Report : Assignment – 3

Training the Naïve Bayes Classifier Model

First, I built the Naïve Bayes Sentiment classifier, using an example which is described on pages 79-80 (Section 6.3, Worked Example) of the textbook. This involved calculation of the prior probabilities, and then calculation of the likelihoods and finally the class chosen by a Naïve Bayes classifier for a given sentence, the equation for which is given on page 78 of the textbook (Equation 6.9).

Next, I extended the program to be able to take as input both the training sets that were provided, and when I executed the classifier on the positive and negative training sets after training on both, I got accuracies of 62% and 80% respectively (on training sets).

In other words I built a unigram language model for each of the two classes (positive and negative), and assessed the probability of each test case/sentences with respect to each of the two models. The program then predicts the highest probability class as the correct choice.

Improvements for training

Then I changed the method for populating the likelihoods slightly - initially it was taking the punctuation marks into the likelihoods for the words (e.g. if the 'here' is a word at the end of a sentence which appears with a positive review, then it was adding ('here.' , pos) as an entry to the likelihoods dictionary). Later I ensured the punctuation marks were not considered (i.e.. this time it would just add ('here',pos) as an entry), leaving out the full stop – this was done using the regular expressions library 're' from Python.

When I tried to print some of the final probabilities for certain sentences (the probabilities based on which the Naïve Bayes classifier predicts the class), I saw that some of them, especially for long sentences were showing 0. I observed this being related to the fact that I was multiplying the likelihoods for all the words in a given sentence, and the likelihoods being probabilities, the final probabilities were turning out to be really small numbers for large sentences. Beyond a certain size for sentences, the console was printing 0 for these probabilities. This led to my classifier incorrectly predicting the sentiment for those sentences. The solution was to take the logarithms of these probabilities, and then adding them up, including adding the logarithm of the prior probability too – this was done using the 'math' library in Python.

Finally, when I trained on a portion of the training set (first 80% of the data), and tested on the remaining 20% (the dev set), I got an accuracy of around 90% for both the positive and negative datasets.

Stop-list

As a measure for improving accuracy, I used a stop-list. I used the list of stop-words from the NLTK package to filter out/remove the most commonly used words before applying the Naïve-Bayes (with Add-1 smoothing) algorithm.

Libraries used

The libraries that I used were the following:

(1) math for taking logarithms
(2) collections for using the Counter method
(3) re for using regular expressions to remove punctuation and special characters from the likelihoods
(4) nltk for stop-list to filter out the stop-words