Predicting Party Affiliation of Twitter Users Using Individual Tweets

KEKOA WONG[*], University of Notre Dame
JULIANA RAMOS, University of Notre Dame
TIM BURLEY, University of Notre Dame
ROSS MCILVAINE, University of Notre Dame

A comparative analysis between machine learning classification methods in the field of computational social science. This project was completed during the 2020 election season for the data science course at the University of Notre Dame.

CCS Concepts: • **Data Science**; • **Computational Social Science**; • **Natural Language Processing**; • **Machine Learning**;

Additional Key Words and Phrases: ID3 Trees, CART Trees, Random Forest Trees, K-Nearest Neighbors, Naive Bayes, Convolutional Neural Networks, Multilayer Perceptron, Support Vector Machine, BERT, GLoVe Vectors, LSTM, TF-IDF

ACM Reference Format:
Kekoa Wong, Juliana Ramos, Tim Burley, and Ross McIlvaine. 2020. Predicting Party Affiliation of Twitter Users Using Individual Tweets. 1, 1 (November 2020), 6 pages.

1 INTRODUCTION

Social media has opened up a plethora of different research methods for computational social science. Twitter is a rich source for real time thoughts of many celebrities, influencers, and political figures. In fact, the platform has transformed the way politicians express their thoughts and opinions, allowing users the ability to consume information directly from a source, unfiltered through a publishing medium. Since the nature of political communication remains inherently text-driven, the composition, timing, and context are necessary components of the overall message, making it a challenging machine learning challenge that has the potential to provide insight on political commentary. Our project goal was to categorize tweets from United States politicians into their respective parties, focusing on Republicans and Democrats. Through this project, we compared the performance of different classification methods and their ability to categorize tweets of political figures into their respective parties while also weighing the effectiveness of group clustering methods. For this project, we utilized core data science principles and machine learning methods with the goal to create models that would accurately predict whether a given tweet was generated by a user who identified as "`Republican`" or "`Democrat.`" These models have been created using large datasets of tweets generated by public officials whose party affiliations were known. With the words, hashtags, and other data contained in a tweet, we applied different machine learning methods in order to categorize these tweets by the party orientation of the user who generated the tweet.

[*]All authors contributed equally to this research.

Authors' addresses: Kekoa Wong, kwong6@nd.edu, University of Notre Dame, Fitzpatrick Hall of Engineering, Notre Dame, Indiana, 46556; Juliana Ramos, jramos3@nd.edu, University of Notre Dame, Fitzpatrick Hall of Engineering, Notre Dame, Indiana, 46556; Tim Burley, tburley@nd.edu, University of Notre Dame, Fitzpatrick Hall of Engineering, Notre Dame, Indiana, 46556; Ross McIlvaine, rmcilvai@nd.edu, University of Notre Dame, Fitzpatrick Hall of Engineering, Notre Dame, Indiana, 46556.

2 RELATED WORK

Other computational social scientists have done previous research into the similarities and differences between twitter users who have identified as Republican or Democrat. [5] In one such project, a dataset that contained a collection of tweets from officials in the US House of Representatives was used, which happened to be similar to the one we had split into our training and testing data. The findings of this previous project highlighted some hurdles that we ran into during our project, including the difficulties in determining a complex group affiliation from a small text sample.
This previous research project also compiled a word cloud for the most common words used in its political speech datasets. When looking at the most popular words for both Republicans and Democrats representatives, we can see that both groups have similar high volume words. In particular, we find that both groups tweet about current events and stick to words that have positive connotations, which was interesting to note.

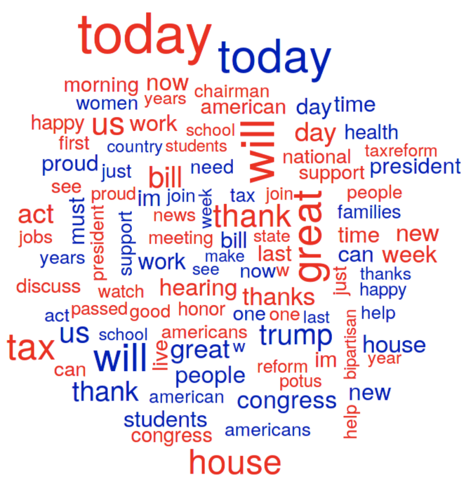


Fig. 1. Highest Volume Words

It is not until you begin to analyze less commonly used words that you begin to see some distinction between parties. Looking at this data, we can begin to distinguish differences in the policies or topics of the users.

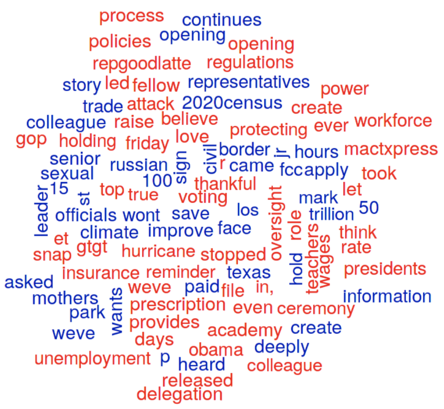


Fig. 2. Less Commonly Used Words

3 SOLUTION/METHOD

To explore the most effective method in classifying the party affiliation of the user, we built nine different types of machine learning models: ID3 trees, CART Trees, Random Forest Trees, KNN, MLP, SVM, Naive Bayes, Convolutional Neural Networks, and (Distil)BERT.
The three different trees methods were trained on four features that were extracted from the individual tweets. These features were sentiment score, language, GloVe dot product, and a boolean representing a retweet. Deciding on the features that would help trees classify the tweet by party preference was a very difficult task. Sentiment score was extracted so that the models would be able to understand the partiality the user may have toward certain topic. The GloVe dot product is the dot product of all the GloVe word representations in a tweet into a singular vector, creating a vector representation of the tweet topic. This dot product, in addition to the sentiment score, were both extracted in the hopes that the model would be able to recognize partiality toward tweet topics characterized through the vector representation. The other language and retweet features were chosen after examining the dataset by eye and using quick empirical assessment to judge whether or not the tweets could be classified by these features. We built three different tree models to test if the different branching methods would yield significantly different results.

With the exception of (Distil)BERT, the remaining classifiers were trained primarily using word-based and sequence-based feature extraction methods. For KNN, MLP, SVM, Naive Bayes, and one of the Convolution Neural Networks, we used TF-IDF as the feature extraction method, as it was the oldest and most suitable method for text data. The first four preceding classifiers were trained using the Sci-kit Learn library. For the second of the two Convolutional Networks, we used GloVe inside an Embedding layer. For the third we used an LSTM Layer. The final classification method we used was a transformer based method called BERT. We decided to use GloVe word embeddings, LSTM, and transformers primarily because of their usefulness for sequential input (LSTM and BERT) and their potential to provide deeper contextual information about the tweets that TF-IDF is unable to find. Each of the deep learning classifiers were assembled and trained with Keras and Tensorflow.

4 DATA AND EXPERIMENT SETTINGS

The two primary datasets we used came from Kaggle. One is a large set of tweets from members of the House of Representatives taken as recently as May of 2018. The second contains tweets from Donald Trump and Hillary Clinton in 2016 during the presidential election of that year. Both datasets are formatted as CSV files.

4.1 Data Cleaning

While the main brunt of the preprocessing was done using the tweet-preprocessor library, there were some features that we could extract from the original CSV file. Namely, the feature values for whether or not the tweet is a retweet, the username or "handle", and the object's label of "`Republican`" or "`Democrat`" were included in the original dataset. The rest of the features were extracted from the body of the tweet using the tweet-preprocessor library. We are able to extract hashtags and mentions from the tweet using the library, cleaning the tweet's body of all non-text objects. After cleaning the data, we had a list of lists, enclosed in a dictionary. Each tweet was in the form of a list containing the tweet's label, followed by its text, any hashtags and mentions (if present), a boolean value representing if the tweet is a retweet, and the user's handle.

4.2 Feature Extraction

Using this cleaned dataset, we performed two different forms of feature extraction. The first form of feature extraction was a word-based method used on the cleaned tweet words, with any non-text objects removed. The following were the four different methods of word-based feature extraction we used: TF-IDF scores, GloVe vectors, LSTM, and BERT.
TF-IDF scores were assigned to a given word based on a corpus that was compiled using all of the cleaned tweet words that we collected. Stanford's GloVe library are vector representations of words that are trained on large datasets that they have collected [6]. Our project collected the word vector representations that were trained on the twitter dataset (containing over two billion tweets) and translated the words in our cleaned tweets into vector forms.

LSTM is a Recurrent Neural Network based feature extraction method that takes a directional sequential input and calculates a feature vector that represents the entire sequence. A feature vector is calculated for each token of the sequence in a hidden state of the network and recalculated for subsequent tokens in subsequent hidden states based on prior tokens. The final hidden state of the network represents the entire sequence.[3] LSTM works really well with Convolutional Neural Networks for text classification and was used this way in this project. [1]

BERT is a transformer based neural feature extraction method that is very similar to LSTM in that it takes sequential input.[2] The primary difference is that BERT does not depend on the direction of

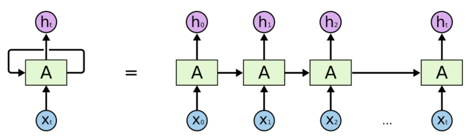


Fig. 3. Recurrent Neural Network

the sequence to create the feature vectors, but instead uses attention mechanisms which enable the token to focus on any other token in the sequence to calculate the feature vector while in a hidden state. There is a significant increase in calculations required for BERT compared to LSTM, thus making it very computationally taxing and virtually unusable without specialized hardware. DistilBERT is similar to BERT, but put through a distillation process, which means training a smaller transformer model to behave like a larger one.[7] While distillation makes the model less computationally expensive, training it is still intractable without specialized hardware.

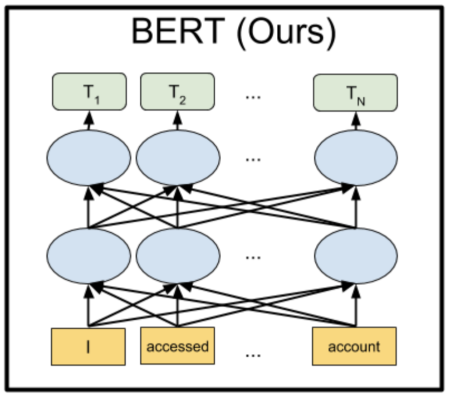
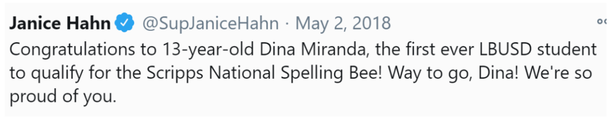


Fig. 4. BERT Architecture

The second form of feature extraction was a general tweet-based feature extraction method that obtained features from the tweet as a whole. These features included an overall sentiment score, the language of the tweet, a GloVe dot-product score, and a boolean describing the retweet status. The sentiment score of the tweet was calculated using a database of words from the Hedonometer website, which were rated using Amazon's Mechanical Turk service on a scale from 1 (sad) to 9 (happy). [4] We downloaded the Spanish and English datasets and calculated the tweet sentiment score by taking the sum of the individual sentiment scores of the words in the cleaned tweet. If a word was not in the database, it was simply ignored in the sum. The language of the tweet was detected using the langdetect library from python. Tweets that were in a language other than Spanish or English were removed from the dataset, as it would have been more difficult to calculate our sentiment score with more uncommon languages. The GloVe dot-product score was calculated by taking the combined dot-product of all the GloVe word vectors in the cleaned tweet. Once again, if a word was not in the GloVe database, it was simply ignored. Finally, the retweet boolean was included in the original dataset from Kaggle, so the process to extract this feature was minimal.

4.3 Experimentation and Case Studies

In this section, we will explore a few examples of the cleaned tweet text that our best models (KNN + TF-IDF, MLP + TF-IDF, SVM + TF-IDF, and Naive Bayes + TF-IDF) classified. One such example that all of these models accurately classified is the tweet text of "This week is National SmallBusinessWeek. I encourage everyone to shop local and support CA50 small businesses." This tweet is from Representative Duncan Hunter. Overall, it performs a call to action for people to shop in their local areas to support small businesses. The emphasis of the tweet on business may have been similar to the policy focus of other Republican representatives on Twitter, pushing all four of the models to correctly classify this user as a Republican.



Janice Hahn @SupJaniceHahn · May 2, 2018
Congratulations to 13-year-old Dina Miranda, the first ever LBUSD student to qualify for the Scripps National Spelling Bee! Way to go, Dina! We're so proud of you.

Fig. 5. An Example of a Party Ambiguous Tweet

This tweet is from Janice Hahn, a former US Representative and a Democrat. This tweet is a prototypical difficult example discussed in the related work section. This tweet is generally very positive and talks about a student qualifying for the National Spelling Bee, which was a current event at the time and lacks any sort of policy or language that would by party specific. Thus, it is easy to see why our models had difficulties classifying this tweet.

5 EVALUATION AND RESULTS

5.1 Tree Results

After training the models, none of the tree classification methods performed very well on the test set. ID3 trees had a precision of 0.55, a recall of 0.53, an F1-Score of 0.54, and an accuracy of 0.53. CART trees had a precision of 0.55, a recall of 0.54, an F1-Score of 0.54, and an accuracy of 0.53. Random Forest trees had a precision of 0.53, a recall of 0.76, an F1-Score of 0.62, and an accuracy of 0.53. Because of these weak results, we hypothesized over our feature extraction methods, proposing a few alterations that may have contributed to the low scores. For one, the dot product score of all the GloVe word vectors in the tweet may have reduced dimensionality too much, and it may have been better to simply take the GloVe word vector of the word with the highest TF-IDF score as a feature. This may have based the topic of the tweet around a single word, perhaps strengthening the relationship between topic and sentiment. Additionally, we proposed that a retweet may only be helpful in distinguishing whether or not a user was an avid follower of another user, and may not be a positive predictor or party affilitaion.

	Precision	Recall	F1-Score	Accuracy
ID3	0.55	0.53	0.54	0.53
CART	0.55	0.54	0.54	0.53
Random Forest	0.53	0.76	0.62	0.62

Table 1. Evaluation for Tree Models

	Precision	Recall	F1-Score	Support
Democrat	0.67	0.63	0.65	7230
Republican	0.67	0.71	0.69	7531
Accuracy			0.67	14761
Macro Avg	0.67	0.67	0.67	14761
Micro Avg	0.67	0.67	0.67	14761

Table 2. Evaluation for SVM

Additionally, the tree classification method may not be not very effective in general at classifying textual features, especially relating to political topics.

5.2 Machine Learning Results

Three out of our four Machine Learning models were deemed successful because of their high accuracy scores. The SVM model reported an accuracy score of 0.67 with a standard deviation of 0.38, Naive Bayes had an accuracy score of 0.67 with a standard deviation of 0.36, and MLP had an accuracy score of 0.66 with standard deviation of 0.29. Our worst performing Machine Learning model was KNN, with an accuracy score of 0.56 and a standard deviation of 1.48.

The SVM model also had the same precision for both Democrat and Republican users at 0.67, but their recall varied slightly. SVM's recall for Democrats was 0.63 while their Republican user recall score was 0.71. SVM's F1-Score was also slightly higher for their Republican classification. SVM had an F1-Score of 0.65 for Democrat and a 0.69 score for Republican.

Our Naive Bayes model had a precision score of 0.67 for Democrat and a 0.66 for Republican. The NB recall for Democrat users was 0.63 and 0.70 for Republicans, while the F1-Score for Democrats was 0.65 and the Republican score was 0.68.
MLP's precision, recall, and F1-Score favored the Republican category slightly. The precision score for Democrat was 0.66 while the score was 0.67 for Republican. On the other hand, the recall score for Democrat users was 0.65 and 0.68 for Republican. Finally, the F1-Score for Democrat users was 0.65 and 0.67 for Republican.
KNN's scores varied through both categories. KKN's precision score for Democrats was 0.66 and 0.56 for Republican. The recall score for Democrats was 0.30 and 0.85 for Republicans and the F1-Score for Democrat was 0.41 while the Republican score was 0.68.
Our SVM, Naive Bayes, and MLP machine learning models performed well. However, because the tweets were very similar in structure, it was difficult for our models to have maintain a high accuracy score. Thus, tuning them did not improve their accuracy, and they never exceeded 0.71.

	Precision	Recall	F1-Score	Support
Democrat	0.67	0.63	0.65	7230
Republican	0.66	0.70	0.68	7531
Accuracy			0.67	14761
Macro Avg	0.67	0.67	0.67	14761
Micro Avg	0.67	0.67	0.67	14761

Table 3. Evaluation for NB

	Precision	Recall	F1-Score	Support
Democrat	0.66	0.65	0.65	7230
Republican	0.67	0.68	0.67	7531
Accuracy			0.66	14761
Macro Avg	0.66	0.66	0.66	14761
Micro Avg	0.66	0.66	0.66	14761

Table 4. Evaluation for MLP

	Precision	Recall	F1-Score	Support
Democrat	0.66	0.30	0.41	7230
Republican	0.56	0.85	0.68	7531
Accuracy			0.51	14761
Macro Avg	0.61	0.58	0.54	14761
Micro Avg	0.61	0.58	0.55	14761

Table 5. Evaluation for KNN

5.3 Deep Learning Results

All of our Deep Learning models, Convolutional and LSTM, Convolutional and GloVe, and Convolutional and TF-IDF, had an accuracy score of 0.51.
The Convolutional and LSTM model reported a 0.0 for the Democrat classification for precision, recall, and F1-Score. However, the model had a Republican precision score of 0.51, recall score of 1.00, and a F1-Score of 0.68.
The Convolutional and GloVe model also reported a 0.0 for the Democrat classification for precision, recall, and F1-Score. The model had the same results as the Convolutional and LSTM model. The Convolutional and GloVe model had a Republican precision score of 0.51, recall score of 1.00, and F1-Score of 0.68.
The Convolutional and TF-IDF model also reported a 0.0 for the Democrat classification for precision, recall, and F1-Score. The model had the same results as the Convolutional and LSTM and the Convolutional and TF-IDF model. The Convolutional and TF-IDF model had a Republican precision score of 0.51, recall score of 1.00, and F1-Score of 0.68.

5.4 Correcting Overfitting

Because the scores for our Deep Learning models were so low, our team decided to investigate our results and came to the conclusion that our Deep Learning models were overfitted. This conclusion is seen through our rising validation loss and stagnating, training numbers, and the similar results in each of the models.

	Precision	Recall	F1-Score	Support
Democrat	0.00	0.00	0.00	7230
Republican	0.51	1.00	0.68	7531
Accuracy			0.51	14761
Macro Avg	0.26	0.50	0.34	14761
Micro Avg	0.26	0.51	0.34	14761

Table 6. Convolutional and LSTM

	Precision	Recall	F1-Score	Support
Democrat	0.00	0.00	0.00	7230
Republican	0.51	1.00	0.68	7531
Macro Avg	0.26	0.50	0.34	14761
Micro Avg	0.51	0.51	0.51	14761
Weighted Avg	0.26	0.51	0.34	14761
Samples Avg	0.51	0.51	0.51	14761

Table 7. Convolutional and GloVe

	Precision	Recall	F1-Score	Support
Democrat	0.00	0.00	0.00	7230
Republican	0.51	1.00	0.68	7531
Macro Avg	0.26	0.50	0.34	14761
Micro Avg	0.51	0.51	0.51	14761
Weighted Avg	0.26	0.51	0.34	14761
Samples Avg	0.51	0.51	0.51	14761

Table 8. Convolutional and TF-IDF

Below are examples of our validation loss trends in our Deep Learning models and the reason we decided to combat our overfitting of data. When the validation loss stagnates, we start seeing signs of overfitting.

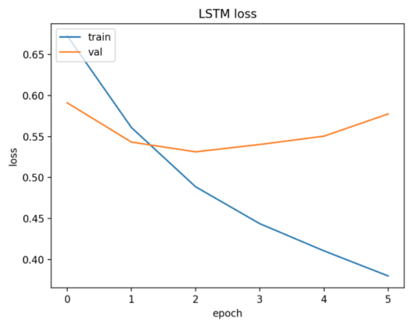


Fig. 6. LSTM Validation Loss Graph

Our DistillBERT training numbers were pretty strange and uncorrelated. Our model reported a loss of 0.17, an accuracy of 0.92, validation loss of 0.69, and a validation accuracy of 0.75.
In our results, overfitting was clearly seen when the model was

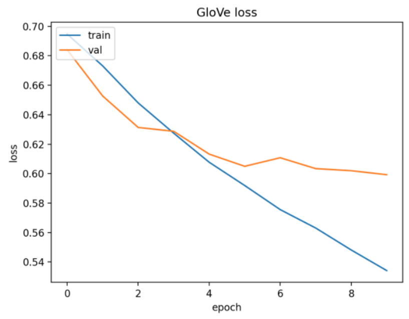


Fig. 7. GloVe Validation Loss Graph

	Loss	Accuracy	Val Loss	Val Accuracy
DistillBERT	0.17	0.92	0.69	0.75

Table 9. Evaluation for DistillBERT

unable to classify in either category. We attempted to combat our overfitting by over and undersampling, stopping early when the validation loss started to rise, and adding dropout layers. Ultimately, we were unable to prevent overfitting in the Deep Learning methods.
At the risk and fear of losing word representation and features in our data, we ended up not using Dimensionality Reduction methods like PCA and Latent Dirchelet Allocation due to our overfitting issues and the fear that our results will miss one category entirely like our Deep Learning models did.

6 CONCLUSION AND NEXT STEPS

Some of the major difficulties in such a classification task arise due to the similarity in the tweet structure and language surrounding political topics. Political officials on both sides end up tweeting about very similar issues and using language that correspond in sentiment because they often positively oriented (at least in these datasets). This makes it hard for a model to differentiate between the tweets to the extent in which it can determine the party orientation of the user.

Another issue that arose was overfitting due to the high dimensionality of text. Pure numerical features are smaller and not as sparse. Attempting to use models that require numerical features resulted in extreme overfitting due to the sheer number of possible feature values.
Following from these issues, there are a few changes we believe we could make if we were to continue our research. If we want to continue with our described task, that is, classifying a user's political affiliation using a single tweet, then we believe that we should alter our data extraction to omit the most popular words from the overall dataset so that there is a clearer division between the two possible classifications and the models that handle text will have an easier time distinguishing between the two groups. Another change that would help is altering the models so that they can understand blocks

of words, this would give more context to the language being used instead of taking each individual word. For example, a Democrat might use a lot of Republican language in a negative way, but our models would currently just see the "Republican" language being used and classify the user as such. Context would do a lot to remedy this issue. Finally, if we wanted to alter the task to classify a user based on their tweets as well as their overall account, we might have an easier time with the larger dataset. Having more text data as well as the account information like race, age, gender, etc. would give us more binary or ordinal features to work with. This would allow us to make more effective models like trees that do not handle text data very well.

Overall, tweets and political talk is more similar than we initially hypothesized. Distinctly classifying a party orientation on a small sample of a tweet can be extremely hard for computers. The difficulty in such a task caused us to question whether such a classification would even be easy for a human, devoid of the context of their twitter feed. Through this project, we learned how much this overall context matters in the transmission of political messaging and that the text content may be less distinct than the polarization that it causes.

REFERENCES

[1] Brownlee, J. Cnn long short-term memory networks.
[2] Devlin, J. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018.
[3] Hochreiter, S. Long short-term memory.
[4] Lab, V. C. S. C. . C. S. Hedonometer word list: labmt-en-v2, 2019.
[5] Nick. U.s. democrat and republican tweet exploration.
[6] Pennington, J. Glove: Global vectors for word representation, 2014.
[7] Sanh, V. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, 2019.