtering for political names and parties. In order to get around this problem, it was useful to use the Term Frequency Inverse Document Frequency. This enabled me to retain the nuance of each tweet, and keeping over 100 variables rather than reducing each tweet to be summarized by just 1 variable (i.e. a cluster number). In the future, a PCA of the text words could be conducted in order to filter down the text data to the most meaningful words (thus reducing the data set from 100 text variables to a smaller number).

3. Finally, this study showed that as a predictive algorithm, the Multinomial Logistic regression was superior to the Naive Bayes algorithm regardless of dataset. The improvement in accuracy ranged from 1% to 70% (in the case of the TFIDF dataset). The reason for this significant difference in predictive power is to be investigated in future studies.

Thus, this study showed a great deal in terms of useful processes in data collection via Twitter, text analysis, and algorithms with high strength.