MAIS 202 - Final Project

Zahur Ashrafuzzaman, Jules Barbe, Benjamin Segall

Deliverable 3

1 Final Training Results

For the previous deliverable we just used SGD classification based on an SVM for just the body, so we first started trying to improve the accuracy by integrating the article title into the model. Out of curiosity, we tried using *only* the title and disregarding the body, which still ended up working surprisingly well. With stop word removal, the accuracy was about 94.3%, and with a lot more false negatives than false positives. When we didn't remove stop words, the accuracy increased, hovering around 95.5%. It's also worth noting that not removing stop words when using the body also increased accuracy a bit.

To combine the title and body, we first just concatenated the two and used that. This improved the accuracy by about 0.2%, putting it consistently over 99.1%. But, this just treated the title as normal text and we wanted to give it a bit more weight. To do this, a method we came up with and implemented was to concatenate not just one copy of the title to the body, but multiple, so that the words in the title would show up with a higher frequency. After playing around with different multiplicities of the title, we found that including it 3-5 times led to the best results, with an accuracy between 99.3% and 99.4%.

After this, we decided to branch out and try a variety of different models, including Naive Bayes, Perceptron, and SGDs with different penalties. The worst model by far was the k-nearest neighbors, and there were a couple that did slightly better than our original. Using a random forest led to one of the best results, with an accuracy of around 99.%, but the main issue was that it took quite a long time to execute compared to the others. Very close in accuracy to the random forest but slightly better (and significantly faster) was a linear SVM with l1 penalty. These both gave about 40 incorrect classifications out of a testing set of around 9000 samples.

We finally started the implementation of Latent Semantic Analysis, where we can use our preprocessing function to preprocess the current article as well as candidate articles from trustworthy sites. This will return the articles with the biggest topic similarity. We still have to create a database of trustworthy news site to search from as well as a method for obtaining articles from them. The most probable would be to search the news sites using topics uncovered through LSA, and after compiling some of them, run LSA on them to calculate the difference. From this we would return the 5 articles with the largest similarity to the original article.

2 Final Demonstration Proposal

For the demonstration of our project, we were considering two options, a web page and a Google Chrome extension. For either of them, there's the possibility of copy/pasting the article title and body into text boxes and reading from there. With a chrome extension, it would also be possible to parse the HTML tags on the page and extract the title and body to put through our model.