Colin Hwang

ECE467 Project 1 Write up

**Background:**

I used python to write my program and used Linux to develop my system. I used naïve bayes as my basic machine learning method, and you can run my program with: $ python3 tc\_ColinH.py. The only library that I used was NLTK, which I installed by doing: pip3 install nltk.

**Word Tokenization:**

I tokenized train and test documents by using NLTK’s word tokenizer. To use it, I imported NLTK’s word tokenizer by doing: from nltk import word\_tokenize. To tokenize the words in the document, I did something resembling: tokenizedDoc = word\_tokenize(testDoc) OR word\_tokenize(trainDoc). This creates a list of strings inside tokenizedDoc consisting of strings NLTK believes to be words.

**Smoothing and Other Parameters/Features:**

In my system, I used a combination of Laplace smoothing, filtering out stop words, stemming, and case insensitivity to improve overall accuracy.

Laplace Smoothing:

Smoothing is important in Naïve Bayes to avoid zero probabilities. I implemented Laplace smoothing in my program with the general formula: P(t|c) = where a is the smoothing factor.

Stop Words:

To filter out stop words, I used NLTK’s list of stop words. To use this list, I downloaded and imported NLTK’s stop words by doing: nltk.download(“stopwords”) followed by from nltk.corpus import stopwords. Then, to get a tokenized document without stop words, I did: [word for word in tokenizedDoc if word not in stop\_words].

Stemming:

Similar to how I filtered out stop words, I used NLTK’s implementation of stemming to reduce words to their root. I imported this implementation into my system by doing: from ntlk.stem import PorterStemmer followed by stemmer = PorterStemmer(). Then, to stem words in a tokenized document, I did: stemToken = Stemmer.stem(token).

Case Insensitivity:

To avoid case sensitivity within tokenized documents, I did: [word for word in tokenizedDoc if word.casefold() not in stop\_words].

**Results of All Data Sets:**

To test the 2nd and 3rd datasets, I decided to randomly assign articles to the train and test lists with an 80:20 split. Looking at the first dataset, the split was seen to be approximately 33:67, but I decided that 80:20 was sufficient.

Without stemming, stop word lists, and case insensitivity:

Text

Description automatically generatedText

Description automatically generatedTable

Description automatically generated with low confidence

These are the results in order from corpus 1 to corpus 3. The results of both corpora 2 and 3 are abnormally good. Although I made sure not to overlap the documents and randomized selection, it seems as though the way the test and train documents were made may have affected results.

With stemming and stop word lists:

Text

Description automatically generated with medium confidenceA screenshot of a computer

Description automatically generated with low confidenceText

Description automatically generated

These are the results in order from corpus 1 to 3. The results from corpora 1 and 2 slightly improved, but corpus 3 got slightly worse. This is the exact program I submitted to Professor Sable for a presubmission, and got the results 88.94, 85.84, 92.45. After comparing the two results, I decided that my test and train documents were flawed for corpora 2 and 3 due to the drastic improvement from the actual test cases and mine. I decided to stick with stop word lists and stemming since it improved for corpus 1.

With case insensitivity and changed stop word lists:

While searching for ways to further improve the results of my program, I noticed that several words from NLTK’s stop word list may negatively affect results. Corpus 2 performed the worst in the presubmission, but words such as “in”, “out”, and “into” were filtered out. I decided to take out various stop words that had I deemed to be significant enough to include.

A picture containing table

Description automatically generated

Changing the stop word list appeared to improve certain categories’ accuracies, however often decreasing the accuracy of other categories as a result. This was also the case for case insensitivity. Since corpora 2 and 3’s results were deemed unreliable, I decided not to take the risk and stick with my original presubmission.