Political Subreddits: An NLP Driven Analysis of Language Used in Political Discussion

Adam C. Yang[1], Armand Kok[1]
[1]University of California, Berkeley, USA

***Abstract*—Machine learning techniques are used primarily for making predictions. In this paper, instead of solely focusing on making correct predictions, we will be examining the various ways of using machine learning not only to understand how the models make decisions, but also to analyze a large amount of text data. Specifically we will be utilizing a combination of algorithms that has high interpretability so that we can draw understanding out of the text data that we have.**

I. Introduction

Politics has always been an emotive subject with very passionate disagreements between separate parties. The heated political climate since the 2016 US presidential election, and the power to anonymously express opinions on online message boards such as Reddit, has made the language behind political thought much more accessible and very interesting to explore. Using embedding based sentiment analysis and feature analysis of a classification model we will be examining ways of extracting information from the language used in political subreddits.

II. Inspiration and Related Work

The inspiration for our work is based on a paper which explores various applications of sentiment analysis (Pang, Lee, et al., 2008). In particular, the paper covers applications of opinion mining and sentiment analysis in intelligence gathering; how it can be used to better understand subjective judgments of people on various topics. In one of the examples, the authors illustrated how laptop manufacturers can utilize surveys, reviews, and complaints with machine learning to better answer business questions (e.g. why their customers have not been buying the company's products, what have the customers been complaining about). This type of application would essentially save companies hundreds of hours of reading a collection of surveys/reviews/complaints to answer their business questions.

Following this example, we are putting the application into practice and use various machine learning techniques to gain understanding about a body of text. In our case, we are trying to gain an understanding of the differences between the sentiments and language used by two political camps (liberals and conservatives).

III. Data Collection

We collected our data from Reddit, which is organized to various forums, called subreddits, that are dedicated to specific topics. We chose reddit because the anonymity of the platform may encourage people to be more emotive with their language choice about certain topics. Although trolling is an undesired outcome of anonymity, all the subreddits that we chose have anti-trolling rules so we are relying on the admins to remove these posts and comments. We pulled comments and posts data from several political subreddits that are associated with the liberals and conservatives. The timeframe that we included lies between 2015 and 2019. For the liberal camp, we chose r/democrats, r/Liberal, and r/progressive. For the conservative camp, we chose r/Republican and r/Conservative. We chose the aforementioned subreddits since we needed to know the political affiliation associated with the subreddits in order to build the politically biased word embeddings and to train the liberal/conservative classifier model. For this reason, we did not draw data from the massively popular r/politics because it does not label the political affiliation of each post and comment.

The relevant data points that we pullled from reddit are: (1) The post's subreddit, (2) UTC timestamp of when the post was created (3) post title (4) selftext, which contains the body of text associated with the post. (5) The UTC timestamp of when the comment was written (6) comments associated with the post.

IV. EDA

All of the subreddits we used were created in 2008 which was the first election year since Reddit was created. Even though each of the subreddits had the same amount of time to mature, as shown in Figure 1, r/conservative has the same amount of members as the top three liberal subreddits combined. In total, the conservative subreddits have 1.4 times the amount of members compared to the liberal subreddits (34.2% difference). What we found to be very interesting is that the conservative subreddits have 1.9 times the amount of posts (64.1% difference) and up to 5 times the number of comments (133.3% difference) compared to the liberal subreddits. Considering the number of members in each group, the conservative subreddits have a disproportionately larger amount of interactions compared to the liberal subreddits. Perhaps conservatives are more attracted to a forum based medium like Reddit to express their ideals compared to their liberal counterparts. It is also possible that the majority of liberal reddit users chose r/politics which has 5.3 million members as opposed to the smaller liberal-specific subreddits.

According to Figure 2, we see that the word "trump" is the most common word used in both corpuses which is not much

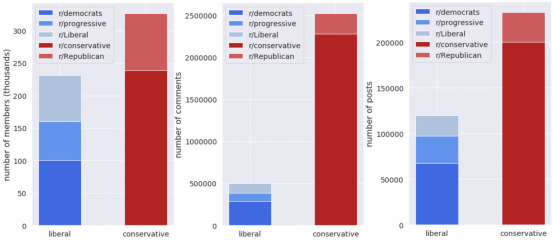

Fig. 1. Bar charts representing the number of members in each subreddit, the number of comments extracted from each subreddit, and the number of posts extracted from each subreddit.

Top Words Used By

	Liberals			Conservatives	
	word	count		word	count
0	trump	862	0	trump	652
1	us	254	1	want	267
2	party	246	2	right	245
3	right	246	3	us	234
4	want	239	4	government	201
5	vote	231	5	conservative	183
6	clinton	222	6	state	161
7	hillary	218	7	left	161
8	democrats	204	8	vote	155
9	republicans	196	9	party	142

Fig. 2. Table showing the list of top words used by liberals and conservatives (out of a random sample of 10,000 sentences and excluding stop words)

of a surprise. What is very interesting, however, is the fact that most of the other words imply some form of "us versus them" context. From this basic word count, we can almost assume that most of the comments and posts in each of our subreddits are about comparing liberals to conservatives, right to left, democrats to republicans, and vice versa.

V. Biased Embedding Sentiment Models

A. Method

When it comes to natural language processing, word embedding is one of the most frequently used and important tools. Word embeddings are essentially vector representations of words with the advantage of preserving some form of context of each word in their respective use cases. Words with similar context would be grouped closer together and the angle between their respective vectors would be close to 0 (cosine of the angle would be close to 1). Despite being such a useful tool, word embeddings are by no means perfect. The context of each word held within their embedding vectors will inherit many of the biases that exist in the corpus that the embedding is trained from (Bolukbasi, Chang, Zou, Saligrama, & Kalai, 2016). For example, in the Word2Vec embedding trained from an extremely large corpus of Google News articles, we can observe the proper relationship that:

$$\vec{man} - \vec{woman} \approx \vec{king} - \vec{queen}$$

. At the same time, we will also see:

$$\vec{man} - \vec{woman} \approx \vec{computerprogrammer} - \vec{homemaker}$$

(Bolukbasi et al., 2016). This second example is a clear case of how the bias from the news articles are preserved within their word embeddings. While a lot of research is going into developing strategies of removing human bias from word embeddings, we aim to take advantage of this flaw in order to gain insight about the sentiment that different political philosophies have regarding certain topics.

To build our word embeddings, we pulled all the titles, posts, and comments from each of the subreddits and grouped them by political affiliation. As mentioned in the EDA section, there are many more comments and posts by conservatives than liberals. Because of this, we randomly sampled comments from the conservative corpus to match the number of comments in the liberal corpus. The same thing was done for posts. We wanted to keep the size of data in each group as similar as possible because our goal is to compare the embeddings created from these two corpuses. This also had the added benefit of significantly cutting down the time it takes to train our conservative embedding model. After this, we preprocessed the text in both corpuses to lowercase everything, tokenize URLs, tokenize different lengths of numbers, as well as stripping out certain symbols and stop words. After that, each of the sentences are tokenized into words and stored into two separate DataFrames, one for each corpus.

We decided to utilize the Skip Gram Word2Vec method developed by Tomos Mikolov in 2013, which is a popular technique that only requires a shallow neural network (Mikolov, Chen, Corrado, & Dean, 2013). According to Mikolov, the Skip Gram Word2Vec method is ideal for a smaller amount of data as well as being able to represent rare words fairly well. Therefore, this method is ideal because our training corpuses are not very large and there are many niche political words that may not occur often. We also incorporated negative sampling for faster training times and learning high quality word representations (Mikolov, Sutskever, Chen, Corrado, & Dean, 2013).

After the embeddings are trained, we take a look at which words are grouped together in the vector space. Figure 3 shows examples of which words are grouped together with "hillary" and "trump" in the vector space. Unsurprisingly, we see that most of these tokens are the different variations of how each group would refer to the two people. What is interesting is that in the liberal word embedding, "hillary" is grouped together with "sanders" while in the conservative embedding "hillary" is grouped with "trump". This suggests that in the liberal subreddits, Hillary Clinton is often discussed in a similar context as Bernie Sanders, while in the conservative subreddits, Hillary Clinton is often discussed in a similar

Words Closest to "hillary"

	Liberal Embedding		Conservative Embedding	
	word	CosineSim	word	CosineSim
0	hilary	0.812887	hilary	0.838069
1	hrc	0.702997	hrc	0.742944
2	sanders	0.698835	trump	0.638722
3	she	0.571965	clinton	0.610870
4	gthillary	0.566821	shillary	0.565243

Words Closest to "trump"

	Liberal Embedding		Conservative Embedding	
	word	CosineSim	word	CosineSim
0	drumpf	0.687529	hillary	0.638722
1	trumps	0.579541	djt	0.617727
2	trumpler	0.574906	he	0.616909
3	gttrump	0.550536	cruz	0.602389
4	he	0.538329	trumps	0.580898

Fig. 3. Lists showing which words are closest to the word "hillary" and "trump" in the liberal and conservative embeddings vector space.

context as Donald Trump. We verified that our embeddings are performing as expected by running numerous examples through them and looking at the closest words.

After we felt confident about the performance of our embedding model, we began to look at various sentiment lexicons to train our model with. We considered Bing Liu's Lexicon which would provide binary labels to a list of positive and negative words (Hu & Liu, 2004). We also looked at the SentiWords lexicon which consists of a large corpus of words, each containing a continuous positive or negative float as an indication of the word's sentiment (Baccianella, Esuli, & Sebastiani, 2010). To train our models, we would map each of the sentiment words to a vector using the liberal and conservative embeddings that we've built. Then we trained these sentiment vectors with various combinations of the sentiment lexicons combined with various types of models. We tried binarizing the SentiWords lexicon in conjunction with classifier models as well as using the continuous labels with regression models. Table I includes some of the combinations we used and their resulting accuracy and RMSE. In the end, the model that resulted in the highest accuracy is a SGDClassifier on Bing Liu's sentiment lexicon while removing stop words when training our embeddings.

B. Sentiment Observations

After we've built our biased embedding based sentiment models, we did a sanity check with a list of Democratic and Republican Presidents. We also included an "Other" category which holds all the presidents that fell into neither party such as George Washington and Thomas Jefferson. The full list

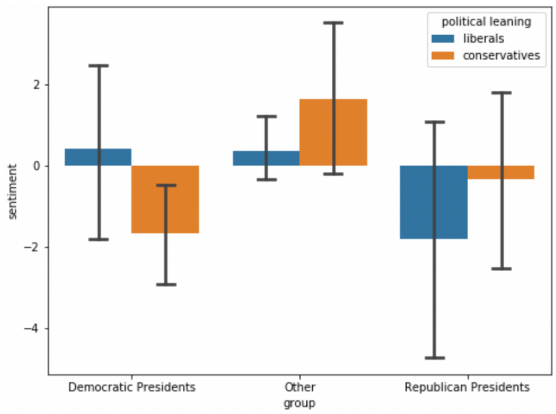

Fig. 4. Bar Graph showing the distribution of sentiments for Democratic presidents, Republican presidents, and presidents that don't fall under either category.

of words we used can be seen in the code associated to this project. Figure 4 shows the results, and as expected, liberals have a higher sentiment for Democratic presidents and conservatives have a higher sentiment for Republican presidents. What was interesting about these results is that even though each group has an obviously low sentiment for the presidents belonging to the opposing party, they don't seem to have an obviously high sentiment for the presidents in their own party. In a way, this suggests that the members of each political philosophy seem to dislike the opposing party more than they like their own party. It is also very possible that most of the posts and comments on these political subreddits are geared towards attacking the opposing party rather than praising their own party.

To further explore this finding, we put together a list of people that each group tends to dislike in order to look at the resulting sentiment. For example, under "Conservative Dislikes" we included names such as "Hillary Clinton", "AOC", and "Bernie Sanders" and under "Liberal Dislikes" we included names such as "Donald Trump", "Ted Cruz", and "Sean Spicer". The full list of words we used in each group can be found in the code associated with this project. Figure 5 shows that conservatives have a very strong negative sentiment towards the "Conservatives Dislike" list and the liberals have a very strong negative sentiment towards the "Liberals Dislike" list. What is very interesting is that the conservatives have a very neutral sentiment towards the people that liberals dislike, and liberals also have a very neutral sentiment towards the people that conservatives dislike. We expected to see at least some positive sentiment, but it seems like dislike for the other group is a lot more detectable than admiration within each political group. This finding suggests that the majority of political discussion regarding politics centers around attacking the opposing group. Furthermore, it is possible to argue that maybe what keeps people in their respective political tribes is not the love for their leaders, but rather the dislike towards the opposing tribe. It is impossible to extract any inference from a study like this, but these findings may influence some extensive studies and field experiments to look into the claim

TABLE I
TABLE SHOWING THE PERFORMANCE OF VARIOUS SENTIMENT MODELS

Model Parameters	Liberal Model Acc	Conservative Model Acc	Liberal RMSE	Conservative RMSE
BingLiu + SGDClassifier + remove_stopwords	***0.803***	***0.802***	N/A	N/A
BingLiu + RandomForest + remove_stopwords	0.747	0.739	N/A	N/A
BingLiu + SGDClassifier + keep_stopwords	0.797	0.798	N/A	N/A
BingLiu + RandomForest + keep_stopwords	0.743	0.736	N/A	N/A
SentiWords(Binarized) + SGDClassifier + remove_stopwords	0.791	0.790	N/A	N/A
SentiWords(Binarized) + SGDClassifier + keep_stopwords	0.791	0.788	N/A	N/A
SentiWords + RandomForest + remove_stopwords	N/A	N/A	0.927	0.934
SentiWords + RandomForest + keep_stopwords	N/A	N/A	0.933	0.931

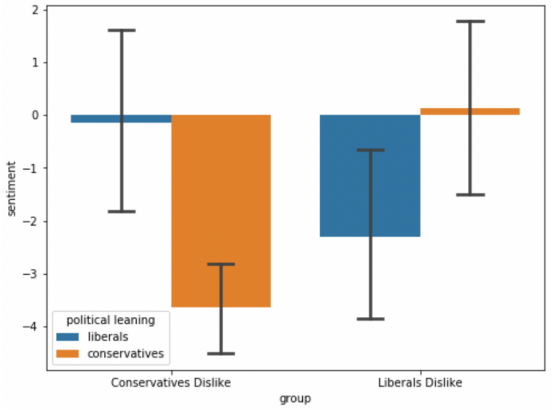

Fig. 5. Bar Graph showing the distribution of sentiments for political figures that liberals dislike and political figures that conservatives dislike

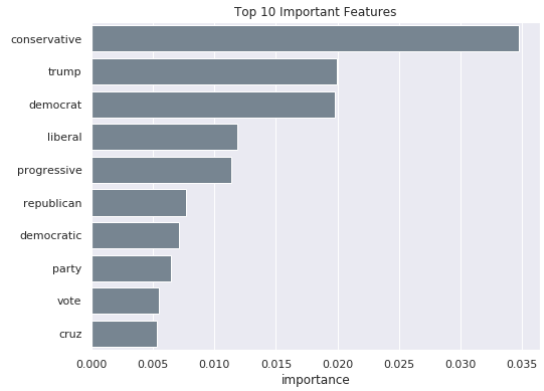

Fig. 6. Chart showing the top 10 most important features.)

of "what keeps us in our political tribes? Mutual agreement or mutual disagreement?".

VI. Political Leaning Classification

A. Method

The project aims to use machine learning in order to understand how the language is used. In particular we are interested in identifying the word features within the post titles that distinguish a conservative post from that of a liberal one. In order to accomplish this we needed to have a classification model and word representation that are interpretable. Thus we chose to use Term Frequency-Inverse Document Frequency (TF-IDF) in conjunction with Random Forest classification to easily draw interpretation about the languages used in the data.

Prior to training the classification model, we applied informed oversampling and undersampling methods to the training data in order to account for class imbalance and improve our model's performance. We applied both informed oversampling and undersampling because previous experiences applying both demonstrated success in improving model performance.

For oversampling we used Synthetic Minority Oversampling Technique (SMOTE) that is grounded on K-Nearest Neighbors. Specifically, SMOTE selects the specified K nearest neighbors based on each document's TF-IDF vectors, randomly selects one of those neighbors, and multiplies it with a random number between 0 and 1 to generate the new data (He & Garcia, 2008).

For undersampling we used the NearMiss method, which is also grounded on K-Nearest Neighbor. In particular, we used a version of NearMiss, called NearMiss-2, that chooses the majority class examples that have the shortest average distance, based on their TF-IDF vectors, to the three furthest minority examples (He & Garcia, 2008).

B. Model Performance

There were three models that were trained and evaluated for the political leaning classification. The first one is just a model that predicts the majority class. The second one is Random Forest classification model without under/oversampling. The third model is another Random Forest classification, but this time it utilizes oversampling and undersampling. Based on Table II both of the Random Forest models clearly performs better than the baseline one. However, the plain Random Forest model seems to not have a good of recall as the one that utilizes oversampling and undersampling. Since we have a minority class, we opted to use the final model for our analysis as it represents the minority class a bit better despite the tradeoff in precision.

C. Model Interpretation

Feature importance in a decision tree is the decrease in node impurity weighted by the probability of reaching that particular node. In Random Forest, the feature importance is averaged across all the decision trees that gets generated out of the algorithm. Feature importance gives us an idea of what are the different variables that are important in making the predictions. In the figure below, we can observe the various word-features that are important for making the class predictions. This gives us an understanding of some of the keywords in our corpus that are significant for differentiating a conservative post versus a liberal one.

Furthermore, we produced partial dependency (PD) plots based on the training data that shows us the marginal effect of

TABLE II
TABLE SHOWING THE MODEL PERFORMANCE OF VARIOUS CLASSIFICATION MODELS

	Conservative			Liberal			Macro-Avg		
Model	Precision	Recall	F1-Score	Precision	Recall	F1-Score	Precision	Recall	F1-Score
Majority Class Baseline	0.66	1.00	0.80	0.00	0.00	0.00	0.33	0.50	0.40
Random Forest	0.80	0.88	0.84	0.70	0.57	0.63	0.75	0.72	0.73
Random Forest with Over/Undersampling	0.82	0.82	0.82	0.64	0.63	0.64	0.73	0.73	0.73

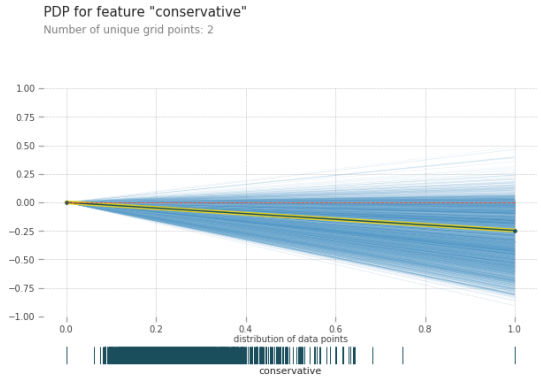

Fig. 7. PD plot for the word ”conservative” along with the ICE lines (blue lines) that represents individual examples and it’s probability changes.

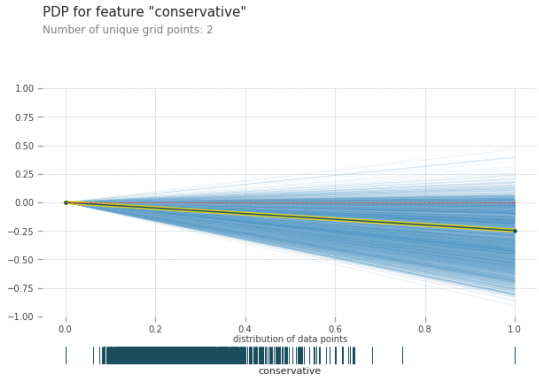

Fig. 8. PD plot for the word ”liberal” along with the ICE lines (blue lines) that represents individual examples and it’s probability changes

the TF-IDF score on the predicted classification. In addition we also overlaid the individual conditional expectation (ICE) lines (one line per instance) on top of the PD plot to show how the prediction probability changes as the TF-IDF score changes for various instances. Lastly, there is also a rug plot at the bottom of every PD plots to prevent over interpretation in areas of the PD plots where there are not a lot of examples.

As observed from the PD plot below, with the word ”conservative” and ”liberal” the average probability of predicting that the post is a liberal one decreases as the TF-IDF score for the word increases. In looking at the ICE lines, there seems to be only a bit of heterogeneous effect, since there are few instances where the probability of predicting a liberal class increases as the TF-IDF score increases. This can be interpreted as the more important the terms ”conservative” and ”liberal” are to the post, the more likely that the post is a conservative one. The ICE lines also implies that the majority of the posts that have the words ”conservative” and ”liberal” tend to be a conservative one. Using this in conjunction with the sentiments Figure 11 we can see what the associated sentiment for the words from both political sides. The word ”liberal” seems to be used in a negative manner in the conservative subreddit, and positively in the liberal subreddit. The word ”conservative” seems to have a positive sentiment with both the liberal and conservative embedding.

In the below PD plot, the word ”democrat” displays little heterogenous effect, very similar to the words ”liberal” and ”conservative.” However in this case the probability of predicting the liberal class gets stronger with a higher TF-IDF score. This PD plot implies that more important the word democrat is to the post, the more likely it is that the post is a democratic one. In addition, the ICE lines for most part goes in the upward direction, implying that most of the posts that contains the word ”democrat” tend to be liberal one. Using the sentiments

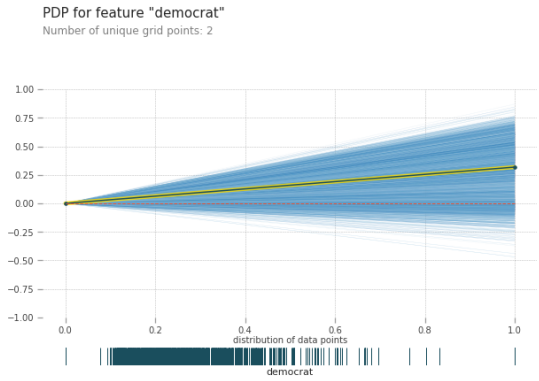

Fig. 9. PD plot for the word ”democrat” along with the ICE lines (blue lines) that represents individual examples and it’s probability changes

Figure 11 we can see that the word ”democrat” seems to have a negative sentiment based on both the conservative and liberal subreddit embeddings.

For the last example, we will look at the word ”trump” as this offers a more complex story. In this case the PD plot is not linear as the PD plot flattens out after reaching a certain TF-IDF score. In addition, the PD plot is also affected by a stronger heterogeneous effect compared to the previous examples as there are a lot of instances where the probability of predicting the liberal class decreases. With this plot it is hard to decipher from the ICE lines which camp tends to have more posts with word trump. Using the sentiments Figure 11 we can see that the word ”trump” have a negative sentiment based on both the conservative and liberal subreddit embeddings, but is more negative with the liberal embeddings.

VII. CONCLUSION

By building custom embeddings for two groups with vastly different point of views, we were able to witness interesting associations between words that are used in similar context

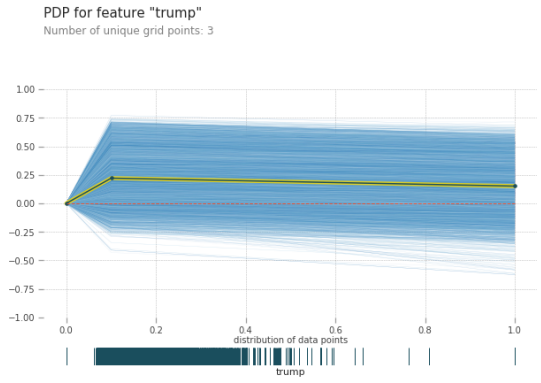

Fig. 10. PD plot for the word "trump" along with the ICE lines (blue lines) that represents individual examples and it's probability changes

	dem_sentiment	rep_sentiment
liberal	3.511913	-4.735556
conservative	3.067661	5.138284
trump	-10.886177	-3.107000
democrat	-2.931628	-10.947041

Fig. 11. The sentiments attached to the top 4 features of the classification model based on liberal and conservative biased embeddings.

based on the cosine similarities. We saw that words like "hillary" and "trump" are associated closely to nicknames and aliases given to the two people by the two oposing groups. On top of this, we were able to see that interestingly enough, "hillary" is commonly used in the same context as "bernie" by liberals and "trump" is commonly used in the same context as "hillary" by conservatives. This may reflect the number of comparisons made between the pairs within each of the groups. From the embedding based sentiment models we built, we were able to explore the sentiment bias that each political group has towards certain words, topics, and people. Most of what we found were not surprising such as liberals having a very negative sentiment of "trump" and conservatives having a very low sentiment of "hrc". What we did find to be interesting is that our results tend to skewed much more towards a low sentiment. Liberals seem to have a very negative sentiment towards people associated to conservatives and conservatives seem to have a very negative sentiment towards people associated to liberals. However, when it comes to the sentiment of liberals towards Democratic presidents and people associated with liberals, the sentiment is not very high. The same thing can be said for conservatives regarding Republican presidents and people associated to conservatives. This finding opens up an interesting question of whether modern political tribalism is based more strongly on dislike of the opposing party rather than an agreement of policies.

Using the political leaning classification model, we were able to infer how the various words in the post title and its corresponding importance (via TF-IDF score) affects the probability of a post belonging to a liberal or conservative subreddit. When combined with the sentiment model, we are also able to observe the type of sentiment associated with those words, which reflects how differently those words are being used by the different political sides. Using the tools above I believe we gave some examples on how to use machine learning algorithms in order to gain some insight to a body of text.

REFERENCES

Baccianella, S., Esuli, A., & Sebastiani, F. (2010). Sentiwordnet 3.0: an enhanced lexical resource for sentiment analysis and opinion mining. In *Lrec* (Vol. 10, pp. 2200–2204).

Bolukbasi, T., Chang, K.-W., Zou, J. Y., Saligrama, V., & Kalai, A. T. (2016). Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *Advances in neural information processing systems* (pp. 4349–4357).

He, H., & Garcia, E. A. (2008). Learning from imbalanced data. *IEEE Transactions on Knowledge & Data Engineering*(9), 1263–1284.

Hu, M., & Liu, B. (2004). Mining opinion features in customer reviews. In *Aaai* (Vol. 4, pp. 755–760).

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems* (pp. 3111–3119).

Pang, B., Lee, L., et al. (2008). Opinion mining and sentiment analysis. *Foundations and Trends® in Information Retrieval*, *2*(1–2), 1–135.