Predicting users' political support from their Reddit comment history

CS229 Project Final Report
Aaron Acosta (ateam91), Silviana Ciurea-Ilcus (smci), Michal Wegrzynski (michalw)

Abstract

In recent years, the political landscape of the United States has undergone significant changes which are speculated to stem from mass adoption of social media. Most notably, research shows that the overlap between liberal and conservative social networks is surprisingly small. In this project, we test whether the conservative-liberal divide on the Internet is large enough to result in significant differences between the styles of communication of users on either side, even when discussing topics that are not politically charged. To investigate that claim, we selected Reddit[1] users who are highly active on the subreddits dedicated to Donald Trump (*r/The_Donald*), Hillary Clinton (*r/HillaryClinton*) and Bernie Sanders (*r/SandersForPresident*), and labeled them as Trump/Clinton/Sanders supporters accordingly. We then attempted to identify their political inclinations based on the comments they made in non-political subreddits. We repeated the process for their comments in politically-oriented subreddits. Our results indicate that there is indeed a subtle difference in the Reddit usage- and communication-patterns of Trump, Sanders, and Clinton supporters. Our top algorithms were more accurate when applied to political conversations than when used on non-political comments, which highlights how challenging it is to identify users based on their nonpolitical comment histories.

Introduction

Recent research in psychology has shown a correlation between political leanings and various psychological factors, such as negativity bias, disgust sensitivity, pursuit of new information etc. [1] Moreover, recent studies also suggest that liberals and conservatives differ significantly in their attitudes towards and taste in areas unrelated to politics, such as art, or leisurely activities. The language used on Twitter by Democrats and Republicans reflects some of the aforementioned psychological differences, and appears distinct enough to suggest a large degree of polarization between the two different groups. [2]

Previous studies have analyzed the language used by left- and right-leaning individuals on Twitter, under the assumption that most followers of the official Twitter accounts of the Republican and Democratic parties would have conservative and liberal views, respectively. These analyses focused on identifying the most differentiating Porter-stemmed words between the two groups and on identifying language patterns that could be mapped to psychological traits. Good predictors for a user being a Democrat were their use of 1st person singular pronouns, impolite words, positive emotion words, and anxiety words. The use of 1st person plural pronouns and religion-related words were good predictors of a user being a Republican. Unsurprisingly, a high proportion of the differentiating word stems, such as 'obamacar', 'beghazi', 'illeg', appear to be related to political leanings and ideological beliefs. [2]

In this project, we hypothesize that the differences in the language used by Democrats and Republicans on the Internet are stronger than previous studies have found, and that they are strongly present even when users discuss topics unrelated to politics. Moreover, since this election cycle the two Democrat candidates, Bernie Sanders and Hillary Clinton, appeared to attract two significantly different groups of supporters, we decided to investigate the language used by Sanders, Clinton, and Trump supporters.

Dataset and Features

Gathering Raw Data

We collected our data from a dataset containing all comments made on Reddit since the site's inception in 2005. Our main assumption for this project is that the most upvoted comments on the exclusively pro-Donald Trump, Hillary Clinton, and Bernie Sanders subreddits were made by supporters of these specific candidates. From empirical evidence, there appears to be little dissent on each of these subreddits; users who post content that does not conform to the majority's opinion tend to get heavily downvoted (reducing the comments' scores) or even banned from these respective subreddits. Essentially, given a user with a specific comment history, our algorithm predicts to which of the candidate-specific subreddits that user is likely to have posted top-scoring comments. We downloaded 300,000 comments[2] of the users associated with the top 10000 comments on each of *r/The_Donald*, *r/hillaryclinton*, and *r/SandersForPresident*, excluding comments made in these three subreddits, other subreddits dedicated to those candidates and other politically-related subreddits, which we manually added

[1] Reddit.com is a website consisting of forums named 'subreddits', organized around diverse and highly-specific topics.

[2] We selected this number of comments by repeatedly plotting learning curves for training sets of varying sizes and observing that for larger sets, the learning curve leveled off in the classifiers we used.

to the query's filter after going through the comments. Our goal was to remove as much political discourse from our non-political data as possible so we could focus on users' style of communication. We also ran a separate query to download the comments made on political subreddits for the same users, to test the efficacy of our approach for identifying users' political preferences based on their conversations about politics.

Feature Selection

We treated a user's entire non-political comment history as a single example, labelled with the user's political preference. We define a non-political comment as one that does not mention either candidate or party directly and was not made in a political subreddit. We experimented with using a bag-of-words and an n-gram model and decided to use the bag-of-words approach since n-grams provided little to no advantage. We transformed a user's entire aggregated non-political comment history. We opted not to stem the words in our final model since stemming reduced accuracy slightly in both the n-gram and bag-of-words preliminary models and added computational cost. We did not remove stop-words since previous studies have found a strong correlation between stop words and the political leaning of the posters, as explained in the Introduction.

We used the term frequency-inverse document frequency of each item in the model as a feature for the classifier, since it is a standard feature in natural language processing due to reflecting better the importance of each word to each user/user type. In addition, we incrementally added the following features:

1) Sentiment analysis-derived features, such as average total sentiment score per user, average negative score per user etc. and found context-based sentiment per word to be the best sentiment feature. We used VaderSentiment for sentiment analysis, since it is specifically attuned to sentiments expressed in social media. [3] [6]
2) The frequency with which each user posts in each subreddit, called henceforth 'subreddit depth', since preliminary analysis of our data suggested marked differences between the top 20 subreddits in which members of each group posted the most. (as seen in Figure 2 in the poster)
3) Comment score. We found the average score over all of a users' non-political comments to be a very bad feature, reducing accuracy by up to ~5% during preliminary tests. This is likely because of its high variance, apparent unpredictability, and its lack of finer detail in capturing other users' reaction to each particular comment.
4) Controversiality, as measured by the ratio between number of upvotes and downvotes a comment received, which proved to be a valuable feature, most likely because it captures more descriptively other users' reactions to a given comment.

Pipeline

To read the data obtained from BigQuery, a script (i.e. the first item of the pipeline) iterates through each comment and joins all of the comments belonging to a user to form an 'aggregate' comment. For example, if user a posts "star wars is great" and "I like baseball", their "aggregate" comment would be "star wars is great I like baseball". Afterwards, the script vectorizes the text into a sparse matrix of values using a TFIDF vectorizer (provided by the sklearn library) [7]. The matrix is then split randomly using sklearn's *train_test_split* method into training and test sets of size 75% and and 25%, and written to a file in .csv format for the second part of the pipeline to use. The next script, i.e. the second part of the pipeline, reads the data file outputted by the first script. Due to the data being vectorized, the script can read each example and its label with little processing. The training script feeds this data directly into our different classifiers and plots learning curves as well as confusion matrices for each method.

Methods

After experimenting with different models, we chose to use Multinomial Naive Bayes with and without class priors, a linear SVM, Multinomial Logistic Regression, Passive Aggressive, Random Forests, and a Voting Classifier.

We chose Random Forests because of their resistance to overfitting and relative robustness to noise and outliers. [3] Due to the large number of features in our model, overfitting and irrelevant features were primary areas of concern. Tree learning is known to be robust to irrelevant features being included. [4] Random forests work by taking the majority vote for each particular data point of the decision trees of which it consists, avoiding overfitting and maintaining the low bias of tree learners. Thus, random forests do not increase bias, but decrease the variance of the model and reduce the influence of noise on the prediction. [3] We experimented with the number of features to be used in each split, and obtained the highest

[3] VaderSentiment assigns each sentence a negative, neutral, and positive score, corresponding to proportions of the text that fall in each category. For example: "Today only kinda sux! But I'll get by, lol" get assigned the score: {'neg':0.179, 'neu': 0.569, 'pos': 0.251} and "Today SUX!", the score {'neg':0.779, 'neu': 0.221, 'pos': 0} As a context-based sentiment feature in our model, the word 'sux' would get assigned the average score of the sentences it appeared in: {neg: 0.479, neu: 395, pos: 125}.

accuracy for $\sqrt{n}$ to $\frac{n}{4}$ features (the maximum number of features we tried). While using $\frac{n}{4}$features appeared to increase the accuracy by ~1-1.2%, it also increased the running time 7-fold, so we chose to use $\sqrt{n}$ features for splitting. We did not observe a difference when switching the criterion from Gini impurity to entropy, for information gain. Limiting the depth of the trees decreased the accuracy significantly.

Online methods, which process a single instance at a time, appeared to be good candidates for the task at hand because of their ease in processing large amounts of data. The Passive Aggressive Classifier is an online method which adjusts the classification boundary using the training example at any given time. If the example is correctly classified, the algorithm doesn't change the decision boundary(*passive*). If the example is classified incorrectly, the algorithm adjusts the boundary by the minimum amount necessary to classify the example correctly (*aggresive*). Moreover, it is rather robust to noise and PA-I in particular, which we used, makes fewer mistakes on multiclass datasets than other online algorithms such as Perceptron. [5] We chose to experiment with logistic regression because it had shown high potential in previous similar studies on Twitter data. To predict the probability of item i having label n, multinomial logistic regression combines multiple binary classifiers in an "One vs All" scheme. This scheme applied to n classes consists of teaching a binary classifier to discriminate between each class and all the other $n-1$ classes. [4]

We also used a soft-margin Linear SVM, since we expected our data not to be linearly separable. This method uses the *hinge loss function*, which is zero if the data point is on the correct side of the margin, and proportional to the distance from the margin when the data is on the incorrect side. The algorithm works by minimizing $\frac{1}{n}\sum_{i=1}^{n} hinge\ loss(x_i, y_i) + \lambda\|\vec{w}\|^2$, where λ is the regularization parameter that determines the degree of importance given to misclassification. [8] Voting classifiers are recommended for a set of equally well performing models, since they have the ability to balance out the individual weaknesses of each model. They combine different classifiers and predict the class labels by using a majority vote and choosing the mode of the labels. [7] Since Linear SVM, Random Forests, and Multinomial Logistic Regression performed similarly well, we incorporated them in a majority voting classifier, and obtained an increase in accuracy from their individual performance.

We also selected Multinomial Naive Bayes, with and without class priors, due to their suitability for classification with discrete features and their ability to use, in practice, both integer and fractional feature counts. [7]

Experimental Results and analysis

We trained sklearn implementations of the seven different classifiers on the non-political data and predicted their accuracy against a validation set with 5 folds cross-validation. Random guessing would be ~33% for three-class classification. To have a term of comparison for the accuracy of the predictions on non-political data, and ultimately be able to determine how large is the democrat-republican divide on the Internet, we then trained the seven classifiers on the same users' aggregated political comments. [4]

The divide in the non-political parts of Reddit

The subreddit depth increased accuracy the most, for all seven classifiers used, and most notably for Naive Bayes with class priors (~6.5%). This agrees with the results we obtained from analyzing our data, where we saw significant differences between the 20 subreddits that were most popular with each group.

Surprisingly, controversiality was also a very good feature, improving the accuracy up to ~3% (for Voting). Even though controversiality is related to a comments' score (what makes a comment controversial is a small score despite a large number of upvotes and downvotes), the comments' score was an outstandingly poor feature, decreasing accuracy in preliminary testing by up to ~5%.

One of the main challenges we faced was overfitting. Because of the large vocabulary generated by our bag-of-words and n-gram models, we had over ten thousand features before any sentiment analysis or inclusion of additional features, resulting in in extreme overfitting of the training data. On average, the classifiers had a training accuracy of ~90% and a test accuracy of ~51%. To overcome this, we reduced the number of features by specifying minimum and maximum document frequencies for each token in our bag of words/n-gram model. Through trial and error, we settled on removing tokens that appeared in fewer than 25% or more than 75% of the analyzed comments, which significantly decreased our training accuracy and reduced overfitting.

[4] The results we report in this final paper differ from the ones presented in the poster and milestone report. We have recently fixed a bug that we discovered was artificially inflating accuracy values.

	Naive Bayes prior		***Passive Aggressive***		***Naive Bayes***		***Linear SVM***		***Logistic***		***Random Forest***		***Voting***	
	train	*test*	train	*test*	train	*test*	train	*test*	train	*test*	train	*test*	train	*test*
Just verbal analysis	0.442	*0.412*	0.407	*0.362*	0.446	*0.391*	0.474	*0.418*	0.434	*0.393*	0.980	*0.399*	0.583	*0.417*
Sentiment	0.438	*0.428*	0.460	*0.428*	0.446	*0.399*	0.474	*0.437*	0.482	*0.424*	0.990	*0.433*	0.462	*0.424*
Sentiment, depth	0.644	*0.492*	0.675	*0.473*	0.645	*0.499*	0.563	*0.462*	0.599	*0.468*	0.998	*0.452*	0.745	*0.473*
Sentiment, depth, controversiality	0.642	*0.513*	0.663	*0.469*	0.644	*0.508*	0.579	*0.486*	0.575	*0.483*	0.999	*0.483*	0.757	*0.503*

Table 1 Training and test accuracy in the non-political comment set

The divide in the political parts of Reddit

	Naive Bayes prior		***Passive Aggressive***		***Naive Bayes***		***Linear SVM***		***Logistic***		***Random Forest***		***Voting***	
	train	*test*	train	*test*	train	*test*	train	*test*	train	*test*	train	*test*	train	*test*
Just verbal analysis	0.693	*0.648*	0.708	*0.665*	0.660	*0.657*	0.753	*0.702*	0.693	*0.679*	0.998	*0.699*	0.854	*0.715*
Sentiment, Depth, controversiality	0.553	*0.529*	0.449	*0.409*	0.551	*0.533*	0.478	*0.452*	0.479	*0.476*	0.996	*0.605*	0.539	*0.464*

Table 2 Training and test accuracy in the political comment set

The accuracy of all seven classifiers was notably higher on the political comments belonging to the same set of users, when using a simple model such as bag-of-words without any additional features. These results are particularly interesting in the context of a marked decrease in accuracy upon adding two extra features which performed very well for classifying users by their non-political comments. This suggests that the content of the comments is the most important feature in determining a user's political preferences. Further data analysis reveals higher contextual sentiment scores in all three categories (positive, negative, and neutral), suggesting that irrespective of political leanings, users approach political discourse with a similar degree of pathos.

N-grams vs bag-of-words

We initially ran our classifiers using a bag-of-words model. We assumed that this would only be a starting point and generate a good baseline due to the model's simplicity. Our belief that N-grams would prove to be better than a bag-of-words model were substantiated by natural language processing literature, where we found that properly chosen n-gram models can provide better results than unigrams. However, after vectorizing the data into n-grams and attempting to run the classification in that configuration, we found that the n-gram approach gave virtually no accuracy benefit over a bag-of-words vectorization, while greatly increasing the computational cost. Furthermore, the use of bi-, tri- and quadgrams resulted in an explosion of the dimensionality of the problem, making our algorithms prone to overfitting as discussed in the previous section. Therefore

we decided to keep using a simpler bag-of-words model and focus on adding different features instead. The table below shows the observed results for the n-gram approach vs the bag-of-words model.

Features: Sentiment, depth	***Naive Bayes prior***		***Passive Aggressive***		***Naive Bayes***		***Linear SVM***		***Logistic***		***Random Forest***		***Voting***	
	train	*test*	train	*test*	train	*test*	train	*test*	train	*test*	train	*test*	train	*test*
Bag-of-words	0.644	*0.492*	0.676	*0.473*	0.646	*0.501*	0.563	*0.462*	0.599	*0.468*	0.998	*0.452*	0.745	*0.490*
1-3-grams	0.648	*0.515*	0.681	*0.472*	0.650	*0.511*	0.563	*0.447*	0.579	*0.473*	0.999	*0.463*	0.758	*0.447*
2-4-grams	0.646	*0.505*	0.660	*0.479*	0.644	*0.502*	0.564	*0.482*	0.570	*0.458*	0.993	*0.477*	0.744	*0.469*

Table 3 Bag-of-words and N-grams training and test accuracies for a feature set

Limitations/Future work

While the overall performance of the classifiers did not meet our expectations of a "good" classifier (>80% accuracy for 3 labels), we were able to reduce overfitting of the training data while maintaining a score that was significantly better than random guessing (~33% for 3 labels). This suggests that the task of predicting a user's political stance based on the content and properties of their Reddit posts in politically-neutral subreddits is possible yet particularly difficult. The differences in sentence structure, word choice, and sentiment of posts among users with different political leanings is too subtle to make highly accurate predictions. Perhaps Reddit is not suitable for this sort of analysis, and a different social networking platform analysis is needed (e.g. Facebook or Twitter) in which there may be more prominent differences in the speech patterns of users with different political leanings.

One highly promising classification that we could use for the problem at hand to improve accuracy is Confidence-Weighted Linear Classification. [9] This is an online learning method which adds parameter confidence information to linear classifiers, maintaining a probabilistic measure of confidence in each parameter of the model and thus enhancing the online algorithm with information that is essentially a memory of past examples. This method appears to learn faster than other state of the art online and batch methods and improve upon them, as applied to NLP problems. Since this method is not available as an off-the-shelf classifier at the moment, we were unable to test it on our data, but given more time to implement the algorithm, it would be of great interest to us to assess its performance on our data.

Conclusions

We hypothesized that the differences in the language used by Democrats and Republicans on the Internet are stronger than previous studies have found, and that they extend to politically neutral discussions. We confirmed the difference in the language used by supporters of either Sanders, Trump, or Clinton in their political comments, where we obtained ~70% accuracy using a simple bag-of-words model. For non-political contexts, however, we observed that features such as context-based sentiment, controversiality, and 'subreddit depth' were, in aggregate, at least as important as the language contained in each comment. These features increased the accuracy from ~40% (which is ~7% better than random on a three-class problem) by ~10% to an average of 50%. We believe that the largest differences we found between the three groups would be more aptly described as Reddit usage patterns which reflect the wider interests and psychological traits of each group. These results thus support previous studies in psychology that found correlations between psychological traits as manifested through behavior on Twitter and political preference.

References

[1] Hibbing, J. R., Smith, K. B., & Alford, J. R. (2014). Differences in negativity bias underlie variations in political ideology. *Behavioral and Brain Sciences*, *37*(3), 297–307. doi:10.1017/S0140525X13001192

[2] Sylwester K, Purver M (2015) Twitter Language Use Reflects Psychological Differences between Democrats and Republicans. PLoS ONE 10(9): e0137422. doi:10.1371/journal.pone.0137422

[3] Breiman, L. (2001). Random forests. *Machine learning*, *45*(1), 5-32.

[4] Hastie, T., Tibshirani, R., & Friedman, J. H. (2001). The elements of statistical learning: data mining, inference, and prediction: with 200 full-color illustrations. New York: Springer.

[5] Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., & Singer, Y. (2006). Online passive-aggressive algorithms. *Journal of Machine Learning Research*, *7*(Mar), 551-585.

[6] Hutto, C.J. & Gilbert, E.E. (2014). VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text. Eighth International Conference on Weblogs and Social Media (ICWSM-14). Ann Arbor, MI, June 2014.

[7] Pedregosa et al. (2011). Scikit-learn: Machine Learning in Python, *JMLR 12*

[8] Cristianini, N., & Shawe-Taylor, J. (2000). 6.1.2. In An introduction to support vector machines: and other kernel-based learning methods. Cambridge: Cambridge University Press.

[9] Dredze, M., Crammer, K., & Pereira, F. (2008, July). Confidence-weighted linear classification. In *Proceedings of the 25th international conference on Machine learning* (pp. 264-271). ACM.