Detecting Political Ideology and Sentiment from Tweets

Gradient Dissenters
Maharshi Pandya, Varun Tanna, Tanvi Tyagi

Identifying political ideology is often a challenge faced by many political scientists. Scientists often rely on surveys to determine the sentiment towards different presidential candidates. However, they face various statistical challenges ranging from low sample sizes to biased data. Twitter, on the other hand, offers a wealth of data instantaneously with more than 68 million monthly active users in the United States alone. People often voice their opinions on Twitter with more freedom than classical survey methods that are prone to response bias. Using tweets, scientists can extract opinions, and political momentum that are prevalent with the public more readily than any other means. Examples of Twitter being used for the sentiment analysis include stores using customer reviews and newly launched product trends in the market and sports analysts predicting NBA All Star rosters based on player popularity. In this project, we focus on two main research questions: detecting political ideology from a particular tweet and detecting sentiment towards different democratic candidates before and after the Nov. 20th democratic debate.

When looking for the Twitter data on sites like Kaggle, we found that there weren't many comprehensive datasets with tweets from the current political leaders from whom we wanted to monitor the tweets. Scraping the tweets directly from the Twitter seemed to be the best plan of action at this point, due to the ability for us to retrieve any and every tweet we wanted without any limitations. Using the python library “twitterscraper”, we were able to retrieve the data straight from the Twitter Developer API. We were able to successfully gather a collection of tweets from both Democratic and Republican Senators, Representatives and Democratic Presidential Candidates. The tweets were passed through standard text cleaning pipelines in NLP before using them in building various classification models.

A classification problem was the result of our first research question. In the first section, we present various models trained on the cleaned tweets from the Democratic and Republican members of the House and the Senate to classify tweets based on political affiliation. Some of the models we tested include Logistic Regression, Stochastic Gradient Descent, Linear SVM and Multinomial NB which we received quality accuracy scores for. However, the nature of tweets requires us to consider the progression of words. To take into account the sequential nature of the text, we also built an LSTM units based RNN network, which resulted in 76.29% validation accuracy. In the second section, we present various models trained on pre-labelled dataset to predict the sentiment of tweets before and after the November 20th, 2019 democratic debate for each of the ten participating candidates. Our results indicated that the Linear Regression model resulted in highest accuracy of 82.30%.

PART I: DETECTING POLITICAL IDEOLOGY

BACKGROUND

As aforementioned, the overarching goal that we are trying to achieve in this project concerned with using Twitter as an opinion mining platform and extract public opinion on a specific topic, which in this case is a political ideology and sentiment behind each tweet. The two main routes that we will explore are detecting whether a tweet is of democratic or republican views, and looking at tweets related to the Democratic debate to see public sentiment on candidates before and after the Nov. 20th democratic debate. Both of these problems in their roots are classification problems which require different opinion mining techniques like natural language processing and text classification to proceed. None of us has had much experience at building models using textual data or scraping, so the process of acquiring and analyzing the tweets involved numerous hurdles, and a chance to expand our skill sets. Hence, once we scraped our data, we needed a way for it to be usable in our models, so we additionally had to undergo pre-processing of the data before training/testing the predictors.

PRE-PROCESSING OF DATA

The data pertaining to the research question 1, classifying a tweet as a democrat or a republican, was scraped from twitter using a python library called "twitterscraper". Our goal was to extract ~500 tweets from each of the congressional members' twitter handle for our dataset. For the Senate, we were able to scrape 21,514 tweets from democratic senators and 25,534 tweets from republican senators. In total, we were able to scrape 47K tweets from 100 senators. For the House of Representatives, total of 76,519 tweets from republican members and 83,572 tweets from democratic members were scraped. In total, 160K raw tweets from the 342 members of the House were scraped. It is important to note that even though there are 432 total members in the House, we only scraped tweets from the members with active twitter handle. In addition, independent members as well as House members of the other parties were excluded from our data extraction.

Once the raw dataset of tweets were collected, another tedious yet a key step of data cleaning was needed to be performed. For any NLP task, cleaning out text is extremely important. For cleaning our tweets, we created a pipeline through which each tweet was passed as a string and cleaned up. In our pipeline, we first tokenized tweets by breaking sentences into words. Then, all the words in the tweet were lowecased and non-ASCII characters from the list of tokenized words were removed. All punctuations and numbers as well as stopwords from NLTK package were also removed from the list of tokenized words. Standard text cleaning pipelines also include stemming and lemmatization, but in this project we decided to not perform these in order to keep the word order and tweet's context intact. The cleaned versions of each tweet and tweet's author's party affiliation (1 if Republican and 0 if Democratic) were saved and later utilized to build several classification models.

INITIAL MODELS FOR CLASSIFICATION OF TWEETS
as Republican or Democrat

Some of the initial models we wanted to conduct on the data for classification into democrat an republican were Logistic Regression, Multinomial Naive Bayes, Stochastic Gradient Descent and a Support vector machine with a linear kernel. In order to apply the pre-processed data to the algorithms, we needed to find a different way of vectorizing the words in the tweets. To solve this issue, we constructed a pipeline in python which allowed us to streamline the process of vectorizing the tweets and training the data on the train set. One vectorizer that we found useful was the TF-IDF (term

frequency–inverse document frequency) vectorizer, which we used to build a dictionary consisting of the most frequent words in the dataset while also being able to account for the number of tweets that the term occured in.

We used a 70:30 split in the data to construct the train and test sets respectively as it allowed for the best results in accuracy score when testing the four models mentioned above. The first classifier we trained was a discriminant model known as Logistic Regression. Here on the Senate data we saw an accuracy score of 35% with a sensitivity of 0.16 and a specificity of 0.51. While on the House of Representatives we saw an accuracy score of 81% with a sensitivity of 0.82 and a specificity of 0.81. The House of Representative has similar sensitivity/specificity values indicating the model was similarly good in predicting correctly as democrat/republican, while the Senate had the republicans being predicted more correctly often. We also saw the accuracy score differ immensely, giving us reason to believe that having more samples will lead to a more well rounded dictionary, thus allowing the classifier to work better. Having scraped ~150,000 samples for the House of representatives, while only ~50,000 for the Senate, it was in our assumptions that models would predict better for the House, a trend we saw through out all our future models.

After using a discriminant analysis technique, we sought to look at a generative model like Naive-Bayes, knowing well that we had met the independence assumption. We used the multinomial version of the Naive-Bayes classifier to account for the multinomial distribution associated with utilizing word counts, a key feature of classifying tweets. The accuracy reading we got for the Senate prediction was 37% with 0.06 for the sensitivity and 0.62 for the specificity. On the contrary, for the House, we achieved an accuracy of 82% with 0.86 and 0.78 for the sensitivity and specificity respectively. Similar to Logistic, the republican tweets were classified correctly more often for the Senate, with similar correctness in the House tweets. We also saw similar accuracy scores between the two models, giving us reason to believe that the Logistic Regression performed slightly better in regard to higher sensitivity and specificity values.

Since the number of words in our dictionary was vast and since there was a lot of tweets, we decided to use the Linear kernel of a support vector machine which allowed us to use the fact that text is linearly separable. The linear kernel is also more computationally efficient, and with a large number of tweets and a time constraint, this was best. We acquired 38% accuracy, 0.19 sensitivity and 0.54 specificity for the Senate data, while getting 82% accuracy, 0.82 sensitivity and 0.81 specificity for the house. Similar trends were noted, as the republican's tweets were predicted better than the democrat's on the Senate data, while performing equally as well on the House. From the results, we realized that House data metrics were similar for the three models, while this model performed better across the board for the Senate data, giving us insight that this was currently a better predictor of political affiliation. Even though we would have liked to find the optimal parameters to use, we were unable to because of the time, >12 hours, it took to run even the basic model we created.

Furthermore, we tested a more computationally friendly model than the SVM to see if it provided better results. Stochastic Gradient Descent was used, which provided us values of 53%, 0.02 and 0.97 for the accuracy, sensitivity and specificity in the Senate, while giving us 79%, 0.80, and 0.77 for the same values. Although this model awarded us a better accuracy score in the Senate, the difference in the sensitivity and specificity is a cause for concern. This meant that the model was predicting most of the tweets to be republican (more false negatives), showing us that relatively high accuracy is a result of the test set containing more republican tweets in general. The final determination was that due to the

inconsistencies in this model, and lower scores across the board for the House, that SVM was a better model. However, one thing we later noted that was important to us, was the sequence of the words in tweets, prompting us to look at another model to address this issue.

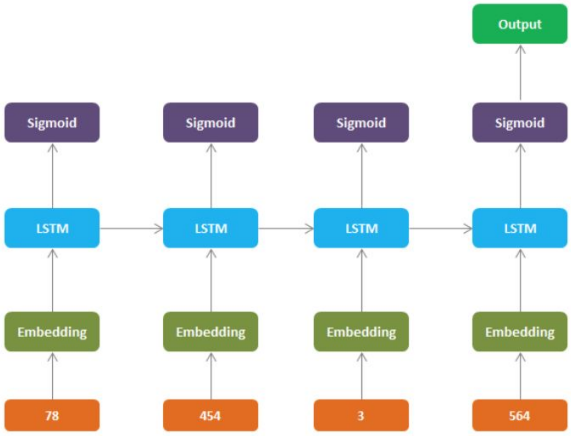


Figure 1. RNN Model Architecture

To address the intrinsic nature of any text, specifically the placement of words in a sentence, we decided to explore the usability of a recurrent neural network, particularly, a network containing a layer of Long Short-Term Memory (LSTMs) units. Our motivation to explore such a possibility came from realizing that our model should be able to successfully classify the following two tweets, “The Republican Party has NEVER been so united” and “The Republican Party has NEVER been united” as a Republican and Democratic tweet, respectively, even though they are only a one word apart. Additionally, our model should be able to recognize any long-term dependencies, that is the relationship between two sequence of words with a large gap between them, from tweets. Using LSTMs over simple RNNs allows us to take in account these long-term dependencies present in the text.

In Figure 1, we present the RNN model architecture used for this project. We started with pre-trained 100-dimensional GloVe embeddings that were trained on 2B tweets. After performing padding to ensure each sequence is equal in length, in the Embedding layer, each word of the tweet was vectorized using 100-dimensional vectors. Next up is the LSTM layer with 100 memory units. The output from LSTM were activated using Sigmoid activation function, which resulted in scores having values between 0 and 1, as required for binary classification. Additionally, the loss was calculated using binary cross-entropy function. As seen in the figure below, the model accuracy and model loss on the validation set leveled off around epoch 10, indicative of model being overfitted on higher number of epochs. The 10th epoch accuracy on the validation set was 76.29% with loss value of 0.4708.

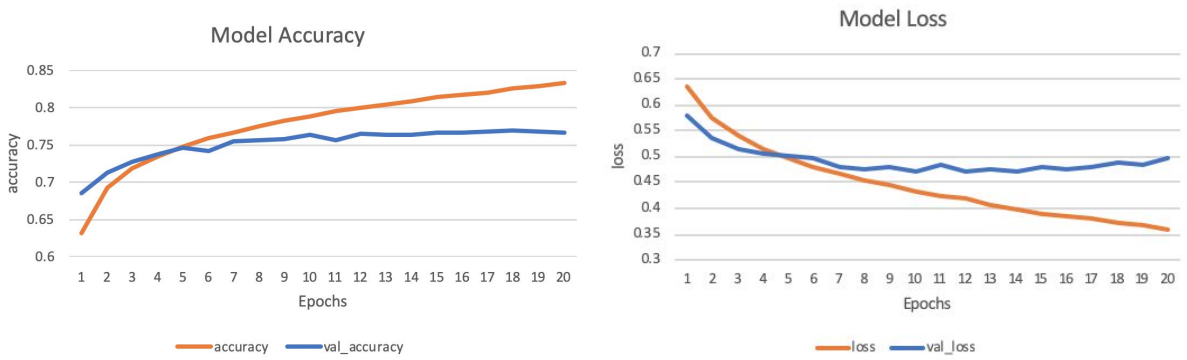


Figure 2. Model Accuracy and Model Loss plotted against Epochs. After Epoch 10, model overfitting can be seen.

With limited resources available, we were not able to perfectly tune the hyperparameters in our model architecture. In the literature[1], use of multiple LSTM layers has shown to increase the model accuracy. In the future, we plan to use free GPUs available from Google Collab and Kaggle to see if our model performs better with multiple LSTM layers.

PART II: SENTIMENT ANALYSIS

BACKGROUND

The second problem we considered for our project was whether we could accurately predict the sentiment on Twitter towards a particular democratic candidate before and after the November 20th, 2019 debate. Researchers in several domains have realized the predictive potential of identifying and extracting opinions from a given text on social media. As an important medium for political discourse, Twitter has often been used to make accurate predictions on election outcomes. Several studies have also been published demonstrating the importance of Twitter in predicting outcomes for the 2010 US Congressional elections as well as the outcome of electoral results in Germany in 2009.

Sentiment analysis is a popular technique used in many research papers to analyze people's opinion on electoral candidates. One research study focused on evaluating Twitter as a platform for political deliberation.[2] They evaluated a sample of 104,003 political tweets along two indicators of "blog-based deliberation", the exchange of substantive issues and equality of participation. They found that substantive issues could be explained in a 140 character tweet, however it was unclear whether the general public participated in political discussion. While promising results from social media predictions have created hype towards this methodology, there are some recent studies that claim that sentiment analysis exhibits inconsistencies. For instance, a research paper, found that Google Trends correctly predicted a set of electoral races in 2008, but performed poorly on election data from 2010.[3]

---

[1] Sak, Haşim, Andrew Senior, and Françoise Beaufays. "Long short-term memory recurrent neural network architectures for large scale acoustic modeling." *Fifteenth annual conference of the international speech communication association*. 2014.

[2] Tumasjan, Andranik, et al. "Predicting elections with twitter: What 140 characters reveal about political sentiment." *Fourth international AAAI conference on weblogs and social media*. 2010.

[3] Gayo-Avello, Daniel, Panagiotis Takis Metaxas, and Eni Mustafaraj. "Limits of electoral predictions using twitter." *Fifth International AAAI Conference on Weblogs and Social Media*. 2011.

We took these research studies into consideration while building our models, and chose the model that had the highest accuracy rate to make our sentiment predictions and focused on tweets involving only the November 20th Democratic debate.

DATA COLLECTION

In order to implement supervised learning techniques, we searched for a training set that contained pre-classified tweets. The sentiment classification models were trained on a dataset we found on Kaggle. The Sentiment140 dataset that we obtained from Kaggle included a .csv file that consisted of exactly 1,578,612 tweets that were classified as 0 (for a negative tweet) or 1 (for a positive tweet). Out of the 1,578,612 tweets, 788,435 tweets were classified as negative and 790,177 were classified as positive. Classifier algorithms, like decision trees and logistic regression, tend to have a bias towards classes that have a greater number of instances. The sentiment140 dataset is balanced, which allowed us to build predictive models that perform well by the AUC measure or rank-order. After selecting the best classification model, our objective also included making predictions for the 2020 presidential elections. We collected tweets before and after the November 20th, 2019 Democratic Debate for each of the ten candidates that participated.

Using the 'tweeterscraper' library in python, the tweets occuring 24 hours before and 24 hours after the November 20th debate were collected from the Twitter Developer API. For each of the participating debate candidate, tweets containing candidate's name or their campaign hashtag were collected. For example, all the scraped tweets for candidate Andrew Yang must contain either 'Andrew Yang' or 'Yang2020' in the tweet. After collecting tweets for each candidate, they were passed through the text-cleaning pipeline similar to the one used to clean the House and Senate data. After tokenization, non-ASCII characters, numbers, punctuation and stop words were removed from the tweets. From the cleaned version of the data, tweets occuring before 9:00 AM ET on Nov. 20th, 2019 were saved in a *before* dataset and those occurring after were saved in the *after* dataset. Below is a plot of frequency of tweets in our cleaned dataset for each candidate participating in November 20th debate.

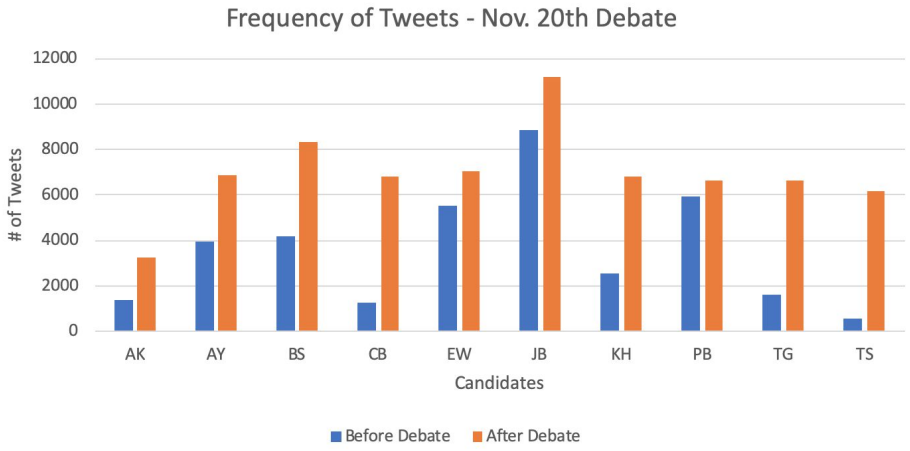


Figure 3. Number of Tweets Collected Before and After November 20th, 2019 Democratic Debate. Candidate names are abbreviated by their First and Last initials.

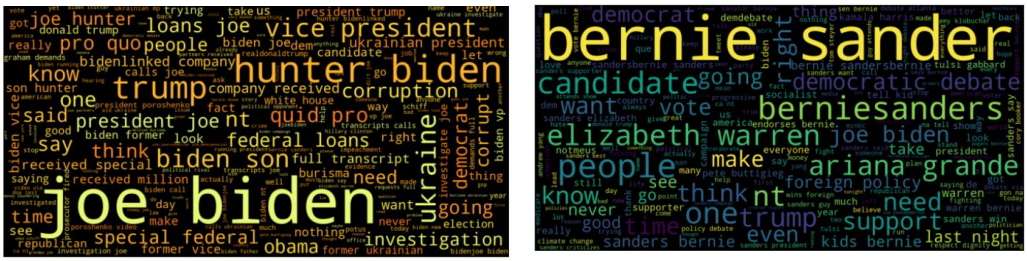


Figure 4. Tweet Visualizations

In the above word clouds, we can clearly visualize the twitter data that we collected. These word clouds consist of the most common words used in the tweets after the Democratic debate. These visualizations provide insight on the most common terms used on social media with regard to each candidate. In Joe Biden's word cloud, some of the common words include "ukraine", "investigation", "quid pro quo", and "corruption". In Bernie Sanders' word cloud, a rival candidate, "elizabeth warren", and a pop star's name, "ariana grande", indicative of her recent endorsement.

MODEL BUILDING

One machine learning approach to a text classification problem involves solving the research question by training the classifiers on a labeled data set. To split the data set, we used the data splitting functions in scikit-learn on python. The data set was divided into three sections. A training set, which was the sample of data that we used to fit the logistic model, a validation set, which was used to evaluate the training set and update hyperparameters, and the test set, which was used to provide an unbiased result of the final model fit on the training data set.

1. Train Set: consisted of 1,547,039 entries with 49.93% negative, 50.07% positive
2. Validation Set: has a total 15,786 entries with 50.75% negative, 49.25% positive
3. Test Set: consisted of a total of 15,787 entries with 50.22% negative, 49.78% positive

Before building models, we used TF-IDF vectorizer, as described in the first section, to convert each token of word into a vector. The TF-IDF vectorizer was used to extract features from the tweets in the train, validation, and the test set. Using the validation set to evaluate the performance of the Logistic Regression model, we ended up with an accuracy rate of 82.30%. For the Sentiment140 data set, the null accuracy rate was equal to 50.75%. The Logistic Regression model performed 29.58% more accurately than a model that would always predict every tweet in the validation set as negative. The following figure is the Receiver Operating Characteristic curve for the logistic model. The area under the curve is equal to 0.902.

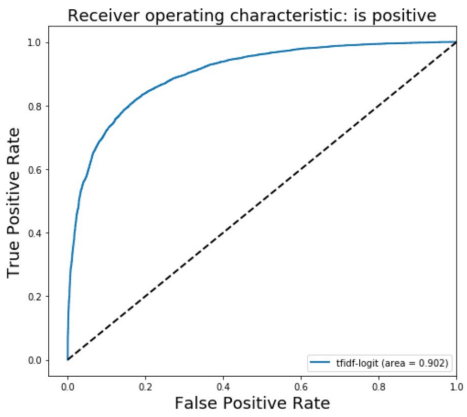


Figure 5. Logistic Regression Receiver Operating Characteristic (Tfidf-logit area = 0.902)

Since the Logistic Regression model resulted in the best accuracy rate, we performed 5- fold cross validation to compute the precision, recall and f1 scores for the negative and positive sentiment classifications. The average accuracy rate across the 5-folds was equal to 82.18% (with a standard deviation of +/- 0.11%), the average precision was equal to 82.19% (with a standard deviation of +/- 0.10%), the average recall was equal to 82.18% (with a standard deviation of +/- 0.11%) and the average F-1 score was equal to 82.17% (with a standard deviation of +/- 0.11%).

The next classification model we considered was the Naive Bayes Classifier. It assumes that the instance of a specific feature in a class (negative or positive) is unrelated to the occurrence of another feature. However, words in a tweet may not always be independent of each other. When evaluating the performance of the Multinomial Naive Bayes algorithm, we found that it classified the sentiment of tweets with a high accuracy rate. ScikitLearn also provides a pipeline that allowed us to implement the Multinomial Naive Bayes classifier and grid search for optimal parameters easily. Using 10 fold cross validation, we were able to grid search over 10 different combinations of parameters with a different test set each time. The highest accuracy rate with 10 fold cross validation was equal to 0.707 using {bow_ngram_range: (1,1), 'classifier_alpha': 0.01, 'tdidf_use_idf: False.}The CountVectorizer function from SciKit Learn was used, which converted the dataset of tweets into a matrix of token counts. Using the second model (with an accuracy rate of 0.707187), we predicted the sentiment of the test set. The accuracy rate was equal to 0.7195, the weighted average precision was equal to 0.72, the weighted average recall was 0.72, and the weighted F-1-score was 0.71.

Model	Accuracy Scores
Logistic Regression	82.30
Linear SVM	81.96
Linear SVM with L1- Feature Selection	81.95
Multinomial Naive Bayes	80.33
Bernoulli Naive Bayes	79.34
Ridge Classifier	81.84
ADA Boost	60.76
Perceptron	75.69
Passive-Aggressive	78.82
Nearest Centroid	72.14

Table 1. Various models trained on the training dataset for Sentiment Analysis along with accuracy scores

We built two SVM classification models using the training set and evaluated the models on the cross validation set with 15,786 observations.

1. Support Vector Machines (with a linear kernel)
2. Support Vector Machines (with a linear kernel and L1 based Features selection)

We found that the accuracy rate was higher for the Linear SVM with L1 based Feature Selection model compared to the Linear SVM model. The Linear SVM had an accuracy rate of 81.96%, which meant that it predicted 31.20% more accurately than a model that predicted all of the tweets to be negative. The Linear SVM with L1 based feature selection had an accuracy rate of 81.95%. We also considered other models as listed in the Table below but Logistic Regression yielded the best accuracy score of 82.30.

Sentiment Predictions

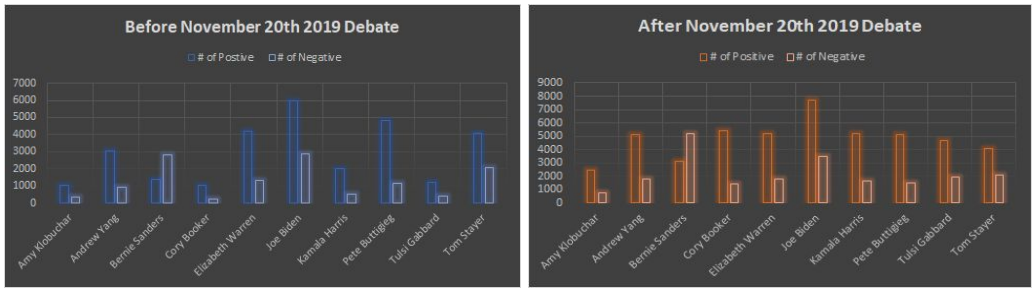
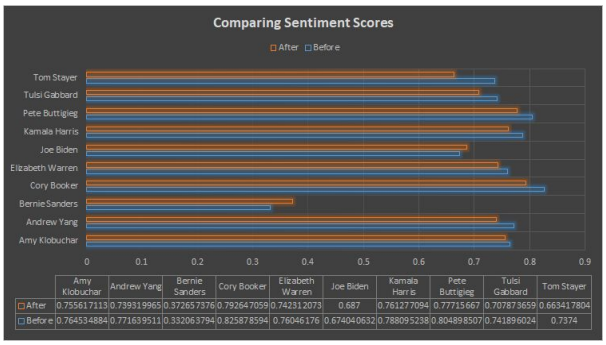


Figure 6. Distribution of Positively and Negatively Classified Tweets Using Logistic Regression Model Before and After Debates

Sentiment Scores



	Amy Klobuchar	Andrew Yang	Bernie Sanders	Cory Booker	Elizabeth Warren	Joe Biden	Kamala Harris	Pete Buttigieg	Tulsi Gabbard	Tom Stayer
After	0.755617113	0.739319965	0.372657376	0.792647059	0.742312073	0.687	0.761277094	0.77715667	0.707873659	0.663417804
Before	0.764534884	0.771639511	0.332063794	0.825878594	0.76046176	0.674040632	0.788095238	0.804898507	0.741896024	0.7374

Figure 7. Sentiment Scores (Percentage of Positive Tweets)

CONCLUSION

This project allowed us to learn more about how text classification works in the field of opinion mining. Since the NLP is one of the hottest topics being researched in the field of Machine Learning, we enjoyed conducting our own project on a topic with an immense growth over the next few years. Although we used Twitter data, applications of NLP are far stretched, in common object in our homes such as Alexa, prompting us to consider NLP as a unique research topic. In the first section of the project considering classification of tweets as Democrat vs. Republican, we learned that several traditional models with high accuracy can be built, however, if we want to consider sequence of words in the text to

be important, we should consider an RNN with LSTMs to take care of the long-term dependencies between text structures. Similarly, in the second portion, we learned the strength of machine learning in today's political outlook in detecting sentiment particular any Presidential candidate. Building several tradition models and performing standard data analysis yielded some very interesting results such as tweets for Bernie Sanders were pretty negative compared to any other candidate.

Some of the practical applications we would take on in future endeavors include predicting the results of the 2020 presidential election by opinion mining to find the sentiment people display towards candidates in the weeks leading up to the election. Currently, we are looking at data after one debate; however, by looking at this trend over time we will able to see the momentum a candidate has. In order to do so, we would also add additional features to our data like retweets, replies, likes and hashtags. We are currently only looking at the text as a predictor, but by looking at multiple additional features, the prediction scores are likely to go up as a result of knowledge on the popularity of the candidate. We are also thinking about limiting the location of tweets to the United States only to take make sure we're only taking into account the sentiment of those who will be voting. When classifying a tweet as a Republican or a Democrat, we noticed some huge jumps in the accuracy when going from the Senate to the House. This was most likely due to our sample sizes, but looking at other characteristics like formality of tweets and reservations that Senators might have when tweeting will provide some more insight on this disparity.