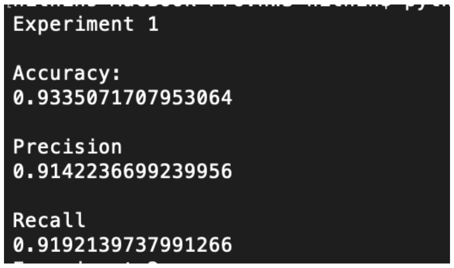
SUPPORT VECTOR MACHINE
AND
FEATURE SELECTION

Here we use support vector machine to classify the mails as spam or not by implementing on spam base dataset. We split the dataset into training and test set and we perform data preprocessing on the features, we scale it through standardization.

Experiment 1

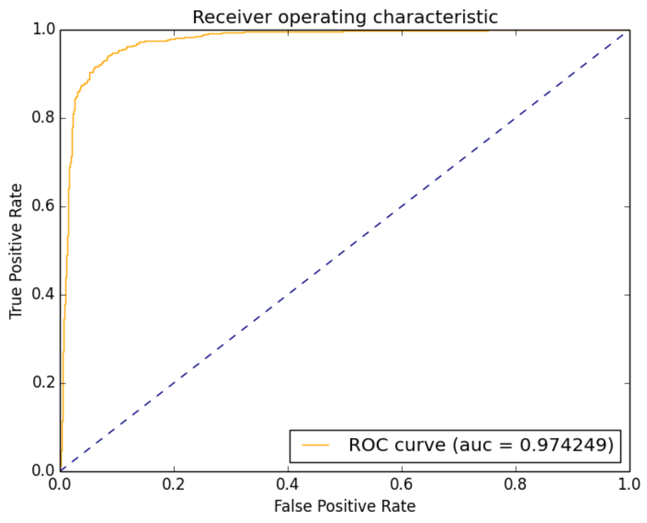
Scikit-learn svm package from python is used to implement the experiment.

svclassifier = SVC(kernel='linear')
Svm is implemented with linear kernel and a default value of c parameter. It is then fitted and the model is used to predict the test data. The accuracy, precision and recall of the learned model are given below.



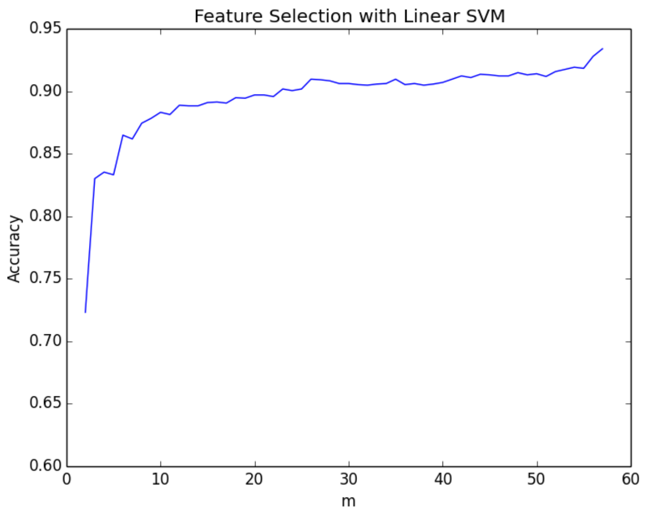
```
Experiment 1

Accuracy:
0.9335071707953064

Precision
0.9142236699239956

Recall
0.9192139737991266
```

ROC curve is created using a library function plot_roc.py from Scikit-learn



<u>Experiment 2</u>

Following is the graph of m vs Accuracy where features are selected with linear svm i.e. based on weights



Top 5 positive features (for spam) are:

3rd feature - word_freq_3d (weight - 8.57273634e-01)
percentage of words in the e-mail that match 3d,
i.e. 100 * (number of times the 3d appears in the e-mail) / total number of words in e-mail.

15th feature - word_freq_free (weight - 7.72194824e-01)
percentage of words in the e-mail that match free,
i.e. 100 * (number of times the free appears in the e-mail) / total number of words in e-mail

52nd feature - char_freq_$ (weight - 7.65052834e-01)
percentage of characters in the e-mail that match $,
i.e. 100 * (number of $ occurrences) / total characters in e-mail

19th feature - word_freq_credit (weight - 7.48215749e-01)
percentage of words in the e-mail that match credit,
i.e. 100 * (number of times the credit appears in the e-mail) / total number of words in e-mail

55th feature - capital_run_length_longest (weight - 5.55782012e-01)
length of longest uninterrupted sequence of capital letters

Top 5 negative features (for not spam) are:

40th feature - word_freq_cs (weight - -7.97717454e-01)
percentage of words in the e-mail that match cs,
i.e. 100 * (number of times the cs appears in the e-mail) / total number of words in e-mail

41st feature - word_freq_meeting(weight - -8.66162063e-01)
percentage of words in the e-mail that match meeting,
i.e. 100 * (number of times the meeting appears in the e-mail) / total number of words in e-mail

45th feature - word_freq_edu (weight - -9.52171344e-01)
percentage of words in the e-mail that match edu,
i.e. 100 * (number of times the edu appears in the e-mail) / total number of words in e-mail

24th feature - word_freq_hp (weight - -1.43050430e+00)
percentage of words in the e-mail that match hp,
i.e. 100 * (number of times the hp appears in the e-mail) / total number of words in e-mail

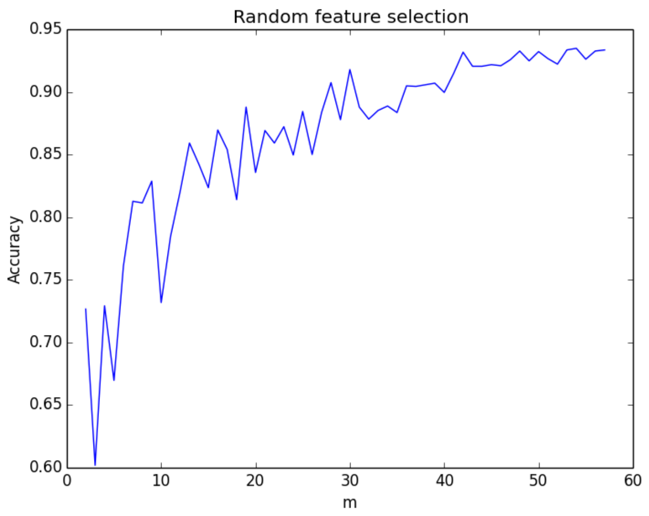
26th feature- word_freq_george (weight - -2.86751671e+00)
percentage of words in the e-mail that match george,
i.e. 100 * (number of times the george appears in the e-mail) / total number of words in e-mail

Effects of feature selection:

If there are too many features then the model overfits, this is because the learned hypothesis fits the training data well but fails to generalize with new examples. Small number of features also cause overfitting sometimes, it is when right features are not chosen and they are not correlated to the class. It is necessary to reduce features especially in large data sets in order to get rid of redundant or irrelevant features (it removes noise in the data set). The presence of irrelevant or redundant features may deviate from the optimal solution. In this experiment, we chose features based on the weights of the features. Weights are calculated using **svclassifier.coef_** of scikit learn package. We select the features with highest weights. Number of features to be selected is given by m, where m ranges from 2 to 57. Here we see as the m value increases the accuracy also increases. The model gives good accuracy and there is no distortion.

Experiment 3

Following is the graph of m vs Accuracy where features are selected at random



The graph shows distortions, we see the accuracy of the test data to be above 0.70 and increasing steadily in experiment 2 for m value between 0 and 10, It almost crossed 0.85. Accuracy of test data is 0.8839 when m is 10. Whereas in this experiment 3 graph, we see accuracy fluctuating, it decreases from 0.7266 to 0.60(m is 3) and then increases to 0.7292(when m is 4). Accuracy of the test data is 0.7318 when m is 10. More distortions or noise is seen when features are selected at random because, features are not correlated to the class and so the model fails to generalize with new data. There may be redundant or irrelevant features which causes noise.