Assignment 1 – NaiveBayesClassifier Report

Candidates: 143 Candidate: 128 Candidate: 76 Candidate: 144 Candidate: 46

How it works
The code for the classifier should be fairly well commented and the purpose of each function and variable. There is also a decent amount of print() messages that should give an indication of what is going on.

Evaulation Results

In Total..
25000 reviews were **scanned in total.**
Of these, **6** reviews were **discarded** because of class probabilities returned were inf or 0.0
This means **24994** reviews were **actually tested**
20753 of them were **predicted correctly.**
4241 of them were **predicted incorrectly.**

The scores have been calculated by collecting four different types of predicitons: True Positive, False Positive, True Negative and False Negative (e.g True Positive means we were corrected abour predicting the individual file to be a positive review).

Total positive precision was: 87 percent
Total negative precision was: 79 percent
Total positive recall was: 77 percent
Total negative recall was: 88 percent
Positive F-measure was: 81 percent
Negative F-measure was: 83 percent
Overall prediction Accuracy was: 83 percent.

These test results can be displayed by running the test_all_reviews() function in our classifier.

Currently the model does not implement log-likelihoods. We tried to implement this in the class_probabilites function, but for whatever reason the log-likelihoods reduced our accuracy to about 50 percent on all the different tests. As an alternative we have divieded certain variables used in the class probabilty by 10000 to avoid having the returned probabilities being either too small or too big (inf, or 0.0) (This can be seen in the initialize(function)) .

The classifier seems to perform fairly well given the background and the test-data. We thing that the F-measure is the best indicator of our Interestingly, its positive precision is signifintly higher then the negative one, which would indicate that it is easier for it to recognize positive reviews. The reason for this, we are unsure about at the moment. There is still, however, room for improvement. Using other Nature Language Processing methods, we would probably be able to achieve that. We have brought upon ourselves the addition of stopwords in order to cut down issues/time of processing the data. Though it seems the stopwords have little bearing on the test results, which is not unexpected. We could for instance combine words with the same meaning(synonyms) or similar words into segmented groups. Another step could be to look at the order, and sequences of words in the contex of the sentence they are used in. For instance, our classifier doesnt know the meaning of phrase«not good». It just looks at words individually and therefore is suspectible to miss out on valuable semantic information. Hence, why it is «naive».

The performance rate of the classifier is also high: It requires little amount of work/time to run through thousands of reviews. We were able to scan through and make predictions for 25000 reviews in less than 2 minutes on a 5 year old laptop, and in about 30 seconds on a more recent and powerful computer.