The model comparison on debate tweets and political leanings on the 2016 U.S. Presidential Election

Disheng Liu, Zixuan Zhu, Chen Lei
University of Pittsburgh
dil36@pitt.edu, ziz57@pitt.edu, chl276@pitt.edu

ABSTRACT

The "socialization" attribute of social media emphasizes interaction and relationship building. Interactivity is a major feature of social media, which is different from traditional media. In 2016 US presidential election, the teams of both candidates, especially the Trump side, had formed a large base of supporters by interacting with their supporters on social media through a variety of online community operations. The election debate, as a good opportunity for candidates to speak to their supporters, encouraged people to express their political stance online, which strengthen the community cohesion within both sides while deepen the trench between them. Those users' online community operation, on the one hand, effectively enables themselves to find a sense of belonging and identity, and on the other hand, enables candidates' information to be more widely disseminated through the channels of private traffic. In a word, by studying people's debate tweets we can better depict the plot of support distribution and then better predict the result.

Given that, the problem we are trying to solve is predicting the political group that a person belongs to from his/her tweets on debate. In this project, we firstly cut off those useless data to avoid noise. Then, we use Term frequency–Inverse Document Frequency (TF-IDF) and Principal Component Analysis (PCA) to reduce data dimensions. Finally, we establish 10 different models to figure out which one has the best performance.

1.INTRODUCTION

In the era of traditional media, it is an important part of the campaign activities of American public officials to conduct campaign propaganda through television, newspapers, and other media to let more people know about candidates. Therefore, the investment in campaign advertising accounts for a high proportion of campaign funds for a long time. It is hard for a political newcomer to win in the polls without strong corporate backing. But social media has changed the rules of the game, allowing candidates to communicate directly with voters on everything from macro policies to what's for dinner. From former US President Barack Obama's first attempt to use social media for campaign promotion in the 2008 US presidential election to Donald Trump, who took office in 2016, Posting his own comments on Twitter almost every day, we can see that social media has an important influence on American politics. Therefore, the 2016 US election was a dramatic event worthy of history. In the election, Democratic presidential nominee Hillary Clinton, who was widely expected to win, narrowly lost to Republican presidential nominee Donald Trump. There's no doubt that social media platform Twitter played a big role.

Given that, the problem we are trying to solve is predicting the political group that a person belongs to from his/her tweets on debate. We try to figure out factors that can distinguish people among two groups the most and we believe a good model of dichotomizing twitter users will help us make less mistake in the next election.

Keywords: 2016 Election; twitter; debate; political leaning; model comparison.

2. DATA PRE-PROCESSING

2.1 Text Cleaning

To start our work, we firstly remove some distractive and redundant characts to make our result more neat, significant, and persuasive. We also standardize the font of all text by converting all of those into lower cases. The processes listed below (Set A. & Set B.) are the detailed steps we go through:

Set A:

1. remove retweet symbol ("RT “);

2. remove “&amp", “url”, “http”;

4. remove all “emoji”;

5. remove all tweeter tags;

6. replace “@HillaryClinton” with “hillaryclinton” & “@realDonaldTrump" with “donaldtrump”;

Set B:

1. convert the text into the lower case;

2. remove all of numbers;

3. remove all of punctuation;

4. remove white space;

5. remove stop words (tidytext::stop_words$word) & remove common words ("debate", 'debates',"debatenight",'tonight',' debate2016')

With all data being cleaned as shown above, we are going to demonstrate the overview of the datasets in the next session.

2.2 Data Exploration and Visualization

It is reasonable to assume that specific people have specific language usage habits, and that certain groups prefer to talk about certain topics. In the section of data exploration, we compare the words in the tweets from the supporters of Clinton and Trump respectively.

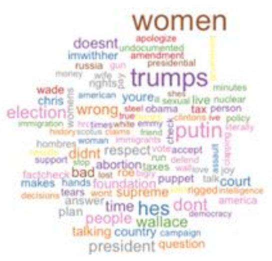

Figure 1. The words cloud of Hillary Clinton’s supporters

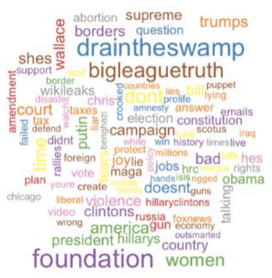

Figure 2. The words cloud of Donald J. Trump’s supporters

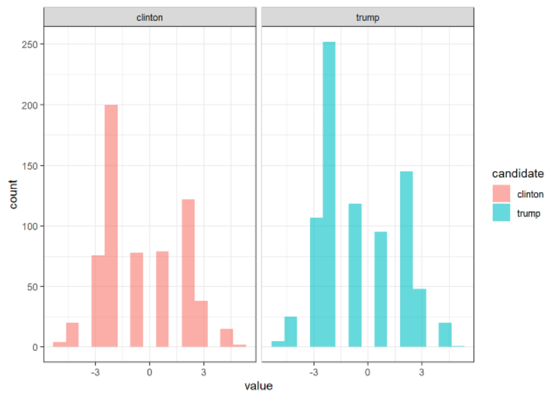

Figure 3. Sentiment analysis with the sentiment dictionary

To have a general overview of the opinions from two candidates’ supporters, after we cleaning the data sets, we create three words cloud to demonstrate the details.

For Hillary Clinton’s supporters as you can see in the Figure 1, her supporters mainly focuse on the domestic policy such as abortion and woman’s rights. Besides that, they also express their concern on foreign threat specifically from Russia as you can see the bigger term of Putin in the cloud. Among Colinton’s supporters Trump himself is also well discussed. Obviously, Trump’s notorious public image and PR stunt strategy of being rude and disrespectul have drawn Clinton's supporters tons of attention.

For Trump's supporters, Trump's anti-establishment and anti-elite attitudes are popular among his supporters as you can see the familiar slogan of "draintheswamp" in the Figure 2. And we also have the suspicion that a lot of his supporters are possibily conspiracy theorists because of the frequently -appeared term "bigleaguetruth".

In the Fig 3. of sentiment, we can see that seemingly Clinton has less negative words in tweets related to her. But the data set we get is too small to be persuasive. Therefore, we check the comparative rates and we find that for Clinton the positive rate is 0.47855 and for Trump is 0.378676503.

3. MODEL ESTABLISHMENT WITH FEATURES ENGINEERING

Feature engineering is the process of using domain knowledge to extract features from raw data. These features can be used to improve the performance of our models. In this section we are going to talk about how we refine our cleaned raw data to make them usable in our models and how we use three different technologies to make the trade-off between the features. After that, we will introduce 6 different methods of modeling discussed in this semester and compare their performance according to different feature engineering strategies.

3.1 Model Establishment Group 1 with Features Engineering of TDM & TF-IDF

3.1.1.1 Term-Document Matrix (TDM)

The usage of word, or say, the distributions of words among Clinton's or Trump's group are different, respectively. Therefore, we can create a Term-Document Matrix (TDM), and each column represent one document, each row represents one term. In TDM, each column demonstrates the word distribution in each document, and we can use word distribution to classify different of people.

3.1.1.2 Term Frequency–Inverse Document Frequency (TF-IDF)

In the corpus, each term (feature) is not equally important. For example, if one term appears frequently in the document or corpus, this term has less discriminatory power and provide less information for a classification model. To get rid of this shortcoming, we use TF-IDF to reweight the TDM, and get a TF-IDF matrix to represent corpus and documents.

3.1.2 Model Establishment

In this sub-section we introduce three models: K-Means; SVM; Classification Tree and compare their performance at the end. Our goal in this session is to find a most accurate yet most cost-effective model to predict which candidate the tweeter user will follow after presidential debates.

First, we decide to use 80/20 training and testing set. To make the data balanced, we used stratified sample with `rsample` package, distributing equally amount of response variables in both training and testing like original data set.

3.1.2.1 K-Means

K-means is the first method we applied to our data. With huge number of documents and variables, a bit of clustering will not ever hurt us. Since we are aiming to separate tweeter users with the candidate they will follow, we picked 2 as our `centers` argument to cluster users into those who follows Clinton and those who follow Trump. The result is not ideal, we end up having 0.5 Accuracy as you can see in the Table. 1 and Table. 2, meaning that by applying k-means we end up having results like randomly clustering users.

Table 1. TP, TN, FP and FN

	Clinton	Trump
Clinton	987	985
Trump	2	2

Table 2. Accuracy, Sensitivity, Specificity and F1 score of K-Means

Accuracy	0.5005061
Sensitivity	2/(2+2) = 0.5
Specificity	987/(987+985) = 0.5005071
F1 score	2(0.5 0.5005071)/(0.5 + 0.5005071) = 0.5002534

3.1.2.2SVM

We have an anticipation that Support Vector Machine will perform well in our data set, because SVM is known by its ability to handle huge information, specifically in handling complex data set of high dimensions. The result is consistent with our thought, SVM performs the best with highest accuracy, sensitivity, specificity combos, resulting in high F1-score (all over 60%) as you can see in the Table 3. and Table 4.

Table 3. TP, TN, FP and FN

	Clinton	Trump
Clinton	136	90
Trump	61	107

Table 4. Accuracy, Sensitivity, Specificity and F1 score of SVM

Accuracy	0.6167513
Sensitivity	107/(107+61) = 0.6369048
Specificity	136/(136+90) = 0.6017699
F1 score	2(0.6369048 0.6017699)/(0.6369048 + 0.6017699) = 0.6188391

3.1.2.3 Classification Tree

The Classification Tree model is also one of our considerations. However, the result on classification tree is interesting, with a lower accuracy (52%) and a sensitivity over 90%. It means that we can correctly predict almost all the positive **r**esults, favoring true positive predictions. Of course, this cost us heavily in true negative predictability, only 12% of data are correctly predicted. With such an extreme prediction, it is not hard to notice that we might have same answer every time we tried to use our model to make prediction. (See Fig.4) Even though the prediction is extreme in classification tree, we can apply loss matrix to make our classification tree move towards higher specificity if that's our goal. (See Fig.5) Consider loss matrix as a prior, without affecting our likelihood function we can make adjustment to our posterior result. As in our practice, a 1:2.5 loss matrix totally flipped our results making 97% specificity prediction while also increased accuracy from 52% to 56% as you can see in the Table 6.

Even though we can play more ratios to make a better model. This is not our main goal in method exploration to our data since it requires even more manual manipulation in modeling.

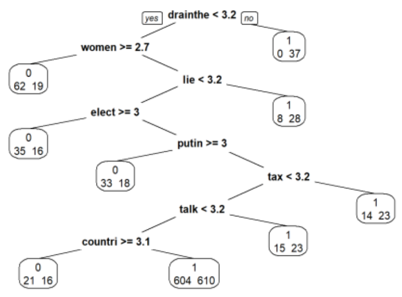

Fig.4 CART (without loss matrix)

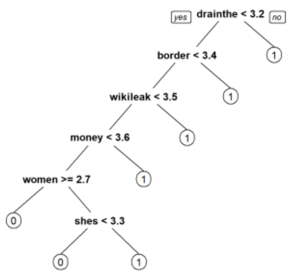

Fig.5 CART (with loss matrix)

Table 6. Accuracy, Sensitivity, Specificity and F1 score of Classification Tree Compared with Loss Matrix and Without

	Without Loss Matrix	**with Loss Matrix**
Accuracy	0.5253807	0.5659898
Sensitivity	0.928934	0.1522843
Specificity	0.1218274	0.9796954
F1 score	0.215405	0.2635952

3.2 MODEL ESTABLISHMENT GROUP 2 WITH FEATURES ENGINEERING OF TDM & TF-IDF & PCA

3.2.1 PCA and Models Involved

In the experiment conducted in the previous section, we find that the TF-IDF matrix is very sparse. Even if we use 2000 tweet texts (50% support Clinton, 50% support Trump), the TF-IDF matrix will yields 3075 terms, or more. We know that if we use more date, like 10k or 100k tweet texts, the number of term will finally be less than the number of document. However, it is impractical, and generate a TF-IDF Matrix is very time-consuming, let alone to data preprocessing and model training. For small data sets, sparse matrices result in more terms (features) than training samples; and for large data sets, it is unrealistic to complete the workflow in an acceptable amount of time.

To address this problem, we use dimension reduction technique, PCA, to compress the TF-IDF matrix, and use the top 100 importance terms as our features to train classification model. In this section we use 6 models including the 3 used in last section and 3 new ones: Logistic Regression, Neural Network, Random Forest.

3.2.2.1 Logistic Regression

In last section of model establishment group 2, one of the candidate methods we intend to use is logistic regression. However, we find this method unapplicable without further processing of our data set. The model does not converge because there are 2937 features while we have only 1976 tweets. Therefore, this model is unable to handle high dimensional problem.

After PCA, we can now apply logistic regression. The result of logistic regression is neither impressive nor bad, it has 55.5% accuracy while achieving the highest specificity as 66% as you can see in Table 7&8. (exclude the result from loss matrix adjustment to classification tree). Here, a big advantage of logistic regression we find is we can easily do analyzation to coefficients with `summary`, `p-value` and coefficient values.

Table 7. TP, TN, FP and FN

	Clinton	**Trump**
Clinton	131	66
Trump	109	88

Table 8. Accuracy, Sensitivity, Specificity and F1 score of Logistic Regression after PCA

Accuracy	0.5558376
Sensitivity	88/(88+109) = 0.4467005
Specificity	131/(131+66) = 0.6649746
F1 score	2(0.4467005 0.6649746)/(0.4467005 + 0.6649746) = 0.5344088

3.2.2.2 K-Means

Of course, we are not satisfied only with a single 60% accuracy. We again applied k means, which produces the worst results we have so far, it fails to find any true positive result.

3.2.2.3 SVM

Thinking about further improving our accuracy, we applied SVM again to our data, while expecting a new higher record of accuracy to occurs, it ends up with an even lower accuracy value, along with a decrease in sensitivity and specificity. We learned the lesson from here that SVM prefers larger data set with more features and dimensionality, the decrease of feature quantity can possibly hurt SVM.

3.2.2.5 Neural Network and Random Forest

So far, we have not yet had any progress in finding better models even after we used PCA to make our data more compact with less noise. However, things changed when we decide to use Random Forest and Neural Network, even though it takes longer time and more memory when training them. The results of both models are satisfying, we have the highest accuracy of 65% so far in the Random Forest as shown in Table 9. Both Random Forest and Neural Network have good sensitivity higher than 65% and excellent F1 score over 0.6. Compared to the previous models, the performance of these two models are outstanding.

Table 9. Accuracy, Sensitivity, and F1 score of Neural Network & Random Forest

	Neural Network	Random Forest
Accuracy	0.609	0.653
Sensitivity	0.665	0.686
F1 score	0.630	0.664

3.2.2.3 Classification Tree

The performance of classification after applying PCA is much more balanced. While the sensitivity being overwhelming high, over 0.9 before applying PCA. Now it has decreased to 0.6 but with a huge increase in specificity from 0.1 to 0.53. The trade-off is good since the F1-score increased from poor 0.2 to 0.57, explaining the new classification tree is a better model compared to the one before PCA.

CART pictures comparison:

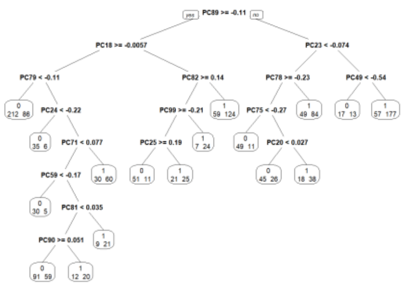

Fig.6 CART (without loss matrix) by current model with PCA

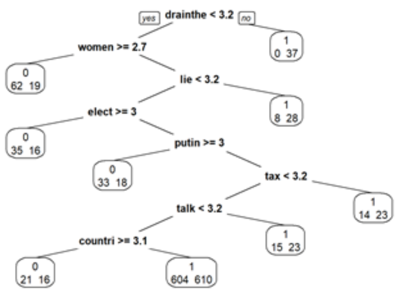

Fig.4 CART (without loss matrix) by previous model in section 3.1.2.3 without PCA

3.3 MODEL ESTABLISHMENT GROUP 3 WITH FEATURES ENGINEERING OF TDM & TF-IDF & PCA & Sentiment Feature

3.3.1 Sentiment Feature and Models Involved

From the result above, we have found that the

TF-IDF weight along, does not work well in several models. Therefore, we introduce a new variable to improve the performance of our models.

As we all know that politicians' supporters always show more positive attitudes toward them and meanwhile more negative attitudes toward their opponents. By measuring the sentiment of text, we can get one more feature indicating the attitude of the publisher towards certain candidate.

Intuitively, if a person tweets a negative comment about someone, that sender is more likely to dislike that person. For example, in the data set, it is easy to identify the person supporting Trump, who tweeted 'debatenight Clinton is a crook'.

Utilize "afinn" sentiment data set, we can rate the sentiment of each document. To be specific, in each text, we look up each word in "afinn" data set and get sentiment score. For each score, we find the closest subject (Trump, or Clinton) and add that score to the subject's sentiment indicator. Therefore, after evaluating each word in the text, we can get the final sentiment score for Trump, or Clinton. For example, if the person dislike Trump, he/she is more likely to tweet negative word in the text, and final sentiment score of Trump in this text is more possible to be a negative value. Inversely, if the person support Trump, the final sentiment score of Trump will be a positive value. In this section we introduce the sentimental features into our 3 models: SVM, Random Forest, Neural Network.

3.3.2.1 Random Forest, Neural Network & SVM

This time we only picked the best models from previous runs which are Random Forest, SVM and Neural Network. Among these models, we find a further decrease of accuracy, sensitivity, and specificity in SVM, though its accuracy goes up slightly. This can possibly be caused by sentiment factor including outliers or extreme values.

As we can see in the Table 10. both Neural Network and Random Forest, have an increase in predictions compared to the performance in previous models listed in Table 9. Especially for random forest, we have reach 70% sensitivity. However, the performance of Neural Network slightly decreases in term of sensitivity and F1 score at the cost of a higher accuracy and we believe this is due to the same reason that causes the performance decrease in SVM.

Table 10. Accuracy and F1 score of Random Forest and SVM

	Neural Network	Random Forest
Accuracy	0.615	0.663
Sensitivity	0.609	0.703
F1 score	0.613	0.677

Table 9. Accuracy, Sensitivity, and F1 score of Neural Network & Random Forest

	Neural Network	Random Forest
Accuracy	0.609	0.653
Sensitivity	0.665	0.686
F1 score	0.630	0.664

4. CONCLUSION AND POSSIBLE IMPROVEMENT IN THE FUTURE

In conclusion as you can see in the Table 11, if we expect to have the highest accuracy, random forest would be the method to choose to predict which candidate a tweeter user follow. If we need to save time/memory or do not have good enough equipment to run random forest, support vector machine is simple and friendly to large data set.

By the way, we considered Logistic Regression to run with sentiment features. However, its accuracy is too low to have any significance. Therefore, we leave this model out.

Table 11. Overall Performance

	Accuracy	**Sensitivity**	**Specificity**	**F2 score**
kmeans(TDM)	0.5	0.5	0.5005	0.5002
kmeans(TDM+PCA)	0.5	0	0.5002	0
Logistic(TDM+PCA)	0.555	0.446	0.664	0.534
Classification tree(TDM)	0.525	0.928	0.121	0.215
Classification tree(TDM+PCA)	0.573	0.609	0.538	0.571
SVM(TDM)	0.616	0.636	0.601	0.618
SVM(TDM+PCA)	0.588	0.622	0.569	0.594
SVM(TDM+PCA+sentiment)	0.576	0.62	0.555	0.586
Random forest(TDM+PCA)	0.653	0.686	0.619	0.664
Random forest(TDM+PCA+sentiment)	0.663	0.703	0.624	0.677
Neural network(TDM+PCA)	0.609	0.655	0.553	0.63
Neural network(TDM+PCA+sentiment)	0.615	0.609	0.622	0.613

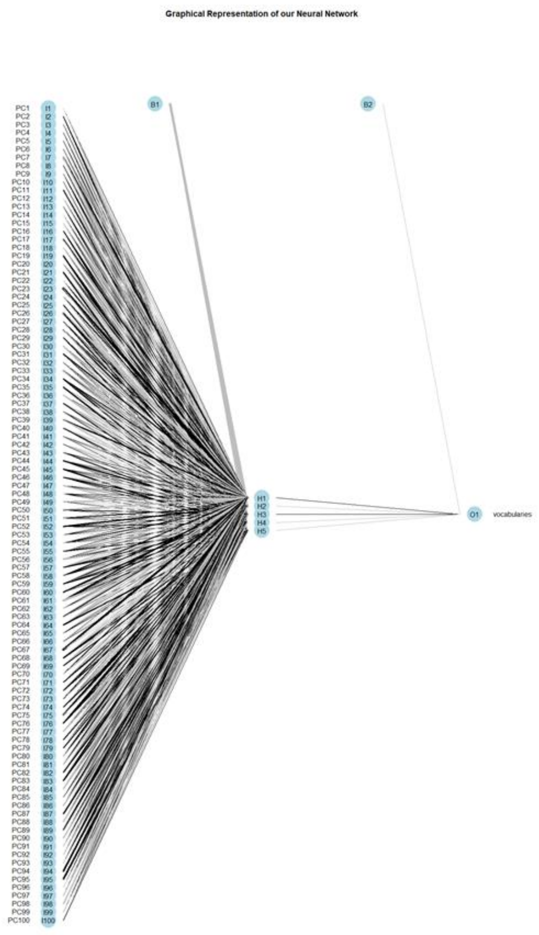

Fig. 5

For the best Neural Network we have, it only contains 5 middle layers as shows in Fig. 5, the run time is not long, and it won't take too much memory.

Besides the Neural Network, It is still interesting for us how decrease of information can deviate performance of SVM method. In the future, we are looking forward to finding a better way for SVM to works better on not only larger dimension but also in lower dimension.

One more important point we also want to mention at the end of the report is that in terms of the sentiment dictionary, we believe there are still much room for improvement in the future. Because during our research we have found some incompleteness in the dictionary.

For example, the size of “afinn” data set is relatively small with only 2,477 records in the data set. Among them some negative words like ‘abort’, ’killer’ and ‘lie’ are not included, which would result that those clearly negative and positive words in the text are not rated, and this would make it impossible to assess the emotion of the text more accurately and comprehensively.

Existing Work We Refer to:

[1] Scott, William. “TF-IDF for Document Ranking from Scratch in Python on Real World Dataset.” Medium, Towards Data Science, 21 May 2019, towardsdatascience.com/tf-idf-for-document-ranking-from-scratch-in-python-on-real-world-dataset-796d339a4089.

[2] Galili, Tal. “K-Means Clustering (from ‘R in Action’): R-Bloggers.” R, R Bloggers, 7 Aug. 2013, www.r-bloggers.com/2013/08/k-means-clustering-from-r-in-action/#:~:text=The%20format%20of%20the%20K,total)%2C%20and%20cluster%20sizes