Erik Green

Project 3 – Executive summary

The purpose of project 3 was to become more comfortable in scraping text from the web, and to learn how to process this text (Natural Language Processing, NLP) to build predictive models.

After scraping the posts of several subreddits, I decided to scrape the comments of the Democrat and Republican subs. I opted to utilize the info scraped from comments instead of submissions because I found them to be more text-heavy. The actual submissions were often links, videos, pictures, and memes. Comments were better for training with NLP.

Comment totals were around 350,000 for Democrats and 315,000 for Republicans. After deleting empty cells, classes were nearly perfectly balanced at 300,000. The DataFrames were merged randomly. Words from the comment ‘body’(ies) were cleaned; all symbols and punctuation were removed, words were converted to lowercase, stop words were removed, and words were lemmatized.

Documents were separated in three ways; CountVectorizer without n-grams, CountVectorizer with n-grams, and term frequency-inverse document frequency (TF-IDF). Due to the size of the dataset, the python kernel died several times throughout this and other processes. CountVectorizer without n-grams was used in predictive modeling.

A train-test-split was done at a ratio of 5% train 95% test. This ratio was chosen so that several models could be run on the training set. A 5% share of 600,000 rows still leaves a significant training set of 30,000. This ratio also serves as a good representation for ‘real-world’ situations, where most of the data is unavailable. Despite the ratio, train-test-split proved to be one of the more difficult tasks for the computer to handle.

4 types of predictive model were run on the training set. The initial plan was to run a gridsearch on these models and choose the best three models with best parameters and run them through a VotingClassifier. This turned out to be unnecessary however, as the RandomForestClassifier performed head and shoulders above the rest. Putting the 2 runner ups into a voting classifier would only hurt our predictions. The final scores for best runs of each model were as follows:

-Random Forest Classifier .987 train, .942 test

-Logistic Regression .882 train, .797 test

-Support Vector Classifier .808 train, .753 test

-Multinomial Naïve Bayes .723 train, .699 test

Hyperparameters for these are available in the jupyter notebook and slide deck.

Word clouds were constructed using the same proportions as found in each corpus. These had no predictive value, but made for a nice visual.

The datetime was converted from UTC seconds into Eastern time, day of week, month, date, year, and time columns. This could be used in future predictive models along with the pseudonym of ‘author’. Another experiment that may be useful to run would be a sentiment analysis of comments based on time of day, day of week, etc.