Classifying Political Social Media Posts
Machine Learning Project

Krishna Shukla
Simon Fang

23 December 2016

1 Problem

For this project, we chose a dataset that was uploaded by Crowdflower's Data For Everyone Library on Kaggle. The dataset provides 5000 entries of messages from US politicians' social media account, along with human judgments about the purpose, partisanship, and audience of the messages. For example, the second entry in the dataset is a tweet from senator Mitch McConnell from Kentucky and the tweet reads: 'VIDEO - #Obamacare: Full of Higher Costs and Broken Promises: http://t.co/dn3vzqIrWF'. Alongside the text of the tweet, there is a human judgment arguing that this entry is partisan and an attack message. Moreover, the tweet is aimed towards a national audience.

Out of the 5000 entries, we have 2500 Facebook posts and 2500 twitter posts, so the dataset is well balanced in terms of the ratio between Facebook and twitter posts. Moreover, out of the 5000 entries we have 1311 partisan posts, while the remaining 3689 posts are neutral. Here, we have a slightly skewed ratio, but it is not as extreme as for anomaly detection for credit card fraud for example. Despite the relatively small number of partisan posts, there is a pretty high variety of posts within them, so we have enough data to obtain some meaningful results.

The dataset was collected in the following way. Contributors looked at thousands of social messages from US Senators and other American politicians to classify their content. Messages were broken down into audience (national or the tweeter's constituency), bias (neutral/bipartisan or biased/partisan), and finally tagged as the actual substance of the message itself (options ranged from informational, anouncement of a media appearance, an attack on another candidate, etc.). Moreover, each post is judged by a number of humans ranging from 1 to 3. Also, the confidence of the judgment is calculated as well based on the unanimity of the overall judgment. Overall, we obtained a dataset that has been enriched with a lot of contextual data.

In the dataset, the bias of a post is judged by a human. However, we would like to train a Machine Learning algorithm that can predict the bias of a

political social media post. We analyzed some examples of biased and neutral messages of the dataset. We observed that there was a significant difference in length between biased social media posts and neutral social media posts. More specifically, neutral posts were considerably longer in length than biased posts. The difference in length was bigger for Facebook posts than for twitter posts. However, we have to take into account that twitter posts have a limit on the number of characters, whereas Facebook posts are unconstrained. Despite the limit of number of characters on twitter posts, there was a noticeable difference among tweets. Therefore, based on these initial observations, we are going to predict the bias of a political social media post based on the length.

2 Approach and choices

The bias of a post is either partisan or neutral, so we are trying to classify two classes. Throughout the course, we have been taught several classification algorithms and we are advised to make use of the sklearn library, which also provides a flowchart for determining what classification algorithms we can use. We can choose one of the following classification algorithms that are supported by the sklearn library:

- k-Nearest Neighbours
- Logistic Regression
- Support Vector Machine
- Decision Trees
- Neural Networks

Since our features are going to be relatively simple, we would like to use a classification algorithm that is relatively simple as well. Therefore, we would like to start off with a Logistic Regression learning algorithm. Then, we would like to compare our results with a Support Vector Machine learning algorithm.

First, we have to extract the right features that we can use as an input for our learning algorithms. For each social media post, we need to know what the source of the post is, i.e, either Facebook or twitter, and we need to know the number of words of the post. We denote a Facebook post as 1 and we denote a twitter post as 0. In order to determine the number of words per post, we have to extract each post and count the number of words that the social media post contains. Moreover, we need to determine the target value for each post as well, which is in our case the bias of a post, i.e., partisan or neutral. In our project, we denote a partisan post as 1, while we denote neutral posts as 0. We collect all the data in a features vector, where in the first column of the features vector we have the source of the post and in the second column we have the number of words per post. As our target vector, we have the bias of each post.

During our final presentation, we discussed the bias of twitter posts in general. The assumption was that twitter is a good medium for attacks on fellow US political candidates. And since twitter has a restriction on the numberof characters per post, this would affect the bias of our algorithm. In order to test this assumption, we have split the data into a dataset with only twitter posts and a dataset with only facebook posts. We approached this problem by running two separate learning algorithms on the two datasets and then comparing the results. For our features vector as our input, it will only contain the length of a post and for our target vector it will only contain the bias of a post.

In order to train our algorithm, we split our data into a training set and a test set, where the training set is 70% of the data and the test set is the remaining 30% of the dataset. Using this this test set, we can calculate the ROC and AUC in order to evaluate our solutions. Moreover, we can also produce a confusion matrix to see the number of false positives and negatives.

3 Results

First, we performed a Logistic Regression algorithm on the entire dataset, where we had the source and the length of the post as our features vector and the bias of the post as our target vector. After splitting our data into a training and test set, we obtained the following results. The results are shown in Table 1.

	Logistic Regression	Support Vector Machine
Accuracy	0.73	0.73

Table 1: Accuracy of learning algorithms on entire dataset

We observe from Table 1 that the accuracy is identical for both the Logistic Regression and Support Vector Machine learning algorithms. Both have a relatively high value considering that we have a simple model. Moreover, we also calculated the learning curves for both algorithms. These learning curves are plotted in the figures below.

In Figure 1, we have plotted the AUROC curve of the logistic regression algorithm performed on the entire dataset. The AUC score is larger than 0.5, which means that it performs better than just random guessing. However, the score is relatively low and that means that there is a relatively small number of True Positives. Therefore, the Logistic Regression algorithm is performing relatively bad.

In Figure 2, we have plotted the AUROC curve of the support vector machine algorithm performed on the entire dataset. We observe that the AUC score is equal to 0.5, which is a bad score. It means that it performs just as good as random guessing. Even though the accuracy of the algorithm is quite high, the algorithm itself is not performing well due to the relatively low number of True Positives. We can conclude that the support vector machine is performing even worse than the logistic regression algorithm.

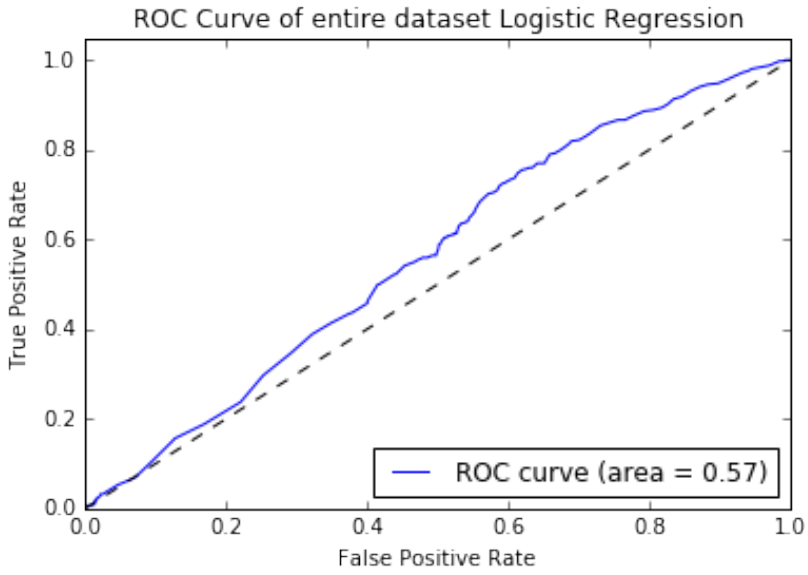

Figure 1: AUROC Curve of the Logistic Regression algorithm on the entire dataset

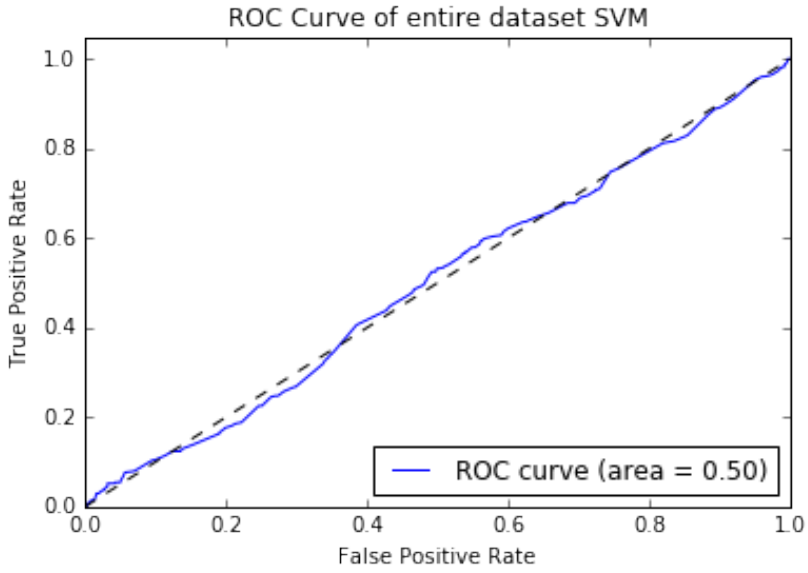

Figure 2: AUROC Curve of the Support Vector Machine algorithm on the entire dataset

Moreover, we can provide more insight into the performance by providing the reader with a confusion matrix. This matrix shows the predicted values against the actual values that were given by the algorithm.

Table 2: Confusion Matrix of Logistic Regression algorithm on entire dataset

	Predicted Positive	Predicted Negative
Actual Positive	1095	1
Actual Negative	402	2

Table 3: Confusion Matrix of Support Vector Machine algorithm on entire dataset

	Predicted Positive	Predicted Negative
Actual Positive	1088	8
Actual Negative	399	5

In Table 2, we have the confusion matrix of the Logistic Regression learning algorithm and in Table 3, we have the confusion matrix of the Support Vector Machine learning algorithm. The two tables are very similar: both have a large number of predicted positive values and a very small number of predicted negative values. The number of true positives is very high, but the number of true negatie is very low. One of the reasons why the algorithm is performing badly is porbably due to the low number of negative values. Moreover, there is a very high number of False Negatives, e.g., 402 for the Logistic Regression algorithm and 399 for the SVM. This influences the ROC and, hence, we had a bad area under the curve. So, from these matrices, we can conclude that the algorithm is performing badly.

We also split the dataset into subsets, where one contains only Facebook posts and the other contains only tweets. Again, we performed a Logistic Regression and Support Vector Machine learning algorithm on the two datasets and we achieve the following results.

In Table 4, we show the accuracy for each learning algorithm applied on each split dataset, i.e., either Facebook or Twitter. We observe that the accuracy

Table 4: Results of split datasets

	Accuracy
Facebook	
Logistic Regression	0.77
Support Vector Machine	0.78
Twitter	
Logistic Regression	0.70
Support Vector Machine	0.70

on the Facebook dataset is performing better than the accuracy on the Twitter dataset. Moreover, we observe that the Facebook subset has a higher accuracy than the overall dataset and that the twitter dataset is performing worse than the overall dataset. This implies that the length of the posts in Facebook have a significant difference and this is in line with our reasoning: Twitter has a restriction on the number of characters. Finally, we can conclude from the table that the algorithm has a pretty high accuracy, even though the features are very simple.

Since the datasets are subsets of the entire dataset, the learning curves are pretty similar and, therefore, not very enlightening. Again, the performance of the algorithm based on the learning curve is bad. This is also due to the low number of negatives. However, from the split datasets, we can conclude that the length of a Facebook post by a US politician can say a lot about the bias of a post.

4 Evaluation of approach and results

Initially, we aimed to predict the bias of a message based on certain words. However, this approach turned out to be out of reach for this project. Instead, we decided to predict the bias based on the length of the posts. Surprisingly, the accuracy of the learning algorithm is quite high, but the number of false positives is high as well, which affects the AUC score. Although the AUC score is relatively bad, we believe that the learning algorithms that were used to classify the problem were appropriate. The two algorithms produced similar results.

In order to improve the results, we can do two things. The first thing is to improve the dataset by balancing out the partisanship of the posts. The current dataset is balanced in terms of number of posts from Facebook and twitter, but the neutral posts outnumber the biased posts. The second improvement we can make is increase the number of features. Perhaps the algorithms are underfitting at the moment and by increasing the number of features, we can prevent the algorithm from underfitting. Lastly, we should try to include the words in order to actually understand the sentiment of the post.