**REPORT**

The dataset used for text classification is *yelp\_labelled.txt* from *Sentiment Labelled Sentences Data Set.*

Result Analysis:

1. Accuracy



About 83% of the text from test data was classified correctly by the model

1. Classification Report

|  |
| --- |
|  |

Of the total reviews classified as positive, 83% was actually positive.

Of the total reviews classified as negative, 83% was actually negative

Of the total actual positive reviews, 85% was correctly labelled by the model as positive

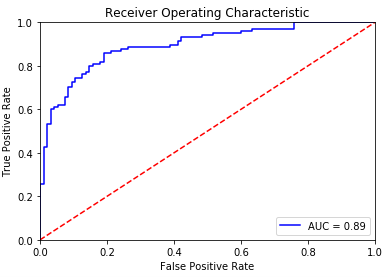
Of the total actual negative reviews, 81% was correctly labelled by the model as negative

1. Confusion Matrix

|  |
| --- |
|  |

Of the total 200 documents from the test set 77 instances were correctly labelled as positive and 89 reviews were correctly labelled as negative. The rest 34 were labelled incorrectly.

1. ROC curve



Area under the curve = 0.89

The model gives a moderate performance. It can be improved by applying these techniques:

* 1. Data Parsing (Pre-processing)
* Stemming: for instance, if you have the terms programmer, program, progamming, programmed in a given data point, a stemmer will reduce them to a single stem (probably program) so your term vector for that data point will have a value of 4 for the feature program.
* synonym finding: same idea as stemming--fold related words into a single word; so a synonym finder can identify developer, programmer, coder, and software engineer and roll them into a single term
* neutral words: words with similar frequencies across classes make poor features
  1. Feature Selection: Try n-gram model instead of unigram: Sequence of n words are considered as a token and vectorized. Best features are often pair of words or larger word groups.