COS 424 Final Project - Predicting Twitter Engagement During the 2016 US Presidential Election

Peter Chen
Princeton University
`xi@princeton.edu`

Rachana Balasubramanian
Princeton University
`rachanab@princeton.edu`

Ben Burgess
Princeton University
`bburgess@princeton.edu`

Abstract

As more people get their news primarily from Twitter, it is important to analyze how people engage with tweets differently, and how politicians can manipulate their tweets to receive more engagement. During the 2016 presidential election, Donald Trump in particular made use of Twitter as a platform to further his campaign and other political messages, so we chose to use him as the base for our engagement prediction. We trained models on Donald Trump's tweets to predict the number of favorites and retweets that a tweet might get based on information about the tweeter and the tweet itself. We then retrained models using different people's tweets to see if there are correlations between certain types of people on Twitter and the amount of engagement they receive. We compared our models to a bag of words model, and expanded our corpus by replacing URLs with the titles of the respective web pages. Overall, we found that Trump and other Republicans, including his campaign members, were good predictors for the number of favorites and retweets his tweets get, suggesting that people were interested in his political messages, and that attempting to manipulate engagement is not successful without an interesting platform.

1 Introduction

Social media played an significant role in the 2016 United States presidential election and continues to play a significant role in the current presidency. Donald Trump especially utilized Twitter to engage with his supporters and draw citizens into the political discussion. The overwhelming influence of social media makes it important to study, in order to determine what affects it could be having in the political climate and on the opinions of voting citizens. In this project, we will be taking a look at 2016 Republican party nominee Donald Trump (@realDonaldTrump) to see if we can predict how much engagement a particular tweet will get based on information about the tweet and the tweeter.

We use data provided by the Trump Twitter Archive [10]. This archive includes tweets from multiple accounts that we used for our tests with extended data sets. We fit multiple regression models to our tweets corpus, and used the best performing model to find which features were most important to high performance. For our features, we extracted:

- sentiment of the tweet (predicted)
- time posted (scraped)
- follower count of tweeter (scraped)
- number of characters (scraped)
- average frequency of tweets (calculated)
- number of words (scraped)
- average characters per word (calculated)

- intensity (predicted)

The scraped information is extracted from the scraped tweets, as these are pieces of information associated with each tweet. Average frequency of tweets is calculated for the tweeter. Average characters per word is calculated for the tweet we are attempting to predict for (total number of characters/total number of words in the tweet).

For the political alignment sentiment classifier, we created a dataset of scraped tweets that we hand labeled as conservative, non political, or liberal. These tweets are separate from the engagement dataset, and were scraped from various politician's and other political accounts. We trained multiple classification models on this dataset, and found the best one - the best performing model is then used to predict political sentiment on the engagement dataset.

For intensity, we used the vader classifier from nltk, which is "a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media" [17, 15]. It performs better than hand-rated data in assessing the sentiment of tweets. We used the vader classifier to predict sentiment intensity, such as whether a piece of text is positive, negtive, or neutral.

2 Related Work

Common machine learning approaches used for analysis of Twitter data involve the use of linear regression or Support Vector Machine models [14] for prediction or classification purposes. Features are typically extracted from the user (such as followers or volume of tweets), circumstances surrounding the tweet (such as timing) [19], as well as the contents of the tweet itself (such as "quality" of the tweet as measured against a dictionary of word scores) [13]. One particularly important method for predicting engagement involves the use of sentiment analysis, a subfield of natural language processing that analyzes and quantifies opinions, emotions, and/or attitudes of text through computational methods. Sentiment analysis is especially common for social media, where text data can be easily collected to gauge sentiment among users for commercial or, in our case, political purposes, usually involving supervised approaches [16]. More complex models seek to extract more qualitiative information using certain key words, such as the use of noun phrase frequency and their Pointwise Mutual Information (PMI) measure to extract relevant topics of the campaign from tweets [18]. In our study, we construct an ensemble method taking features we found important to predict for specific engagement statistics.

3 Methods

3.1 Data Cleansing

Data for our political alignment sentiment classifier is taken directly from Twitter using a custom scrapper. The data used by the other classifiers is taken from the Trump Twitter Archive [10]. Both of these data sources have the text encoded with full unicode support. This results in numerous emojis, foreign symbols, and special charters in the tweet text of the datasets. To resolve this we first remove special characters then cast the entire text corpus to ASCII, ignoring any characters that can't be translated.

3.2 Linear Regression

To predict for our continuous variables, number of retweets and favorites, we use an ordinary least squares regression, which fits a linear model with coefficients $\hat{\beta}$ to fit a linear relationship of x_i's on y's in the training data. The equation will be in the following form:

$$y = \beta_0 + \beta_1 * x_1 + \beta_2 * x_2 + ... + \beta_m * x_m$$

where we calculate the estimate $\hat{\beta}$ by minimize the residual sum of squares between the observed and predicted responses, represented as:

$$min||(\beta_0 + \beta_1 * x_1 + \beta_2 * x_2 + ... + \beta_m * x_m) - y||^2$$

We can then use the predicted $\hat{\beta}$'s to predict for $\hat{y}$'s using x's from the testing data.

$$\hat{y} = \hat{\beta_0} + \hat{\beta_1} * x_1 + \hat{\beta_2} * x_2 + ... + \hat{\beta_m} * x_m$$

We evaluate the model with both the mean squared error, which measures the average of the square of the difference between the predicted and true response, and the predicted r-squared r^2, which measures correlation between predicted and true response values with higher values (max of 1 which indicates perfect prediction) indicating better predictions. They are calculated as follows:

$$MSE = \frac{1}{N}\sum_i (y_i - \hat{y_i})^2$$

$$r^2 = 1 - \frac{\text{Regression Sum of Squared Error}}{\text{Total Sum of Squared Error}} = 1 - \frac{\sum(y_i - \hat{y_i})}{\sum(y_i - \bar{y_i})}$$

[1]

3.3 Feature Extraction

The regressors in our regression are features extracted from the tweet itself. We first control for **text size**, adding word and character count as regressors, **timing**, adding year, month, and time of day as regressos, and **followers**, adding the number of followers of the user during the tweet as a regressor. We then focus on the tweet text itself, using the follow three approaches:

3.3.1 Political Sentiment Classification

We predict the political sentiment of the tweet testing a variety of classifiers. To get the training corpus, we scraped a collection of 700 tweets from Twitter from multiple politicians and political accounts. We manually labeled each tweet as **liberal, conservative, non-political**, which become our target values. To get our features for regression, we convert the tweet text to a bag of words, and evaluate performance using the following models:

- Gaussian/Multinomial/Bernoulli Naive Bayes - uses the conditional probabilities of the training data being in each label to construct our probability according to Bayes' Theorem (bag-of-words representation is binary in the Bernoulli case, frequency in Multinomial case, and normalized as continuous in Gaussian case). [4, 6, 5]

- Decision Tree Classifier - classifies using a tree model with leaves representing labels and branches reprsenting conjunctions of features leading to said labels. [9]

- Support Vector Classifier - support vector machine fits a hyperplane, or linear separator, between training samples to maximize the margin ebtween classes of samples, which is then used to classify new data points. [8]

- K-Nearest Neighbors Classifier - the intuition behind a nearest neighbor model is to find and subsequently classify using k number of training samples closest in distance to the new point. [7]

- Logistic Regression - similar to OLS, the logistic regression maximizes the likelihood of observing the sample values we use the train the model, as follows:

  $$log\frac{P(y_i = 1|x_i)}{1 - P(y_i = 1|x_i)} = \beta^T x_i,$$

  $$P(y = 1|x_1, ..., x_k) = \frac{exp(\beta_0 + \beta_1 x_1 + ... + \beta_m x_m)}{1 + exp(\beta_0 + \beta_1 x_1 + ... + \beta_m x_m)}$$

  [2]

We found the logistic regression performed the best (Table 5) . We run the logistic regression on our manually-labeled corpus, then use the model to fit a label (liberal, conservative, non-political) in turn used as a regressor in the regression.

3.3.2 Sentiment Intensity

We also use VADER, a parsimonious rule-based model for sentiment analysis designed for social media text, to measure both the polarity and intensity of tweets (polarity classified text as either positive or negative, and intensity assigns valence-based numeric measurement of sentiment). The model begins with a list of lexical features and asociated sentiment intensity measuresments, which are then combined with consideration for five general rules representing grammatical and syntactical conventions for expressing sentiment intensity. This model has been tested to outperform industry standard methods including LIWC, ANEW, the General Inquirer, SentiWordNet, and other techniques involving Naive Bayes, Maximum Entropy, and Support Vector Machines. For our model, we will incorporate positivity score, negativity score, and neutrality score, each ranging from -1 to +1, each of which will be used as a regressor in our regression. [15]

3.3.3 Latent Dirichlet Allocation

From a collection of documents, the LDA infers the per-word topic assignment $z_{d,n}$, the per-document topic proportions θ_d, and the per-corpus topic distributions β_k (Figure 1). The LDA works by assigning words to a few high probability topics in each document and assign high probability to a few terms in each topic.

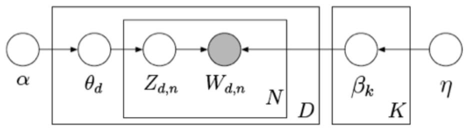

Figure 1: Model of LDA

For each dataset, we quantify each tweet as a mixture of 5 latent categories derived from the dataset (Figure7). We proceed to use these topic proportions as regressors in our overarching regression. [11]

3.4 Cross Validation

We implement K-fold cross in all our prediction models. We set K = 10 as commonly used in practice. To perform this, we partition our data randomly into K equal disjointed subsets, then for for i = 1, 2, ..., K, we let fold i be the test fold to be held out, fit our model on the other K-1 folds, predict on the test fold, and compute generalization error for each sample (i.e. if we were taking a look at R^2 values, we would end up with K=10 separate R^2 values, one for each subset). [3]

3.5 Regularization: Ridge and Lasso Regressions

We perform ridge and lasso regressions by penalizing higher degree polynomials to reduce overfitting, a process called regularization. We test these methods in hopes of improving our predictions.

In a ridge regression, we apply a L2, or least squares error, penalty. As a result, we prefer small but non-zero coefficients in this regression. Our objective can thus be formulated as:

$$min||Y - X\beta||_2^2 + \alpha||\beta||_2^2$$

Our α is a regularization/shrinkage parameter and controls the size of the coefficient (the strength of regularization). As α approaches infinity, our coefficients approach 0, and as α approaches 0, our coefficients approach that of OLS.

In lasso regression, we apply a L1, or least absolute deviation, penalty. Here, we prefer sparsity, which can lead to some zero coefficients. Our objective can thus be formulated as:

$$min||Y - X\beta||_2^2 + \alpha||\beta||_1$$

Our α is a regularization/shrinkage parameter and controls the size of the coefficient (the strength of regularization). As α approaches infinity, our coefficients approach and become 0 (not the case in ridge), and as α approaches 0, our coefficients approach that of OLS. [1]

We select our α in each case by using cross-validation methods noted below.

3.6 Tuning Hyperparameters

Furthmore, we use k-folds to tune hyperparameters, the αs, in Ridge and Lasso regularized regressions. We do this by performing the following:

- Split training data into k folds: $S_1, S_2, ..., S_k$
- For hyperparameter C in $C_1, C_2, ...C_m$
  - For i = 1,2,...,k, fold S_i, fit on the remaining k-1 folds, and predict on S_i
  - Compute generalization error E_c from one prediction
- Pick $C* = argmin_c E_c$

[12]

3.7 Support Vector Regression

As an alternative model to predict for predicting favorites and retweets, we used support vector regression. Support Vector Machines, in a classification case, fits a hyperplane, or a linear separator, between training samples to maximize the margin between classes of samples. Similarly, the regression case maps input x to target y through function $F(x) = w \cdot x - b$, the goal is to estimate parameters w, the weight vector, and b, the bias. For training set D, we solve the followig optimization problem:

$$\begin{aligned} \text{minimize} \quad & L(w,\xi) = \frac{1}{2}||w||^2 + C\Sigma_i(\xi_i^2, \hat{\xi}_i^2), C > 0 \\ \text{subject to} \quad & y_i - w \cdot x_i - b \leq \epsilon + \xi_i, \forall (x_i, y_i) \in D, \\ & w \cdot x_i + b - y_i \leq \epsilon + \hat{\xi}_i, \forall (x_i, y_i) \in D \end{aligned}$$

where slack variables $\xi, \hat{\xi}$ allow for errors to deal with noise in the training data and C is a trade-off parameter. The prediction performance can again be quantified using the MSE and prediction r^2.

[20] [8]

3.8 Extending the Dataset

As a social network, Twitter is heavily affected by the interpersonal connections. We predict that there are groups of people that experience similar engagement. We created three groups: Trumps family (this includes Donald Trump Jr. (@donaldtrumpjr), Ivanka Trump (@ivankatrump), and Eric Trump (@erictrump)), other major figures in the Republican party (this includes Paul Ryan (@speakerryan) and Mitch Mcconnell (@senatemajldr)), and people associated with his presidential campaign (this includes Mike Pence (@govepencein + @vp)). We retrained the models using tweets from people in a particular group, to compare against the results from just using Trump's tweets.

3.9 URL Replacement

Over a quarter of the tweets in the Trump corpus contain URLs. It is very hard to extract sentiment from URLs, especially shortened ones. To attempt to improve the perfomance of our models, we built a header grab script to replace the URLs with the title of their corresponding page.

The header grab script scrapes the tweets in the corpus for any instances of http:// or https:// then grabs all the remaining text in the word which contained those strings. This extracted text is then further processed to remove stray characters at the end of the URL such as periods and quotes which can be present if the URL is placed near the end of a sentence or if it is retweeted. The page content the URL points to is then fetched using the urllib2 requests module with redirects being followed and a total request time limit of 5 seconds. The resulting page content is then processed using the BeatifulSoup library which allows us to easily extract a page's title.

3.10 Bag of Words Model

As a baseline model to evaluate our results sentiment analysis results against, we build a simple bag of words model. We converted the Trump tweet corpus into a bag of words, selected the top 400 features for perfomance reasons, and then tested using an 80/20 split on the regressors we use with our sentintment analysis model.

4 Results

Bolding indicates the best performing model. Each MSE and R^2 result is an average across k-fold cross validation where k=10. Hyperparameters (alphas) in our regularized regressions were selected using cross validation in grid search

The Trump corpus contained 32,452 tweets, the campaign members corpus contained 11,543 tweets, the family members corpus contained 31,038 tweets, and finally the party members corpus contained 13,100 tweets.

	Favorites MSE	Favorites R^2	Retweet MSE	Retweet R^2
OLS	134367134.33	0.0513767	25525455.16	0.0528250
Ridge	**135888256.93**	**0.0548321**	**15477038.56**	**0.0936846**
Lasso	135647897.63	0.0480529	15008773.37	0.0898889
SVR	145792801.58	-0.0816225	19643080.26	-0.0861291

Table 1: Performance using only Donald Trump's tweets, using 80/20 split for training and testing.

	Favorites MSE	Favorites R^2	Retweet MSE	Retweet R^2
OLS	**141211236.83**	**0.0208398**	**21176128.92**	**0.0594757**
Ridge	142533184.82	0.0116734	21195181.65	0.0586295
Lasso	144216689.43	0.0	21196334.73	0.0585783
SVR	154994896.80	-0.0747362	24035370.78	-0.0675157

Table 2: Performance using other family member's tweets for training

	Favorites MSE	Favorites R^2	Retweet MSE	Retweet R^2
OLS	**136506015.51**	**0.0534659**	**21191759.64**	**0.0587815**
Ridge	136642827.16	0.0525172	21206245.25	0.0581381
Lasso	144216689.43	0.0	21212300.39	0.0578692
SVR	154990170.83	-0.0747034	24031598.10	[-0.0673481

Table 3: Performance using other party member's tweets for training

	Favorites MSE	Favorites R^2	Retweet MSE	Retweet R^2
OLS	**137256876.43**	**0.0482594**	**21322704.26**	**0.0529657**
Ridge	137382516.59	0.047388	21346739.68	0.0518982
Lasso	144216689.43	0.0	21350575.54	0.0517278
SVR	154990787.07	-0.0747077	24034315.29	-0.0674688

Table 4: Performance using other campaign member's tweets for training

For the extended datasets, OLS performed the best, but for the 80/20 split of Trump's tweets, Ridge regression performed the best. SVR consistently performed poorly. Generally, retweets are better predicted than favorites.

4.1 Feature Analysis

	Precision	Recall	F1	Time(s)
GaussianNB	0.65	0.62	0.61	0.001
MultinomialNB	0.64	0.62	0.60	0.001
BernoulliNB	0.71	0.70	0.70	0.002
DecisionTree	0.63	0.63	0.63	0.002
SVM	0.72	0.63	0.61	0.031
KNN	0.70	0.67	0.67	0.015
LogisticRegression	**0.72**	**0.69**	**0.68**	**0.002**

Table 5: Political Sentiment Classifier Performance

Logistic Regression has the best precision, and high recall and F1, while being relatively fast. We thus chose to use this model to predict sentiment as a feature for regression on the engagement dataset.

Topic 0	**Topic 1**	**Topic 2**	**Topic 3**	**Topic 4**
obama	interview	trump	china	thanks
true	tonight	president	debt	america
mittromney	foxandfriends	good	congratulations	history
deal	foxnews	run	clinton	economy
people	celebapprentice	love	gop	proud
better	speech	best	debt	military

Table 7: Key words from LDA Categories in Donald Trump's tweets

The topic categories determined by LDA for Donald Trump's tweets appear to be as follows: Topic 0 appears to be campaign-related, topic 1 emphasizes television/media appearances, topic 2 contains many words of praise, topic 3 appear to be more policy-based, and topic 4 appeals to American-exceptionalism.

4.2 Extensions

For our extensions, we used an 80/20 training/testing split on the Trump tweet corpus to provide a comparison to our proposed model.

4.2.1 Replacing URLs

6582 out of 7428 URLs were able to sucessfully be replaced with their corresponding titles. The remaining URLs could not be replaced either because they were corrupted or the sites they pointed to intentionally blocked crawling.

	Favorites MSE	Favorites R^2	Retweet MSE	Retweet R^2
OLS	**112124832.15**	**0.0398159**	**16155917.9**	**0.0382457**
Ridge	120234504.21	0.0289586	17051618.54	0.0395903
Lasso	168741410.92	-5.686473	24315400.68	-6.566333
SVR	146384140.74	-0.0757311	28009360.75	-0.0637483

Table 8: Performance using dataset with URLs replaced

4.2.2 Bag of Words Approach

	Favorites MSE	Favorites R^2	Retweet MSE	Retweet R^2
OLS	2.648224e+29	-1.506941e+21	25165676.15	0.0262356
Ridge	**134035685.06**	**0.0547304**	**27527414.44**	**0.0244745**
Lasso	156179027.93	0.0862696	25022031.97	0.0313541
SVR	138641171.35	-0.0796666	29101540.18	-0.0556807

Table 9: Performance of bag of words model with 400 features

5 Discussion and Conclusion

An initial observation is that retweets are more accurately predicted in the base models than favorites. This could potentially be because people retweet tweets in order to comment on them, but will not favorite them because they do not agree with them. Alternatively, people may not favorite tweets if they already have favorited something similar, because they have no interest in saving the same content multiple times.

In examining our extensions, we found that URL replacement actually performs worse. This may be because Twitter includes links to the tweet itself when retweeting on mobile. This essentially causes our model to double weights the value of tweets that are sent on mobile because the tweet is included once as the tweet text and once as the replaced content from the link. In the future, we would have to ensure that we filter out these links before completing the URL replacement. Additionally, we found that the bag of words approach performed much worse when predicting retweets compared to the model, and performed similarly when predicting favorites. This suggests that the sentiment of a tweet is a much better predictor of whether someone will retweet a post.

Overall, we found that Trump's own tweets were the best training set to predict his other tweets. This makes sense given that there would be the least amount of variation in the features of his own tweets compared to other people's tweets. An example of features that would be significantly more accurate when training on the person's own dataset are the average frequency of tweets and the follower count. The other training set still produced interesting results. Notably, predictions when training with his family members tweet corpus perform poorly, suggesting they do not tweet in a similar manner. This also suggests that the people who engage with Donald Trump's tweets generally do not do so because of his celebrity status, as other people of his "celebrity" family do not receive similar numbers of retweets and favorites. Additionally, it can be seen that training on other campaign members performs better than the family member dataset. Mike Pence's tweet content, especially in the months leading up to the election, reflects much of the same ideas and topics of Donald Trump, just at a lower intensity and frequency. This suggests that the people who interact with these tweets are doing so because they either agree with the ideas being espoused in these tweets or in order to comment on them. The highest performing set included Republican party leaders, which further supports the idea that engagement is based on Donald Trump generally expressing Republican ideals, sentiments, or ideas that conservative Twitter users can interact with.

In summary, this paper provides a detailed analysis of the potential motivating factors that cause varying levels of engagement between different tweets. To do this, we created a custom sentiment classifier based on a handlabelled dataset and combined it with other more common features and sentiment analysis models. We then tested with multiple different training datasets to judge their affect on the predictions we could make for the testing dataset and then we inferred what the results might suggest about a user's behavior. Finally, we noticed a large portion of the Trump tweet corpus contained shortened URLs so we attempted to improve our model's perfomance by replacing the URLs with their corresponding page titles.

Acknowledgments

Prof. Barbara Engelhardt, Dr. Xiaoyan Li, who provided the lectures for COS 424

References

[1] sklearn.linear_model scikit-learn 0.19.1 documentation.

[2] sklearn.linear_model.LogisticRegression scikit-learn 0.19.1 documentation.

[3] sklearn.model_selection.KFolds scikit-learn 0.19.1 documentation.

[4] sklearn.naive_bayes.BernoulliNB scikit-learn 0.19.1 documentation.

[5] sklearn.naive_bayes.GaussianNB scikit-learn 0.19.1 documentation.

[6] sklearn.naive_bayes.MultinomialNB scikit-learn 0.19.1 documentation.

[7] sklearn.neighbors scikit-learn 0.19.1 documentation.

[8] sklearn.svm.svr scikit-learn 0.19.1 documentation.

[9] sklearn.tree.DecisionTreeClassifier scikit-learn 0.19.1 documentation.

[10] Trump twitter archive.

[11] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. 2003.

[12] Jason Brownlee. How to tune algorithm parameters with scikit-learn, 2014.

[13] Sean Gransee, Ryan McAfee, and Alex Wilson. Twitter retweet prediction. 2011.

[14] Ali Hasan, Sana Moin, Ahmad Karim, and Shahaboddin Shamshirband. Machine learning-based sentiment analysis for twitter accounts. 2018.

[15] C.J. Hutto and Eric Gilbert. Vader: A parsimonious rule-based model for sentiment analysis of social media text. 2014.

[16] Efthymios Kouloumpis, Theresa Wilson, and Johanna Moore. Twitter sentiment analysis: The good the bad and the omg! 2011.

[17] Muthuraj Kumaresan. Sentiment analysis using vader. 2016.

[18] Martin Ringsquandl and Duan Petkovi. Analyzing political sentiment on twitter. 2013.

[19] Lei Shi, Neeraj Agarwal, Ankur Agrawal, and Rahul Garg. Predicting us primary elections with twitter. 2011.

[20] Hwanjo Yu and Sungchul Kim. Svm tutorial: Classification, regression, and ranking, 2012.