**INTRODUCTION**

This assignment was an attempt to build a Classifier to perform Sentiment Analysis/ Opinion Mining (i.e. to determine whether a given review has a positive/negative tone) using a Naive Bayes Classifier with the **“naive”** assumption being that “every pair of features are independent to each other” which is of course, an over simplified assumption, but is nonetheless faster than other more sophisticated methods.

The dataset given was a corpus of movie reviews split into training and test set files respectively, in which the training test files were used only for the purposes of calculating all the required **prior** probabilities as well as the **likelihood** functions (i.e. to pre-calculate ), since the Bayesian approach to classifying the new instance aims to assign the most probable target value, .



Using Bayes Theorem, we get



Now since  due to the naive assumption,

the Naive Bayes Classifier outputs



wherein there are only two vj (true or false).

In this case, a small optimization can be made by considering each word in the corpus to have simply occurred or not occurred (i.e. we **“binarize”** the occurrences of the words as occurred/not occurred) since we only need to predict whether there is a positive sentiment or not and this was how the Binary Naive Bayes was implemented.

Also, there were some words with high frequency which appeared in almost all documents (movie reviews) which doesn’t help much in predicting the overall sentiment of a review, these are called **stopwords** and removing them before processing led to slightly better results.

**RESULTS**

**Positive Sentiment (Results in %)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | **No Stopwords** | **Short Stopwords** | **Long Stopwords** |
| **Naive Bayes** | *Accuracy* | 81.36 | 82.64 | 82.29 |
| *Precision* | 85.90 | 86.56 | 86.15 |
| *Recall* | 75.03 | 77.26 | 76.96 |
| *F-Score* | 80.10 | 81.65 | 81.29 |
| **Binary Naive Bayes** | *Accuracy* | 82.99 | 83.79 | 83.35 |
| *Precision* | 87.23 | 87.23 | 86.53 |
| *Recall* | 77.30 | 79.18 | 79.01 |
| *F-Score* | 81.97 | 83.01 | 82.59 |

**Negative Sentiment (Results in %)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | **No Stopwords** | **Short Stopwords** | **Long Stopwords** |
| **Naive Bayes** | *Accuracy* | 81.36 | 82.64 | 82.29 |
| *Precision* | 77.84 | 79.47 | 79.18 |
| *Recall* | 87.69 | 88.01 | 87.62 |
| *F-Score* | 82.47 | 83.52 | 83.19 |
| **Binary Naive Bayes** | *Accuracy* | 82.99 | 83.79 | 83.35 |
| *Precision* | 79.62 | 80.94 | 80.69 |
| *Recall* | 88.68 | 88.41 | 87.70 |
| *F-Score* | 83.91 | 84.51 | 84.05 |

**INTERPRETATIONS**

It is found that the accuracy is same for both the positive sentiments and the negative sentiments, as expected since it is simply the ratio of the total number of correct classifications (true positives as well as true negatives) to the total number of samples.

It is also found that removing some stopwords (As given in the Short Stopwords file at SentimentAnalysis/shortswords.txt) improves the performance of the classifier, however as more stopwords are removed (Long Stopwords file at SentimentAnalysis/longswords.txt), the performance again starts to reduce and hence, it can be concluded that some important words were also removed in the process.

In both cases, it is found that binarizing the occurrence of the words (occurrence 1, non-occurrence 0 instead of taking the frequency into account) also optimizes the performance of the classifier, and this seems to work better because *more importance is given to whether the words appear or not, rather than its frequencies*.

**CONCLUSIONS**