Sentiment analysis: Political hate speech in Twitter

Mark Goldman and Charles (Xueyang) Lin

Dec 14, 2016

Executive Summary

Modelling of politics has become increasingly common since mid-2000s. This has increasingly involved understanding people's emotions in more time-sensitive manner than done by traditional polling metrics. Sentiment analysis can be used to get this information. This project aims to analyze possibilities in this lucrative space. In this analysis, we take 15,000 tweets labeled as hate speech or offensive, process the text with standard tokenization techniques. After dividing the data into validation, training, and test, we analyze it with neural network, naive Bayes and decision trees. We used validation to determine the optimum nodes, variables and tree depth for each case respectively. The misclassification error in the test data, through cross validation, was $25.0 \pm 0.9\%$ for naive Bayes, 23.1 ± 0.6 for decision tree, and $24.2 \pm 0.8\%$. From this, the best model was determined to be decision tree.

To try out the data on politically relevant data, Twitter API was utilized to obtain 27,000 tweets containing comments about Hillary Clinton, Donald Trump, and birthdays (as a control). The best performing model was applied to classify the tweets. Hate speech with Trump in it had the highest frequency, over five times the hate speech rate of the 'birthday' control. Many of these hate messages appeared from critics of the President Elect's cabinet choices. The highest frequency of offensive messages occurred for the 'birthday' category, which may indicate an issue with the model recognizing friendly messages that contain language that may offend some people.

Introduction

Sentiment analysis is the study of people's emotions using machine learning. This technique is useful for many organizations to quickly understand how people are feeling about product or ideas. Knowing a populations feelings can help inform decisions and make an organization successful. The benefits of sentiment analysis has led to huge growth in this field as a major part of natural language processing.

Building and testing a sentiment analysis model involves two main steps. The first is converting the text into a computer-readable format. This is done all throughout natural language processing, and has become very sophisticated. If preprocessing is not done properly, the resulting sentiment model could be infeasible. Common techniques involve converting words to attributes, labeling words with part of speech, grouping words together, and putting parts of speech together to make computer-understandable phrases. This high-dimensional parameter space for machine learning is much bigger than this class has covered since each word or group of words can form its own 'variable'. Many of the steps in preprocessing involve intricate model building as well.

Once preprocessing has been completed, a model can be built. This generally involves the standard data mining techniques that we've used in the class. For this project, we analyzed naive Bayes, decision trees and neural nets. We then took the most accurate model and applied it to twitter posts about political candidates.

Obtaining model data

Since sentiment analysis is structured learning, we needed data that has already been classified to train our model. An interesting data set on Hate Speech by CrowdFlower Inc. contains about 15,000 tweets from Twitter where people have labeled them as 'hate speech', 'offensive' or 'benign'. With this dataset, we can begin to look at some trends in demeaning statements.

Pre-processing Model data

The data did not have any manual explaining the meaning of variables. Most of the columns were sparse, so they were removed. The dataset then had twitter text and the human-discerned offensiveness of the statement.

After figuring out the meaning of columns, we converted the tweets into computer-readable format through a technique called tokenization. This involves converting the tweet into separate variables, one for each word. We chose to remove twitter handles (the parts of a tweet that notify other users '@helen') since they are likely not going to contain much useful information. We also removed the hash tags located in front of words ('#rage') and made all text lowercase to reduce the number of parameters that the model has to deal with.

There is more that we chose not to do. We did not perform trunking, or the removal of tenses on the back of words, since this is known to affect sentiment. We also did not perform stemming, where we label parts of speech and group them, as this would have required another model with labeled parts of speech that was not available.

Model development

We divided the models into training validation and test data sets with cross-validation to compare model performance. We then built three models using different methods, using the validation data to determine key parameters in each. We then compare the models and choose the best one (based on misclassification rate). Each run will give the following confusion matrix. For the choice of model, we are looking at the misclassification rate, as we don't necessarily have a profit matrix. An example of the confusion matrix produced is shown in Table 1 for the final model chosen.

Naive Bayes

The naive Bayes method is excellent for high-dimensional data, since it scales well with number of parameters. We trained our model using this method and received a misclassification rate of 28%. To improve the accuracy of the model, we performed dimension reduction on the data set and saw how the validation data improved. For dimension reduction, the classifiers with the highest absolute parameter values were kept and the model was re-run. Running this process over various numbers of parameters resulted in Figure 1.

Figure 1: Misclassification error over number of features for naive Bayes classifier

Using the validation data revealed that the optimal number of model parameters is approximately 5000 to capture significant information. Using a number of 5000 features, we found the error and uncertainty of the data using cross-validation. The resulting models had an error of $25.0 \pm 0.9\%$.

Decision tree

We then looked at the error associated with decision trees. The max depth of tree nodes was varied with the model and validation misclassification rates recorded, shown in Figure 2

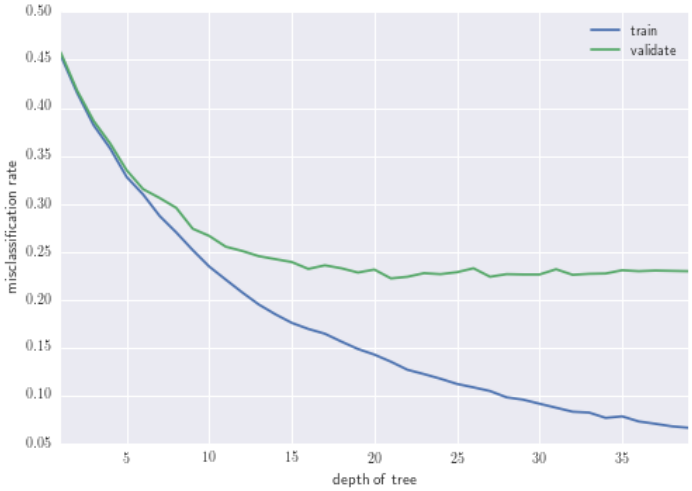

Figure 2: Misclassification error over the depth for decision tree classifier

After twenty levels, the validation data stops improving. Like the naive Bayes model, all ten cross-validation sets were run for the decision tree with depth of 20 nodes. The resulting model errors were 23.1 ± 0.6.

Neural net

A one layered neural net was tested for its ability to predict the types of speech. The neural net was actually divided into 4 sets of data. In this case, part of the training data was kept aside internally by the program to determine which epoch it should stop at. The regular validation data was then used to vary the number of nodes for the first layer.

Figure 3 shows how the training and validation misclassification error changes as the number of nodes are varied. The largest decrease occurs between 1 and 2 nodes, which makes sense because two values are necessary to classify something into three categories. The training does not seem to show a smooth trend, which is likely due to the random start position of the dataset. Despite this issue, the data does seem to improve slightly after two nodes and increases after a few hundred. With that in mind, we took a rough estimate of 60 nodes as a decent value.

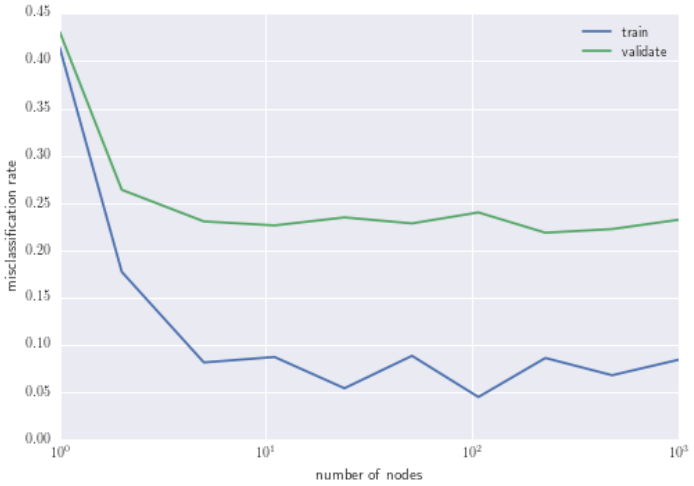

Figure 3: Misclassification error over number of nodes in the hidden layer for neural network classifier

When running cross-validation for the neural network, the models gave an misclassification error of $24.2 \pm 0.8\%$. This honestly did not seem as effective as expected. A second layer was tried without better results, so it was not considered in the study.

Model comparison

These three models behaved quite different. Figure 4 shows how the model's misclassification rates compare. The naive Bayes model has the widest range of accuracy, which is likely due to the model over-predicting significance of rarely occurring words. Weighting the more common words when eliminating variables will likely narrow the range of accuracy among cross validation steps.

Table 1: The confusion matrix for the best performing decision tree model

true value	hate speech	benign	offensive	sum
hate speech	234	55	195	484
benign	3	1443	25	1471
offensive	207	176	597	980
sum	444	1674	817	2935

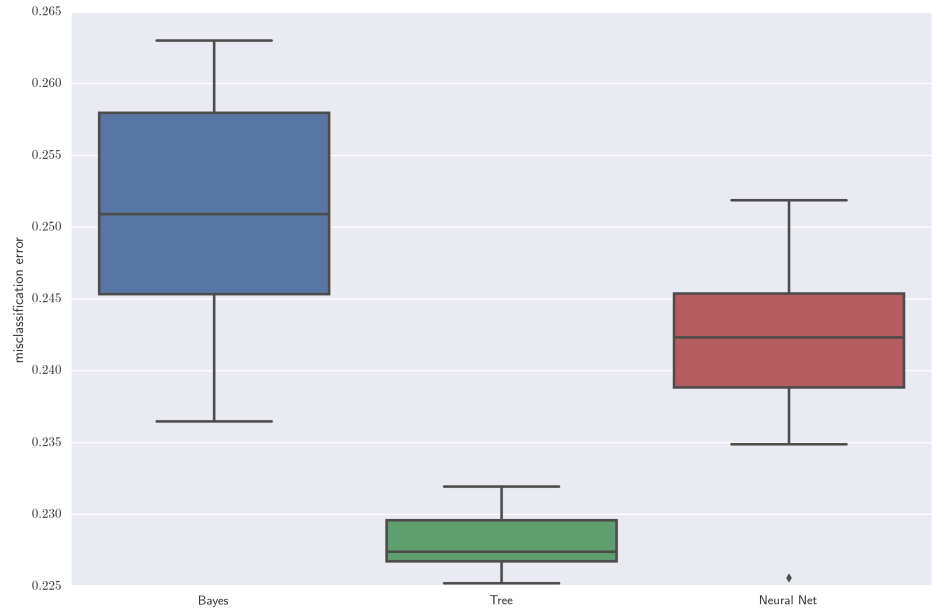

Figure 4: The misclassification errors in the test data for the three models during cross-validation

The neural net had a narrower range than the naive Bayes, though it had an outlier as well. The non-linear behavior of the neural net may curb the improper classification from naive Bayes, while the random seed can impact the reproducibility, which could increase presence of outliers.

The decision tree classifier undeniably performed the best in the scenario. Surprisingly, it had the least variation out of all the models. This may be due to the entropy term weighing more common spreads at the top. Though at the same time it seems counterintuitive since trees are known to be less robust than other methods.

The 'best' model used in subsequent analysis was chosen as the decision tree classifier with the lowest misclassification error on the test case. It's confusion matrix is shown in Table 1. From this table, we can see that real benign responses are rarely misclassified though some hateful or offensive tweets are confused as benign. The model does a worse job at distinguishing between offensive and hate speech, which is also likely harder for humans to decipher as well. Due to the overlap between offensive and hate speech, another useful metric may be combining offensive and hate speech categories, as is done in Figure 7 of the next section.

Table 2: The model's prediction on the set of tweets obtained from the Twitter API

	benign	hate speech	offensive	total
overall	26863	49	295	27207
Hillary Clinton	3077	0	34	3111
Donald Trump	19875	47	197	20119
I'm With Her	366	0	5	371
Make America Great Again	503	1	6	510
birthday (control)	5226	2	74	5302

Applying to political tweets

After testing the model on test data to find out that it had a 22% misclassification rate and that it performed better at differentiating benign from offensive and hate speech, we obtained more current Twitter data, to see what the model would predict.

Data collection and cleaning

We utilized the Twitter API to obtain tweets that relate to Donald Trump, Hillary Clinton, both of their campaign slogans, and the word 'birthday' as a control. Over 150 MB of plain text data was collected from 2:25am to 2:36am GMT on December 9th 2016. It contained over 2,800 tweets and their corresponding metadata.

The data required substantial cleaning and formatting removing data not containing a message in text form. We then classified it as relating to one of five categories Clinton, Trump, 'I'm with her', 'Make america great again', and birthday (as a control). Tweets that included multiple terms were included in both categories.

The classification method involved primarily searching for names. For example, if a message included 'TRUMP', then it would be put in the Trump section even if the writer opposes views of the President Elect. The amount of tweets in each category is shown in Table 2

These tweet texts were tokenized using the same algorithm as in Section .

Results

The decision tree with the lowest misclassification rate was utilized to classify the data. Table 2 shows how the tweets containing each message were classified. Figures 5-7 shows the frequency of each class having hate speech or offensive language.

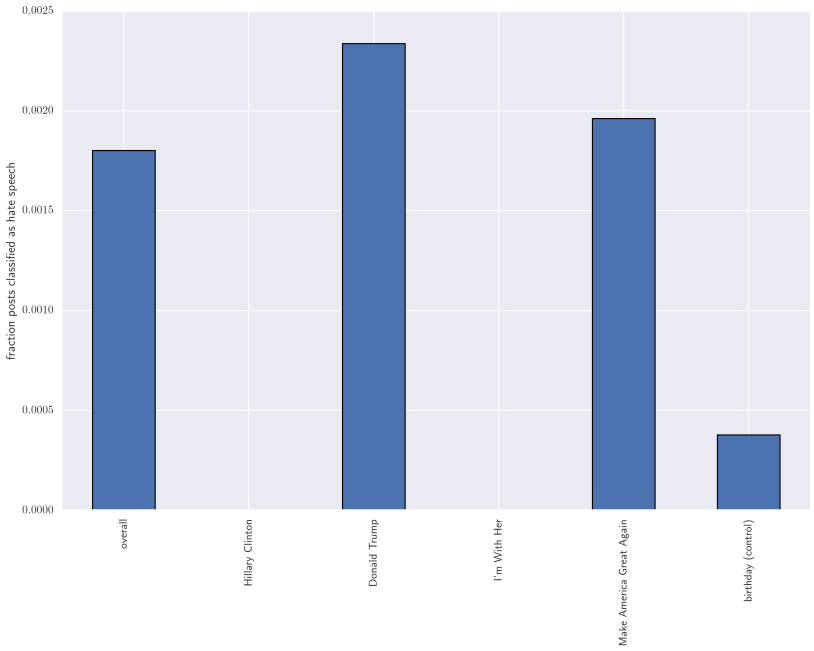

Figure 5: The fraction of posts that are classified as hate speech by the model

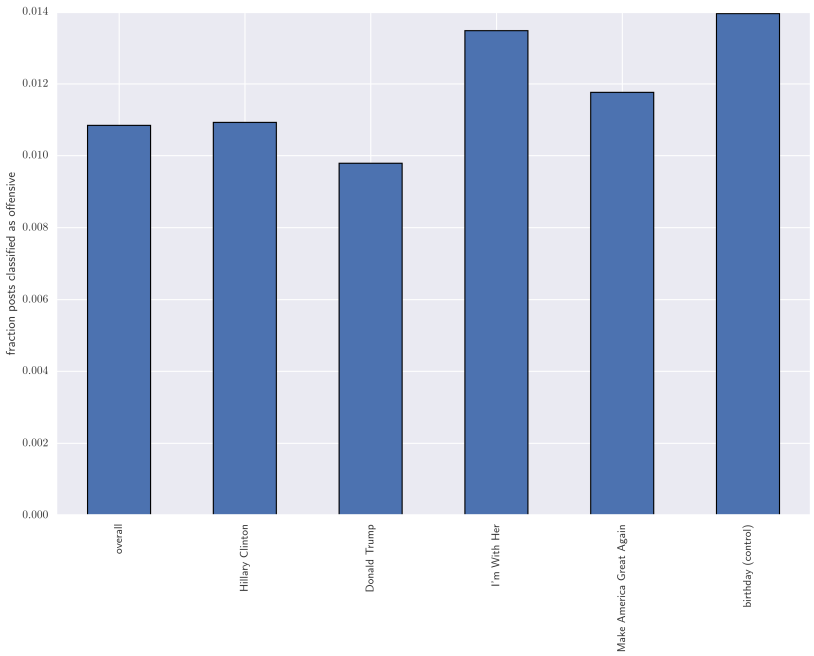

Figure 6: The fraction of posts classified as offensive by the model

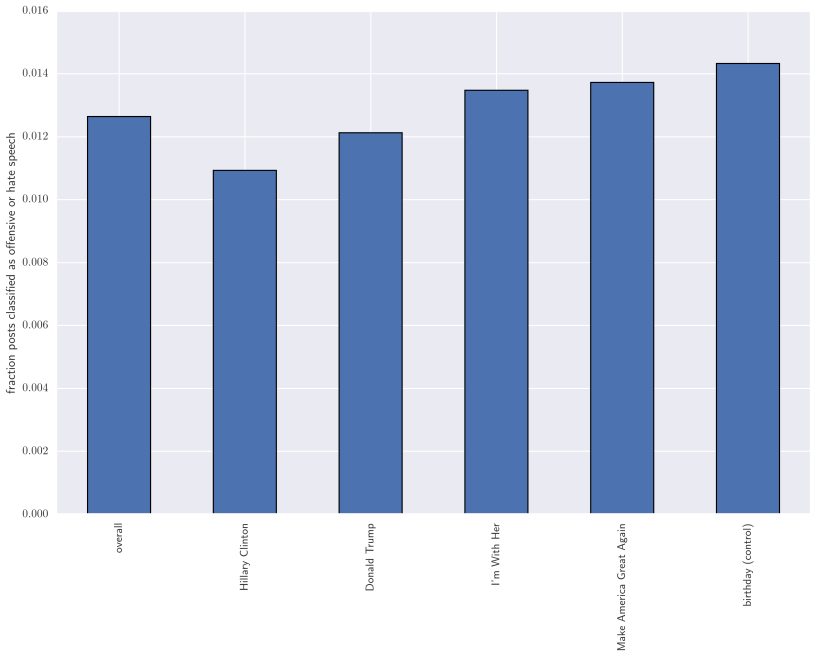

Figure 7: The fraction of posts classified as either offensive or as hate speech by the model

Analysis

Much of the data and analysis corroborates with current trends in politics, which is a major purpose of sentiment analysis. First off, Donald Trump received almost ten times as many tweets as Hillary Clinton, which is likely due to him being more active on twitter and also being the blunt of most recent news reports about cabinet selections. The classification engine predicted almost all hate speech messages to relate in some way with Trump. Since NPR recently covered anger on Twitter related to Trump, our result on some level supports their conclusion.

Ironically enough, the highest proportion of offensive language came from messages about birthdays, which was meant to be a control as something with little negativity, to help pick up background misclassification. The birthday category did do a good job at that for 'hate speech' where it picked up very little. The high rate of offensive language in 'birthday' messages might signify the way friends communicate online by using traditionally demeaning language in a non-demeaning way. This indicates one difficulty with sentiment analysis: the computer has trouble sensing the context of language. The context was likely an issue when classifying birthday messages. The algorithm did a good job at not classifying them as hate speech but did not recognize that birthday messages are very unlikely to try to offend someone.

Adjusted misclassification rate

The labeled dataset had a higher proportion of hate speech than the political dataset collected. The hate speech dataset has about 16% hate speech where our model only identified 0.2% hate speech in the political data set. If the model predicts the correct proportions of classifications, we can estimate the fraction of hate speech which is misclassified. The assumption here is not correct (see Table 1), but this analysis still gives us an idea of our accuracy.

Table 3: Proportion of datasets classified in various ways. The political data values were taken from the model predictions, an assumption mentioned in section

	benign	hate speech	offensive
hate speech test data	0.501193	0.164906	0.333901
political data	0.493678	0.000901	0.005421

By weighting the decision tree test data confusion matrix, Table 1, by the relative proportions in each category, Table 3, we can assess how much we can trust the model.

From the comparison, we have a decrease in the misclassification rate from 22% to 0.1% due to the vast majority of political tweets being benign, which are effectively classified. However, the precision of predicting hate speech properly decreases from 52% to 30% due to much less hate speech existing in the political tweets than in the hate speech data set.

One thing to keep in mind is that the ability for humans to classify hate speech is also based on individual interpretation, so we would not exactly expect our model to be perfect. Regardless, the low precision at finding hate speech should be a warning sign that the model is not very accurate.

References

Links are located in the digital PDF. Actual URLs appear here for those reading the physical version.

- CrowdFlower's publicly accessible databases: https://www.crowdflower.com/data-for-everyone/
- Twitter API: https://dev.twitter.com/docs
- Tutorial followed to extract twitter data: http://adilmoujahid.com/posts/2014/07/twitter-analytics/
- Tokenizer used in the project: http://www.nltk.org/_modules/nltk/tokenize/casual.html
- NPR article about Trump and social media: http://www.npr.org/2016/11/18/502306687/commander-in-tweet-trumps-social-media-use-and-presidential-media-avoidance