Politweet: Predicting Party Affiliation Using Tweets

Aaron Chan, Andrew Tate, Austin Kolander, Jonathan Dutson
CS 478, Winter 2018
Department of Computer Science
Brigham Young University

Abstract

We use a variety of machine-learning models to predict party affiliation using Twitter tweet text. Tweets were processed using a Doc2Vec model and sentiment analysis to create a numerical vector to use for model training. We used tweets from current United States Senators and Representatives to train our models. Our Support Vector Machine model achieved the highest accuracy (79%), comparable with similar tweet party prediction studies. We test the model on a set of tweets from influential Americans and find that it does not perform well on apolitical tweets. We suggest that future work should include identification of apolitical tweets and a larger data set size.

1 Introduction

Twitter is becoming increasingly important for politicians who want to develop a social media presence to increase their public visibility. The platform can provide politicians with direct contact with voters, free advertising, fundraising, and a sense of public opinion regarding important political issues. Tweets can be highly political, and some believe that Twitter contributes to the divisive partisanship prevalent in the United States government today [Cowen, 2019]. We wanted to explore how we could use machine learning to predict party affiliation based on an individual's Twitter activity. Predicting the political affiliation of politicians is a natural starting point for a generic prediction model, since politicians are more likely to tweet about political issues, and labeling a politician's political party is easier than labeling the political parties of other Twitter users, since in the Unites States most politicians align themselves with a major national party.

Initially we planned on using our model to rate each tweet with a political ideology score. However, we realized that providing labels for such a model would be challenging, so we decided to instead classify each tweet as simply liberal or conservative. This made the task of labeling tweets for politicians much simpler.

A general political party prediction model could be attractive to Twitter advertisers, allowing them to target ads at individuals based on political ideology. Twitter could use such a model to provide recommendations about who a user may like to follow. A successful model would likely be transferable to other social media platforms with minimal adjustments. Facebook already uses a variety of data (including pages liked, personal information, places a user checks in, and the predicted political preference of a user's friends) to attempt to predict user political ideology [Facebook, 2019]. A political prediction model based on text a user could augment existing models, allowing for better ad-targeting.

However, there are ethical questions when considering providing content to Twitter users based on political ideology. Twitter has been found to act as a political echo-chamber, reinforcing individuals political beliefs and providing few opportunities for exposure to new or opposing ideas [Garimella *et al.*, 2018]. A political party prediction model could be used to combat this trend if Twitter chose to deliberately provide users with some exposure to tweets that may not align with their political ideology.

In this paper we describe the methods we used to create a party-prediction model, the areas our chosen model was successful and the areas it failed, suggestions for future work, and the key takeaways we learned while completing this project.

2 Methodology

To collect the data we needed, we first had to assemble the list of politicians that we want to query. We found ideology ratings for both the House of Representatives and the Senate from `https://www.govtrack.us/congress/members/report-cards/2018/`, so we constructed a crawler to obtain the ratings and put them in a JSON file. These ratings gave each politician a score from 0 to 1, with 0 being the most liberal and 1 being the most conservative. To simplify our training, we rounded each score to either 0 or 1. The rounded scores became our labels for the politician tweet data set. This method allowed us to classify independent senators and representatives as liberal or conservative. We recognize that this labeling method is imperfect, since all tweets from the same politician are given the same rating. Centrist politicians would be more likely to post tweets that might not align with their political party. However, we determined that grouping tweets from the same politician together would be a simple and mostly effective way of labeling our data.

We then obtained a list of politician Twitter handles (found by a combination of Internet postings of politician handles and also manual gathering), and merged these two lists. Finally, we wrote a script that utilized the Twitter API to retrieve tweets from each politician to form our dataset. We then realized that some of the tweets gathered were truncated at a certain number of characters, so we constructed yet another web crawler that followed links to the original tweets and recovered the missing part.

We ran several preprocessing steps on our data, including cleaning the dataset by removing several unnecessary features (timestamp, geographic location, etc). We removed stop words and made text lowercase as well, for consistency. We then transformed the texts from tweets and hashtags into more meaningful document vectors using a pretrained model trained on the English Wikipedia corpus, which we hoped would capture the semantic meaning of the text to make tweets more comparable.

In summary, the steps to obtain the full dataset included:

1. Manually assembling web pages that contain ideology data, Twitter handles, and names for each politician in our training set.
2. Creating a crawler to retrieve ideology statistics, Twitter handles, and names for each politician from the web pages we assembled.
3. Creating a crawler that uses the Twitter API to retrieve tweet data for each politician, matching on name and Twitter handle.
4. Creating a crawler that fetches long tweet text ($>$ 150 words) directly from the URL and merges it with our dataset so far.
5. Python scripts to merge the scraped data into a single JSON file.
6. Running our pretrained Doc2Vec model on each tweet text and storing the resulting 300 length vector in the dataset.
7. Outputting to a CSV file.

2.1 Data Set

Initially our data set consisted of: the favorite count of the tweet and the results of our Doc2Vec model (300 numbers ranging from 0 to 1 that represent the semantic meaning of the tweet text). We stored our data in csv files that ranged from 0.5GB to 1.5GB large and did our data loading in Python using the pandas library.

In the following table we show two example instances of the raw data we scraped, merged and filled in. Note that although we show two politicians in the table, it is not the politician that defines a row; it is a tweet. We constructed our dataset to have, for each tweet that we scraped, the politician's name, ideology rating, tweet text (omitted from example instances for brevity), vector produced by our doc2vec model, and Twitter handle.

Politician	Ideology	Doc2vec	Handle
Bernie Sanders	0.008	$\begin{bmatrix} 0.122 \\ 0.359 \\ \vdots \end{bmatrix}$	@SenSanders
Trey Gowdy	0.613	$\begin{bmatrix} 0.567 \\ 0.251 \\ \vdots \end{bmatrix}$	@tgowdysc
$\vdots$	$\vdots$	$\vdots$	$\vdots$

Table 1: Data used

2.2 Selected Models

We decided to run initial tests using the following models:

- Decision Tree (SKlearn implementation)
- Naive Bayes
- K-Means with K = 2
- Support Vector Machine with an RBF kernel (SVM)
- Multilayer Perceptron with 3 hidden layers with 300 nodes each (MLP)

In particular, we chose decision tree and naive bayes for a baseline comparison, k means because we hoped there would be two major clusters, support vector machines because we read that they work well with natural language models, and multilayer perceptrons because of their flexibility and accuracy.

We used Python 3 and Scikit Learn for our testing framework and created a series of scripts that allowed models to be easily loaded in as modules.

3 Initial Results

To initially test our model we randomly divided our full dataset into a training and test set (80% for training, 20% for test). We ran the data across our array of models that we had chosen and found the following results.

Model	**Accuracy**	**MSE**	**Training time (s)**
Decision Tree	57.1%	0.429	34
Naive Bayes	58.9%	0.411	1
k-means	43.7%	0.563	23
SVM	73.5%	0.265	2868
MLP	68.8%	0.312	18601

Table 2: Results for Initial Dataset

In our initial results, SVM turned out to have the highest accuracy on the test set (73.5%) while k-means actually had the lowest accuracy at 43.7%. This seemed surprising: k-means seems suited to clustering Doc2Vec vectors because all the features are continuous so a similarity between documents should imply a lesser euclidean distance. All the same, these were our results, and they provide an illustration of the no free lunch theorem in that models that have performed well in past

situations didn't perform as well in this one, while models we didn't necessarily expect to perform extremely well actually did, as in the case of the SVM.

We felt like SVM and MLP seemed like the most promising models. Initially we were pleasantly surprised that we were able to achieve above 70% accuracy where the baseline for the dataset was roughly 55% accuracy (if the model labeled everything as conservative). It was hard to get an intuition for the reason why our SVM was performing well, however. We decided that a good way to start going about improving accuracy would be to derive additional features.

4 Feature and Model Improvement

4.1 Data Improvement

We recognized that our Document to Vector model was generally good at figuring out which topics were related, but we worried that a large part of making the classification was deciding what kind of emotion was behind the text meaning. To address this, we imported a pretrained sentiment analyzer called VADER which was specifically trained on data from social media posts. VADER output the positive rating, the negative rating and the neutral rating of the given text. All of these values are between 0 and 1. We added sentiment analysis to our preprocessing script and computed these 3 values for each of the tweets in our dataset. Below we listed our changes in accuracy as a result of adding sentiment analysis metrics:

Model	Sentiment	No sentiment
Decision Tree	57.1%	56.8%
Naive Bayes	58.9%	59.1%
k-means	43.7%	56.7%
SVM	73.5%	73.2%
MLP	68.8%	68.3%

Table 3: Improving results via sentiment analysis

In addition, our crawler only grabs the latest 250 tweets from each politician, so we recognized that in the weeks following the first scrape of data that we could do another scrape of data and augment our dataset. We ran our crawlers over night and found the following results on our second dataset.

Model	First Dataset	Second Dataset
Decision Tree	57.1%	56.4%
Naive Bayes	58.9%	58.3%
k-means	43.7%	56.2%
SVM	73.5%	73.5%
MLP	68.8%	68.7%

Table 4: Comparing results between two data scrapes

We figured that we could perhaps achieve even better results if we combined our datasets together. We found that indeed our accuracy increased in both test and training set, however our training time also practically doubled. Training the MLP practically takes the whole afternoon. Nevertheless, we were pleased with the results, which we have shown in the following table.

Model	Accuracy	MSE	Time to Train (s)
Decision Tree	60.4%	0.396	81
Naive Bayes	58.9%	0.411	1
k-means	56.5%	0.435	39
SVM	79.0%	0.210	12097
MLP	77.4%	0.226	18241

Table 5: Final results

4.2 Model Improvement

To improve our models we built a small framework to generate and measure accuracy on a test set for various incarnations of the models, each with tweaks to its model-specific parameters.

For example, we tested 4 different basis kernels for our support vector machine, including radial basis, polynomial, linear, and sigmoid. We found that the linear and sigmoid performed significantly worse than RBF and polynomial. We initially performed our analysis with RBF so we decided to keep it.

We also tested multiple configurations of hidden layers and numbers of nodes for our multilayer perceptron. Even though we tried up to 6 hidden layers with varying number of nodes, we found that our initial educated guess of three hidden layers of 300 nodes each actually performed better than the other configurations we tried. Part of our limitation in choosing parameters for the MLP was simply how much time it took to train. We therefore tested high numbers of layers and nodes only a few times before deciding that we were satisfied with our original configuration.

5 Final Results

5.1 Politicians

Our final model achieved 79% accuracy in identifying the political party of politicians using Twitter. This is similar accuracy to what others have achieved in similar work [Colleoni *et al.*,]. We achieved this accuracy using SVM on our expanded data set. While SVM took significantly longer to train compared with methods such as naive bayes or decision tree, it achieved more than a 20% increase in accuracy.

5.2 Other Popular Twitter Users

To really test our model, we hand collected a separate data set consisting of previous presidents and other politicians, as well as those unrelated to politics such as actors, actresses, musicians, etc. We didn't expect the model to perform particularly well on tweets that did not have political content, but we wanted to see what would happen.

To gather the data, we used a modified version of our original API/Screen Scraper. We did not have ideological ratings for our data so some minor modifications were needed. The screen crawl took a couple hours to process all the data.

For our data set, we wanted people who are influential to the American lifestyle. We chose previous presidents,

businessman, actors, singers, news anchors, and authors. We wanted a group of individuals that have shaped modern American culture. We wanted to see if our models could reasonably predict these people political preferences.

We found that overall the model performed fairly poorly on our generalized data set (which is what we expected). It labeled 71% of the tweets as conservative, and every individual we measured had a majority of conservative tweets, although every person had some tweets rated to be liberal as well. We hypothesize that our model's poor performance including the following reasons:

1. The data baseline for our initial data set is mostly conservative, so it turns out that when the model doesn't know how to classify the tweet, it is classified as conservative. Therefore, most politicians regardless of their ideology are classified as conservative overall. A better way to look at the results is to go by the individual tweets.

2. Since the model is trained on politicians, and other people that we observed in our new test set had many tweets without particularly political content, our model just labeled those tweets conservative.

3. In addition, many of the tweets that we retrieved were very small, such as "had a good day" as one tweet. These tweets had to be classified one way or another, so they were classified as conservative even though anyone looking at the tweet would say it's too simple to say.

4. Since Twitter is a social media platform, emojis are pretty ubiquitous. They are distinct from words because they are single characters that carry a lot of (generally sentimental) meaning. We're not sure how these emojis affected our doc2vec output (if they did anything significant at all). However we are assured by the documentation that VADER, our sentiment analyzer, does support expressive emojis. So, at least our sentiment score feature picked up something from emojis.

5. While sentiment analysis can be a helpful tool, our model may have had difficulty using it in combination with our Doc2Vec vector. For example, there could be two tweets (one liberal and one conservative) about the president of the United States with very similar content, but a different sentiment. Unless the sentiment value is given a very high weight, the two tweets may be classified together because of the similarities in their Doc2Vec vectors.

We include a selection of tweet texts and predictions in table 6 to provide an example of how our model classified tweets. The table shows one tweet rated as conservative and one rated as liberal for five individuals. We included a wide variety of tweets, both political and apolitical, to show examples for some of the difficulties described above.

6 Future Work

Taking the following steps might result in a more generalizable, accurate and interpretable model in the future.

6.1 Identifying Apolitical Tweets

For future work, we would like to expand our model to identify apolitical tweets. We found that many of the tweets in our training set, which we labeled in batches using the ideological rating scores, were not political in nature at all. Unfortunately, manually labeling our training set of over 100,000 tweets was infeasible.

We hypothesize that our test set of tweets from influential individuals did more poorly than the politician set because so many of the tweets in the test set were not necessarily political. While identifying apolitical tweets could be very challenging, it would likely improve the accuracy of identifying political tweets.

6.2 Increasing Data Set Size

We are confident that our model could improve with an increase in data across a longer time line. Increasing from a single week of data to two weeks of data gave us a large increase in accuracy. If our data spanned multiple months, it could likely learn to classify tweets based on the common political themes of the time.

The Twitter API limited our retrieval of tweets to the 250 most recent tweets for that individual. We could gather more data by scraping data every two weeks for a period of time. Merging the data together to drop duplicates. This would gather more data for us to train on. Additionally, we should collect a data set of non political tweets. Labeled data that is not political in nature will allows us to train new models on liberal, conservative, and neutral outputs.

7 Conclusion

Overall, we thought it was a fun and educational experiment. We feel that our model would not perform very well in the wild because of the problems listed under the final results, but it was an enjoyable learning experience. Some of the big takeaways were:

1. We now know how to gather data to apply machine learning to other real world problems. One of the biggest challenges in this project was building the tools necessary to get the data we needed. Sometimes we tried to do it copy/paste but after a few entries realized we need a bot that could scrape the data from the website and merge it into other data sets we had scraped from the web. This was tricky and was a valuable learning experience that will benefit us in future applied machine learning projects.

2. It can be challenging to create machine learning models that work on data sets different than the ones they were trained on. While our model achieved 79% accuracy predicting the political ideology of the tweets of elected representatives, it did not perform well on tweets from other well known politicians and celebrities.

3. Sometimes the best-performing model is a surprise. While one might expect a clustering model that makes use of cosine similarity or an MLP to perform the best, an SVM might perform significantly better than anything else (as it did for us). This shows how important

Individual	Tweet Text	Classification
Donald Trump	Facebook Google and Twitter, not to mention the Corrupt Media, are sooo on the side of the Radical Left Democrats. But fear not we will win anyway just like we did before!	Conservative
Donald Trump	93 Approval Rating in the Republican Party. Thank you!	Liberal
Taylor Swift	This week has been unforgettable. I love you guys. Thanks a million 😉	Conservative
Taylor Swift	❤️	Liberal
Bill Gates	This book is a must read for anyone interested in AI, machine learning, and machine vision.	Conservative
Bill Gates	This surprised me: theres more carbon in soil than in the atmosphere and all plant life combined. Heres what that means for how we fight climate change:	Liberal
Joe Scarborough	These Trump numbers from Michigan are horrible; They're just as bad as the ones released earlier today from New Hampshire.	Conservative
Joe Scarborough	I'm also confused as to why these self-righteous pundits are seizing on the conclusions of a man they slandered for two years as untrustworthy. Will they now apologize to Marine war hero and celebrated FBI Director Robert Mueller?	Liberal
Britney Spears	Can you believe that this album was released 20 years ago today?! I can't. Its been the journey of a lifetime, but I'm grateful for every moment. Getting to know you all over the years has been such an incredible experience. Thank you for all your support ❤️ BabyOneMoreTime20	Conservative
Britney Spears	Another gem from the ...Baby One More Time album!!! "Sometimes" will always have a special place in my heart. Can you believe the album turns 20 in January???!!! BabyOneMoreTime20 limited edition picture disc vinyl is out on Friday	Liberal

Table 6: A selection of test tweets and classifications

it is to remember the No Free Lunch theorem and not expect models that have worked in the past to work as well in a new situation. A researcher is in a bad position to make initial decisions about selecting models and parameters because he/she likely doesn't understand the nuances of the data from the outset. A researcher must be willing to make data-driven instead of assumption-driven decisions about model choice.

References

[Colleoni *et al.*,] Elanor Colleoni, Alessandro Rozza, and Adam Arvidsson. Echo chamber or public sphere? predicting political orientation and measuring political homophily in twitter using big data. *Journal of Communication*, 64(2):317–332.

[Cowen, 2019] Tyler Cowen. The twitter takeover of politics is just getting started, Feb 2019.

[Facebook, 2019] Facebook. About Facebook Ads. `https://www.facebook.com/ads/about/?entry_product=ad_preferences`, 2019. [Online; accessed 10-April-2019].

[Garimella *et al.*, 2018] Kiran Garimella, Gianmarco De Francisci Morales, Aristides Gionis, and Michael Mathioudakis. Political discourse on social media: Echo chambers, gatekeepers, and the price of bipartisanship. In *Proceedings of the 2018 World Wide Web Conference*, WWW '18, pages 913–922, Republic and Canton of Geneva, Switzerland, 2018. International World Wide Web Conferences Steering Committee.