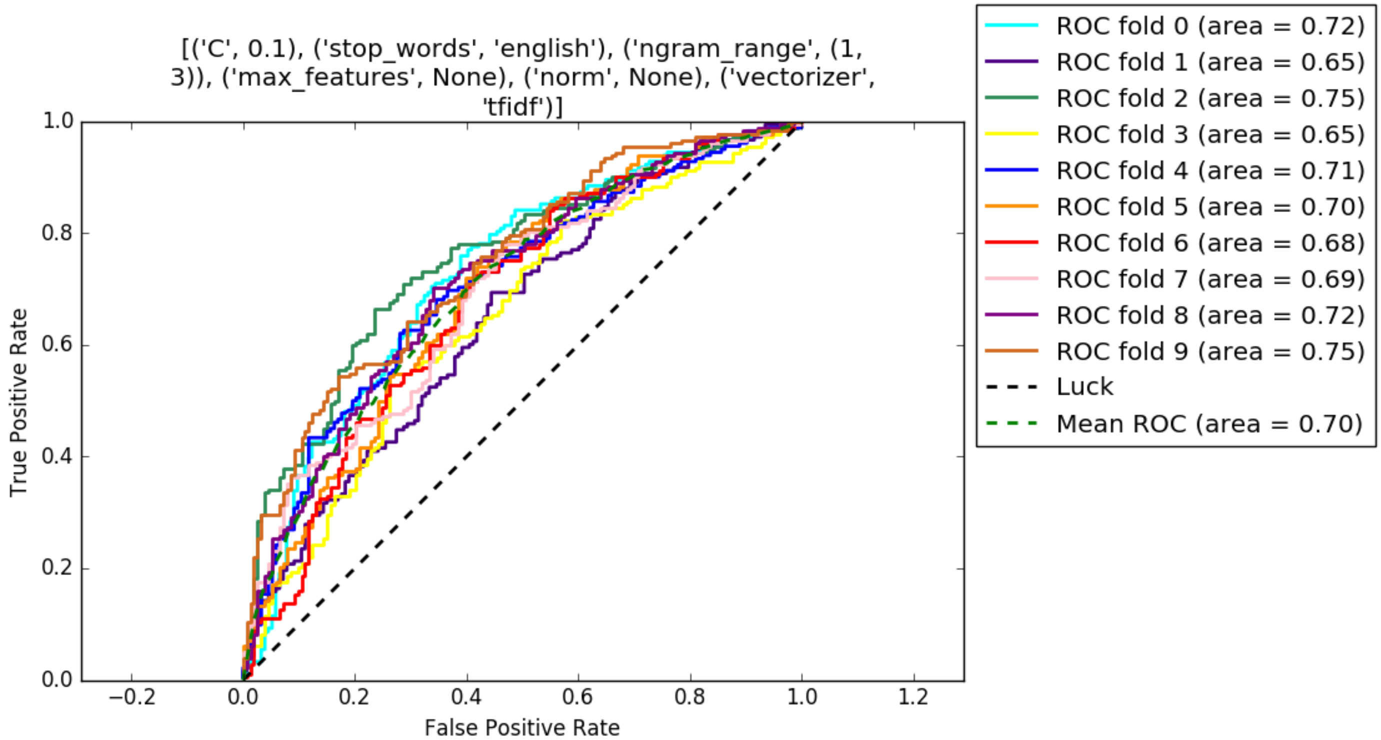
<table>
<tr><td>Members</td><td>Savannah Baron and Varsha Kishore</td></tr>
<tr><td>Title</td><td>Predicting Political Ideology</td></tr>
<tr><td>Hours</td><td>Savannah (8), Varsha (8)</td></tr>
<tr><td>Predicting</td><td>Our goal is to classify the political ideology of sentences. In particular, a binary classification problem to start (conservative/liberal) and a three class problem later (conservative/liberal/neutral).</td></tr>
<tr><td>Data</td><td>We will be using data from the Ideological Books Corpus (IBC). This data set contains sentences from authors with known political standings. All sentences are labeled as either liberal, conservative or neutral. There are 4062 sentences total, with 2025 of them being liberal, 1701 conservative, and 600 neutral.</td></tr>
<tr><td>Features</td><td>Currently we are using bag of words for our feautres, so our feature space is dependent upon the number of words we choose to use in our vocabulary. We've been experimenting with using between 2000 and all (about 11000) words, and have been experimenting with how to form and select these words in a variety of ways. Some examples of things we have tried so far include removal of stop words, n-grams, using stem words, and Tf-Idf (Term Frequency-Inverse Document Frequency).</td></tr>
<tr><td>Models</td><td>So far, we are focusing on a textual classification problem. So, we are using the ever popular SVM and Logistic Regression models to begin with because these models have been used for other similar textual classification problems. Using Logistic regression also gives us a direct probabilistic interpretation and this might be useful in our analysis. An advantage we have with SVM's is that we can use kernels to encode additional relationships between words.</td></tr>
<tr><td>Results</td><td><table>
<tr><th>Model</th><th>Best Params</th><th>Average Accuracy</th></tr>
<tr><td>Random</td><td>N/A</td><td>50%</td></tr>
<tr><td>SVM</td><td>b</td><td>c</td></tr>
<tr><td>LogReg</td><td>Remove English Stop Words, Number of Features $\geq$ 1000, Other features tried make little difference</td><td>About 65%</td></tr>
</table></td></tr>
<tr><td>Problems</td><td>The lone professional paper that uses this dataset achieves a maximum accuracy 69.3% (using recursive neural nets (RNN)). This is not that great, so whether we'll be able to improve from where we are currently, or beyond that far enough to get interesting results is an open question. Additionally, we don't have a lot of data, which may limit the amount that our model will generalize.</td></tr>
<tr><td>Future</td><td>We are planning to look into better feature extraction techniques. This includes looking for negation in sentences, n-grams and specifically politically biased n-grams, and <code>nltk</code> chunking which groups parts of sentences. We also plan to pursue multiclass classification to include neutral examples using random forests. Finally, LDA seems as though it may be interesting in conjunction with our supervised models if we used the results as inputs, however tuning the number of topics in LDA is difficult and subjective, so we plan to time box ourselves to 2 hours initially, and evaluate the results.</td></tr>
</table>

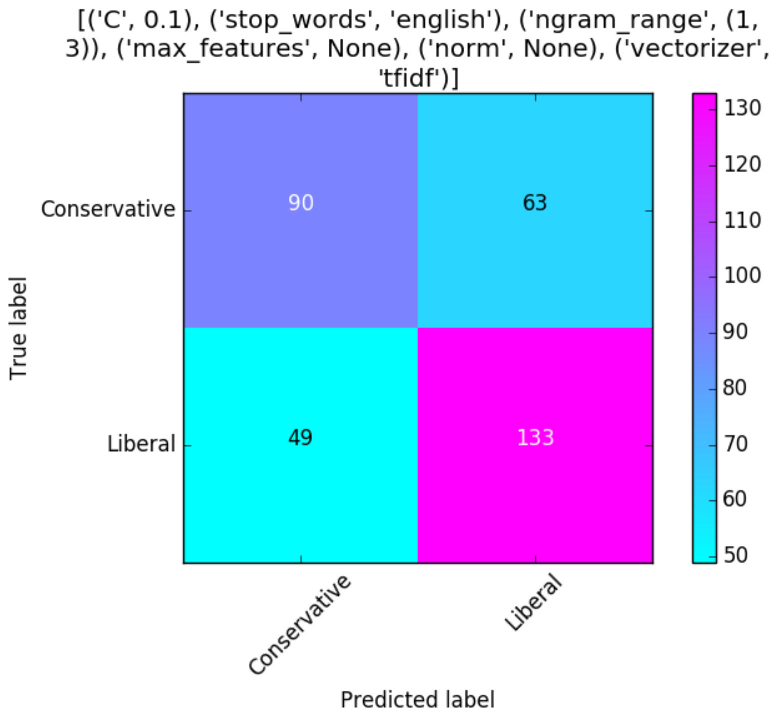
Specific Questions	We feel like we don't have great intuition on how to improve the accuracy of our models, or what sorts of features will perform best. This is perhaps more of a"how to do machine learning" question than a specific one though. Additionally, we wonder whether there are there other classifiers that we haven't talked about that could give us better results with text classification.

Appendix: Example Visualizations:

ROC Plot For Log Reg with parameters as shown in title using 10-Fold cross validation. Area seems about as one might expect given the average performance of 0.65 found for this model.



1 of 10 confusion matrices for the same LogReg model as above. For most of the folds including this one, the confusion matrix shows that misclassifications are relatively balanced between the two classes, which is good given that the proportion of the classes is slightly unbalanced in the data. We are using class weights to offset this problem, which does seem to be working.


[('C', 0.1), ('stop_words', 'english'), ('ngram_range', (1, 3)), ('max_features', None), ('norm', None), ('vectorizer', 'tfidf')]
True label
Conservative
Liberal
90
63
49
133
130
120
110
100
90
80
70
60
50
Conservative
Liberal
Predicted label