MAIS 202 - PROJECT DELIVERABLE 2

Elinor Poole-Dayan

1. Problem statement

Using a dataset of twitter posts from the IRA controlled accounts (which are known to be Russian trolls) between 2012 and 2017 and from various celebrity and scientist accounts, I aim to build a model to learn the categories of different troll-tweets and non-troll tweets. Then, I would run the model on other tweets and measure to what extent a certain post is troll-like, and its similarities to each category.

2. Data Preprocessing

I am still using the dataset I initially proposed from Kaggle that contains 2,973,371 IRA tweets. I will also be using another dataset from Kaggle that contains around 88,625 tweets from various celebrities, political figures, scientists, and organizations. The initial dataset has 15 columns, but I am only using three: content, language, and account category. The reason behind this is because the second dataset does not have all of these features and I want my model to be compatible with both. I am not using author right now because the categories in the IRA dataset are directly related to the author, which would make predicting on the test set very accurate, but then predicting on the new set less certain. In addition, handles are unique and aren't split into words, which would make them less insightful for the algorithm. In the future, these handles could be processed and transformed into something meaningful, but for now they will be disregarded.

For data preprocessing, I am removing the non-English tweets for two reasons. The first being that it makes no sense to have the model consider over 10 languages when my goal is to use the model to predict categories on a dataset containing tweets that are all in English. Secondly, I am considering these tweets in the context of the American twitter scene, and most Americans would not understand the other languages. I also removed all of the tweets in the Unknown category because there is no obvious trend in these tweets and I don't want the classifier to default to categorizing tweets as Unknown. After getting rid of the non-English tweets, I dropped the language column. The total number of remaining troll tweets is 2,121,741. Combined with the non-troll tweets, the total is 2,210,365.

I created my own text cleaning function because I wanted to customize it to preserve the important aspects of tweets and drop the less important parts. To do this, removed punctuation not pertaining to mentions ('@') or hashtags ('#') in order to make it easier to vectorize. While punctuation could be somewhat meaningful in the context of short tweets, I believe that removing most of it will help in terms of reducing complexity and standardizing the use of punctuation across different accounts. I made everything lowercase and stemmed the regular words to reduce the size of the vocabulary. Even though it is possible that troll tweets with hate speech or insults use all caps more often than commercial or non-troll tweets, I think the benefit of standardizing capitalization will make the model better. I am also removing stop words, which consist of short words that have no meaning out of context and won't affect the category, such as 'I,' 'me,' 'my,' 'myself,' 'we,' 'our,' 'ourselves,' 'you' etc. I based my stop words on the stop

words from NLTK, with few modifications to better fit the context. Lastly, I only included words mentioned at least 5 times in the vocabulary to reduce the size (and sparsity) and speed it up. Words used fewer than 5 times throughout the training data are almost guaranteed to have no effect on the category and are thus safe to ignore.

3. Machine learning model

I chose to use the Stochastic Gradient Descent model because it works well with text classification and larger datasets. Due to the large vocabulary that comes with having many data points, each vector becomes pretty sparse. Thus, SGD is more efficient in addition to being compatible with sparse matrices.

I split my data into training and testing using scikit learn train_test_split. I used 75% of my data for training and 25% of my data for testing. Since I have 2.2 million data points, I wanted my training data to be the majority so that the vocabulary becomes large enough to contain the most common words and therefore be able to classify the test tweets more accurately.

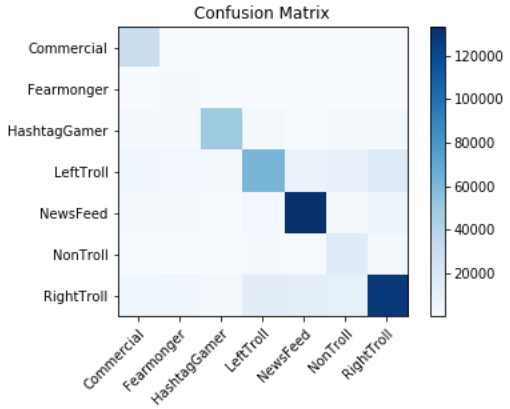
I left all of the hyperparameters as default except for the class weight, which I set as balanced. This means that the model uses the values of y to automatically adjust weights inversely proportional to class frequencies in the input data. The reason I used this is because there are categories with significantly more data points in certain categories than others, so I wanted to weight the smaller classes a bit more (proportional to their size) to make sure the classifier gets them right as well.

In order to test the accuracy of the model, I used the score method on the classifier. This returns the mean accuracy of how many of the predicted labels matched the actual labels. I don't think my model is severely over or underfitting because the test and train mean accuracies are pretty close.

In terms of challenges, I had a lot of difficulty in getting the vectorization to work because I defined the cleaning/preprocessing manually. Thus, it was hard to match up data types and make sure exceptions got handled correctly because the preprocessing methods from scikit learn were made to match up with the vectorization method, but I had to figure this out on my own. In addition, my cleaning methods took a long time to run and this made testing and debugging a lot slower. I solved this by removing NaNs in the beginning and trying things line by line until it ran smoothly.

4. Preliminary results

The mean accuracy of my model is 76%, which is not the best. I believe that this could be improved upon. However, it is possible that some of the troll categories are hard to distinguish. For example, it might be hard to tell apart a real Donald Trump tweet from a Russian troll because the word "Trump" or mentions to him "@realdonaldtrump" occur frequently in both. Also, the ambiguity of the Fearmonger category could be the reason the model found it hard to differentiate between it and the other categories, as evidenced by the relatively low numbers (light green) for that category. Additionally, the model frequently confused left and right trolls, which could possibly be an effect of the political nature of both. Moreover, the fact that I used a bag of words representation of the tweets as opposed to a bigram might lead to an increase in misclassifications.



I ran the model on some test tweets that I made up in order to see which words affect the classification the most. "#maga" and "#blacklivesmatter" were two words that largely affected the classification of the tweet. For example, the tweet "my dog is so cute #pets" was classified as a Fearmonger tweet with 16.56% certainty. When "#maga" was added to the end, it changed to a RightTroll tweet with 59.39% certainty and when "#blacklivesmatter" was added, it changed to a LeftTroll tweet with 65.26% certainty.

5. Next steps

The next steps are to fine tune the hyperparameters and see if I can increase the mean accuracy of the model. A pro of this method is that I got it to work with my own text cleaning methods, and this would possibly be an obstacle to using other models if the vectorization needs to be changed. I might want to try changing it from a bag of words to an n-gram, such as a bigram and see if this preserves more context and therefore increases the accuracy. I would also possibly want to try other models if the accuracy doesn't increase enough, such as SVM or even a neural network.