Categorizing News Articles on Political Bias

Final Write-up

Authors: Dylan M. Crain (006276473), Yağmur Erhan (006206283)

1 Introduction

In the current political environment of the United States, the country appears to be more divided along party lines than ever before, in living memory. One possible explanation for this division is polarization created by the customization of news and social media feeds that suits the preferences of a given user. Over time, this over-saturation of one-sided content can be associated with radicalization towards one party or the other. Citizens on either side of the political aisle may possibly live in entirely different worlds of information consumption.

In an attempt to help combat this polarization and radicalism, the authors attempt a machine learning implementation that can distinguish news articles that belong to one political party or the other. The hope is that consumers being explicitly made conscious to the political leaning of each article they peruse may lead to at least an understanding of a given individuals biases. This understanding may encourage a reader to at least be more open-minded to other perspectives — knowing how one-sided their consumption typically is.

The initial question to ask in an implementation of such an approach is how to objectively assign party affiliations to a given news article based on the words chosen by the author. In this study, we have decided to train the machine learning algorithms implemented using compiled Tweets from Congressmen and Congresswomen. This makes objective training more straightforward, since the vast majority of Congresspeople claim to belong to one party or the other, e.g., Republican (R) or Democrat (D). Furthermore, members of the House of Representatives are targeted, since (due to their short terms) it is expected that they will have their fingers more on the pulse of their districts, so to speak.

Therefore, the training sets for the machine learning models are to be composed of either the word existence or frequency from a vocabulary for each Tweet from these Representatives. From here, the trained models are used to predict the political affiliations of selected news articles. Many learning approaches are investigated with three being chosen for this work. These three methods are: Naive Bayes with a Bernoulli event model, Naive Bayes with a Multinomial event model, and linear support vector classification.

2 Dataset

There are two sets of data required for this implementation. Firstly, the training data is to be gathered, categorized, and cleaned. This set is composed of the Tweets of Congresspersons. The secondary data set is composed of selected news articles that the trained machine learning algorithms are to categorize as either Republican or Democratic leaning. These articles need to be gathered and assigned labels.

2.1 Congressperson Tweets

Beginning with gathering the Tweets, the process was made much easier by a Github site[1]. This site provides a repository of Congressperson Tweets. This prevents the need to scrape the information from Twitter™ directly. However, the repository[1] is limited. Evidently, Twitter has changed its API fairly recently, meaning that the most recent Tweets in the repository are only up to April 1st of 2021. This is somewhat limiting. However, the 117th Congress was just inaugurated on January 3rd, 2021. This means, that three months of Tweets can directly be used with the current Congress.

The data stored in the repository[1] is stored in JSON files. Scripts written by the authors retrieve the tweets from a given date range and converts all of the information to a CSV file format. Furthermore, the only data of interest is the Tweet text, as well as the Twitter handle of the user. Overall, for the three months investigated, a total of $103,852$ Tweets are available for training.

With the Tweets gathered, they now need to be assigned political leanings based on the declared affiliations of their authors. This is

performed by taking advantage of the table displayed in the provided website[2]. This site tabulates the Congressmen and Congresswomen with their political affiliations as well as their Twitter handles. The data set described was cross-checked with a more widely available source[3] table for the Twitter handles of the 116th Congress. With this information gathered, the collected Tweets can now be assigned appropriate political affiliations.

2.2 News Articles

With the Tweets gathered for training, the next step in data collection is to attain news articles to be categorized by the algorithm. The goal in this process is to fairly evenly collect articles from each suspected political party that was written in our valid time-frame, i.e., from January to April 2021. The more challenging portion of this collection is objectively assigning "true" political affiliations to these articles. This was accomplished by polling 23 individuals with 120 news articles on what affiliation they thought best suited each work. The party having the highest average of respondents was chosen as the label for each of the articles. From these 120 articles, 100 were chosen such that there is an even split on the labels, i.e., 50 articles assigned 'D' and 50 assigned 'R'.

3 Methods

There are two primary algorithms used in this work. The first is the Naive Bayes, and the second is the Support Vector Machine (SVM).

3.1 Naive Bayes Classifier

For the Naive Bayes Classifier, the primary assumption made is that any two features (x_i, x_j) are conditionally independent *given* the label y. This effectively means that the now:

$$p(x_1, x_2, \ldots, x_d|y) = \prod_{j=1}^{d} p(x_j|y),$$

where $x \in \mathbb{R}^d$. This greatly simplifies the computations necessary to classify. Thus, maximizing the joint likelihood:

$$\mathcal{L}(\phi_y, \phi_{j|y=1}, \phi_{j|y=0}) = \prod_{i=1}^{n} p(x^{(i)}, y^{(i)}),$$

which results in the following parameters (when Laplace smoothing is used for the Bernoulli event model):

$$\begin{aligned}
\phi_y &= \frac{\sum_{i=1}^{n} 1\{y^{(i)=1}\}}{n} \\
\phi_{j|y=0} &= \frac{1 + \sum_{i=1}^{n} 1\{x_j^{(i)} = 1 \text{^} y^{(i)} = 0\}}{2 + \sum_{i=1}^{n} 1\{y^{(i)} = 0\}} \\
\phi_{j|y=1} &= \frac{1 + \sum_{i=1}^{n} 1\{x_j^{(i)} = 1 \text{^} y^{(i)} = 1\}}{2 + \sum_{i=1}^{n} 1\{y^{(i)} = 1\}}.
\end{aligned}$$

Furthermore, with these parameters fit by training the model, the predictions can be made from the following:

$$\begin{aligned}
p(y = 1|x) &= \frac{p(x|y = 1)p(y = 1)}{p(x)} \\
&= \frac{\left(\prod_{j=1}^{d} p(x_j|y = 1)\right) p(y = 1)}{\left(\prod_{j=1}^{d} p(x_j|y = 1)\right) p(y = 1) + \left(\prod_{j=1}^{d} p(x_j|y = 0)\right) p(y = 0)}.
\end{aligned}$$

The theta parameters determined above are for the Bernoulli event model, i.e., when each word in the vocabulary either exists in the sample or does not, $x_j \in \{0, 1\}$. However, for the Multinomial event model for Naive Bayes (the second algorithm in this work), the features represent the vocabulary index for each word in the Tweet. Thus, the feature vectors are now a type of sum of the times that a word appears in a message. Thus, for a vocabulary (V), the Multinomial event model of Naive Bayes is altered in the following parameters:

$$\begin{aligned}
\phi_{j|y=0} &= \frac{1 + \sum_{i=1}^{n} 1\{x_j^{(i)} = 1 \text{^} y^{(i)} = 0\}}{|V| + \sum_{i=1}^{n} 1\{y^{(i)} = 0\} d_i} \\
\phi_{j|y=1} &= \frac{1 + \sum_{i=1}^{n} 1\{x_j^{(i)} = 1 \text{^} y^{(i)} = 1\}}{|V| + \sum_{i=1}^{n} 1\{y^{(i)} = 1\} d_i},
\end{aligned}$$

where d_i is the number of words in a given Tweet. Note that in this work, the Naive Bayes algorithms are coded from scratch.

3.2 Support Vector Machine

Support Vector Machines (SVMs) rely on maximizing the minimum geometric margin from the separating hyperplane. That is, the algorithm operates by finding a hyperplane separating the classes of data such that the minimum distance between the hyperplane and any point is maximized. From this simple definition, there are many steps, including Lagrange duality and optimizing the margin classifiers that are applied to get the final form of a SVM.

For the SVM used in this work — the Linear SVC — the optimization function is given by:

$$\min_{w,b} \frac{1}{2} w^T w + C \sum_{i=1} \max(0, y_i(w^T \phi(x_i) + b),$$

where $\theta^T \phi$ is now given by $w^T \phi + b$. For this case, notice that there is no inner product for the kernel trick to be applied to. This is because the method described is linear. Furthermore, note that a SVM with a Gaussian kernel as well as the Sequential Minimal Optimization algorithms were attempted on the Twitter data. Both these methods had quite low accuracies compared to the three approaches just described, so they were dropped.

The C term in the previous expression is a hyperparameter that must be set. Note that C is a regularization parameter that applies weighted penalties to prevent non-separable data sets from having a few outliers damage the results. In summary, this term balances the desire to maximize the minimum margin and not being overly sensitive to outliers. By testing the method on a test set of Tweets for many different values of C, it was determined that $C = 0.06$ was optimum for this problem. Finally, note that, unlike the Naive Bayes, the authors utilize *scikit-learn*[4] to implement this Linear Support Vector Classification.

4 Results and Discussion

The first set of results pertains to splitting the Tweets into a training and test set to observe how each algorithm fairs when trying to predict a Tweet, instead of the possibly unrelated articles. These observation provided a benchmark, confirmation in accuracies, and the relationships between sentence structure in these short messages.

The second set of results stems from using all of the Tweet information to train each algorithm, which is then used to predict the political affiliations of the articles.

4.1 Testing with Tweets

To begin testing the algorithms on the Tweets, themselves, the data set is first randomly split such that 90% of the messages are used to train the models. This having been achieved, the model is first trained and tested with the Bernoulli Naive Bayes approach. This is done by directly training with respect to the text in the Tweets – with no more cleaning performed. With this training complete, the most indicative words for the Democratic party (D = 1) can be found by the following:

$$\log \frac{p(x_j = i|y = 1)}{p(x_j = i|y = 0)} = \log \left(\frac{P(\text{token} i|\text{Tweet is by Democrat}}{P(\text{token} i|\text{Tweet is by Republican})} \right).$$

Using this, for the case of Bernoulli Naive Bayes, the top seven words indicative of a Democratic author are: “#stopasianhate”, “@replloyddoggett”, “@joaquincastrotx”, “@repcasten”, “#americanrescueplan”, “http://healthcare.gov”, “#americanrescueplan.”. Note that #americanrescueplan occurs twice with the exception that the second occurrence has a period appended to it. Thus, it may

Table 1: Top words using the Bernoulli Naive Bayes for different vocabulary structuring.

All Words	No Punctuation	No #’s or ’s
#stopasianhate	stopasianhate	latino
@replloyddoggett	joaquincastrotx	nevadans
@joaquincastrotx	replloyddoggett	inequalities
@repcasten	repcasten	gerrymandering
#americanrescueplan	repchuygarcia	doggett

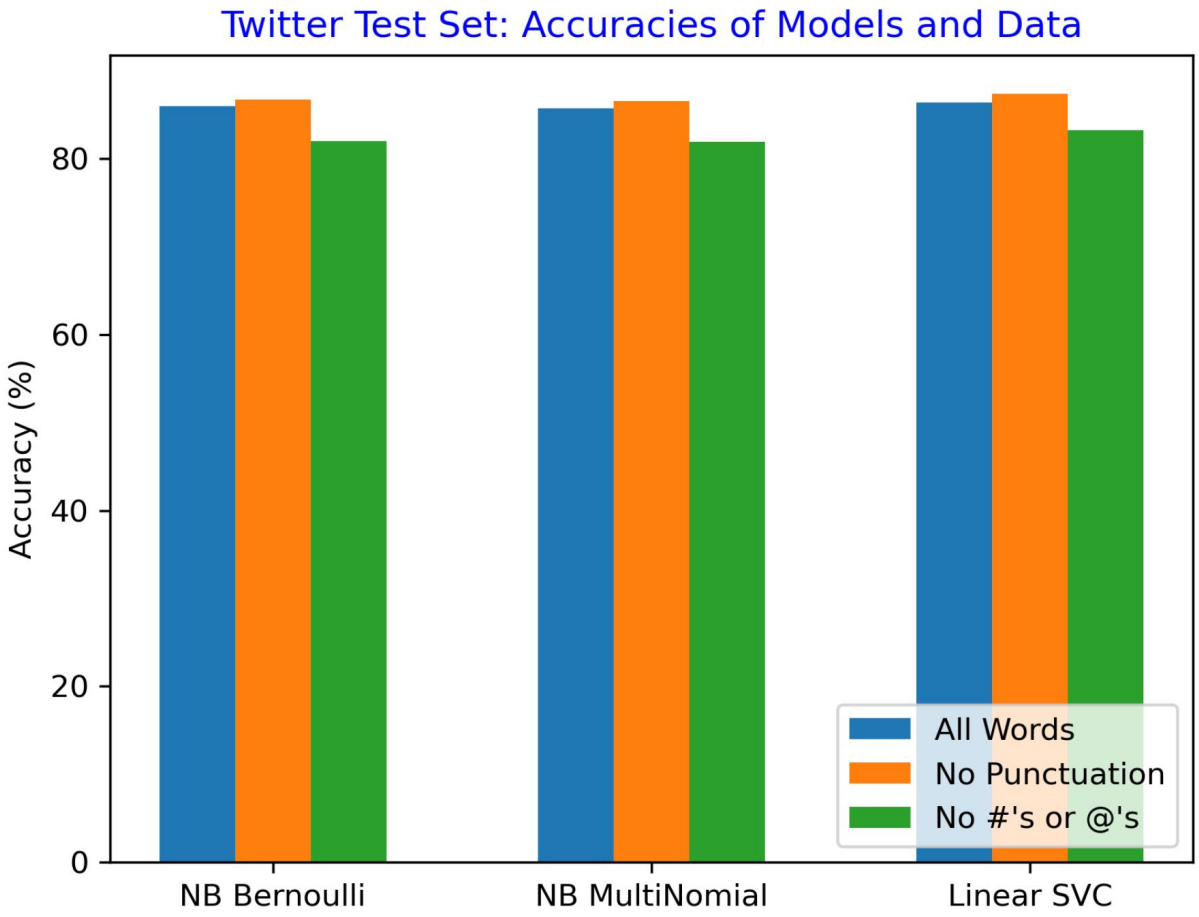

Figure 1: Accuracies for using 10% of the Twitter data as the test set. Note that all accuracies, for our models, are quite close to one another with SVC being slightly preferred. Furthermore, removing hashtags does not decrease accuracy by much.

Table 2: Accuracies of models trained with Twitter posts to predict news articles.

NB Bernoulli (%)	NB Multinomial (%)	Linear SVC (%)
58.0	55.0	63.0

be possible to attain better results if punctuation were removed. This proves true, since doing this adjustment during training results in a 86.74% accuracy as opposed to before when the accuracy was 85.96%.

With this adjustment, the top five words are now: stopasianhate, joaquincastrotx, repllowyddoggett, repccasten, and repchuygarcia. Notice that these words are still characterized by "#" and "@" parameters. These words are highly unlikely to appear in news articles. Thus, training the models on these extraneous terms, that happen to be very important in distinguishing Democrat writers from Republican, could prove harmful. However, it was concerning that since Tweet's rely so heavily on hashtags and at symbols, that removing the these expressions from the training may return useless results. Regardless, the elimination of these markers, along with all other punctuation as before, was performed. Fortunately, this adjustment resulted in an accuracy on the Twitter test set of 82.06%. The accuracy certainly went down compared to 86.74%, however, it is far lower of a drop than was feared. This implies that the model learns primarily from non-marker words, which was a surprise. Albeit a pleasant one. Furthermore, the top five words that are particularly indicative of a Democratic Tweeter is found in the third row of Table 1. Note that they make some intuitive sense. For example, "latinos" and "inequalities" are historically concerns of Democratic politicians.

With the determination of Tweet structure discussed, Figure 1 displays the accuracies to the Twitter test set using different combinations of vocabulary, as discussed above. In general, the Bernoulli Naive Bayes is superior to the Multinomial Naive Bayes approach; whereas, the Linear Support Vector Classification is superior to both of the other methods. However, the differences between the three are very minute. Briefly, we mentioned earlier that a SVM with a Gaussian kernel and an SMO were briefly implemented on the problem. The former had an accuracy of 75%, and the later was at 64%. Thus, the very close values of the three methods used is an encouraging indicator of a possible upper-limit on learning accuracy for these types of algorithms. This may, in fact be possible, as there are many moderate politicians on either side of the aisle that may message their constituents in a leaning that is sometimes to the left and some to the right. Thus, it is challenging to categorize their Tweets. Furthermore, these results indicate that, for this problem at least, the simpler model is performing better than the more complicated options.

4.2 Testing with Articles

With this analysis performed, the results for the real case, i.e., using Tweets to train a model to differentiate news articles is discussed. Each of the three models discussed are trained on the full set of Tweet data from Congressmen and Congresswomen for a three month period. Once the models are trained, they are applied to the set of 100 news articles to determine how well the Tweet training can match political bias in these sources. The results are displayed in Table 2. By a significant margin, the Linear Support Vector Classification beats out the two Naive Bayes approaches with an accuracy of 63% with the news articles. This result is not high as I would have liked. It is only just over the accuracy of randomly assigning labels to all of the news articles, i.e., 50%. However, this result is certainly better than performing the Naive Bayes approach, which had an accuracy of 58% and 55% for the Bernoulli and Multinomial, respectively.

These results imply that either:

1. The learning model, training size or model itself, is not sufficient, or
2. There is not enough information in Tweets from politicians to accurately assign political leanings in news articles.

Khan et al. (2019)[5] claims that problems like these are greatly improved whenever data is taken from many different sources, i.e., not all from Twitter. This may partially explain the poor accuracies when transferring over from Tweets to news articles.

5 Conclusions

In this study, we have discovered structure in Tweets from Congresspersons due to the removal of #'s and @'s still resulting in comparable learning algorithms to the cases where these supposedly important words are not included. Furthermore, it was determined that using the Tweets from the House of representatives alone is unlikely to yield exceptional results when predicting the political leaning of a given news article. Finally, it was found that, at least in this case, simpler models tend to result in superior results.

Future directions for this work have the potential to be enormous. One example is that the primary limiting factor in this study was retrieving and labeling the news articles. The amount of time needed to get poll assignments from a large group of individuals is large. In this work, only 100 articles were available with only 23 individuals voicing their opinions. It is very likely that there is some bias on the mean of their opinions on the leaning of the articles they read. Therefore, a direct method to move forward would be to improve upon this data collection.

6 Contributions

- Dylan Crain: Gathering Tweet data, code, report, and analysis.
- Yağmur Erhan: Gather news articles and send out poll for labeling.

7 References

1. Tweets of Congress. (n.d.). Tweets of Congress. Retrieved from https://alexlitel.github.io/congresstweets/
2. Congressional Social Media Handles. (n.d.). Triage Cancer-Finances-Work-Insurance | Triage Cancer. From https://triagecancer.org/congressional
3. Congressional 116th Tweet Handles. (n.d.). From https://www.sbh4all.org/wpcontent/uploads/2019/04/116thCongress TwitterHandles.pdf‘
4. scikit-learn: machine learning in Python — scikit-learn 0.20.3 documentation. (2019). Scikit-Learn.org. https://scikit-learn.org/stable/index.html
5. Khan, W. (2019). Predicting stock market trends using machine learning algorithms via public sentiment and political situation analysis [Review of Predicting stock market trends using machine learning algorithms via public sentiment and political situation analysis].