REPORT
ASSN - 2

PRASHIT RAJ
2017CS10359

PART-A (Naive Bayes Classifier):

1. NB (Vanilla) ;

Implemented Naive Bayes on raw data using basic libraries like Counter and numpy. In the code, I have used the logarithm of probabilities to eliminate underflow and Laplace smoothing with c =1 to avoid any zero probabilities as suggested in the assignment.

Accuracy on test set = 80.5013927577 %

2.(Random and Majority Prediction):

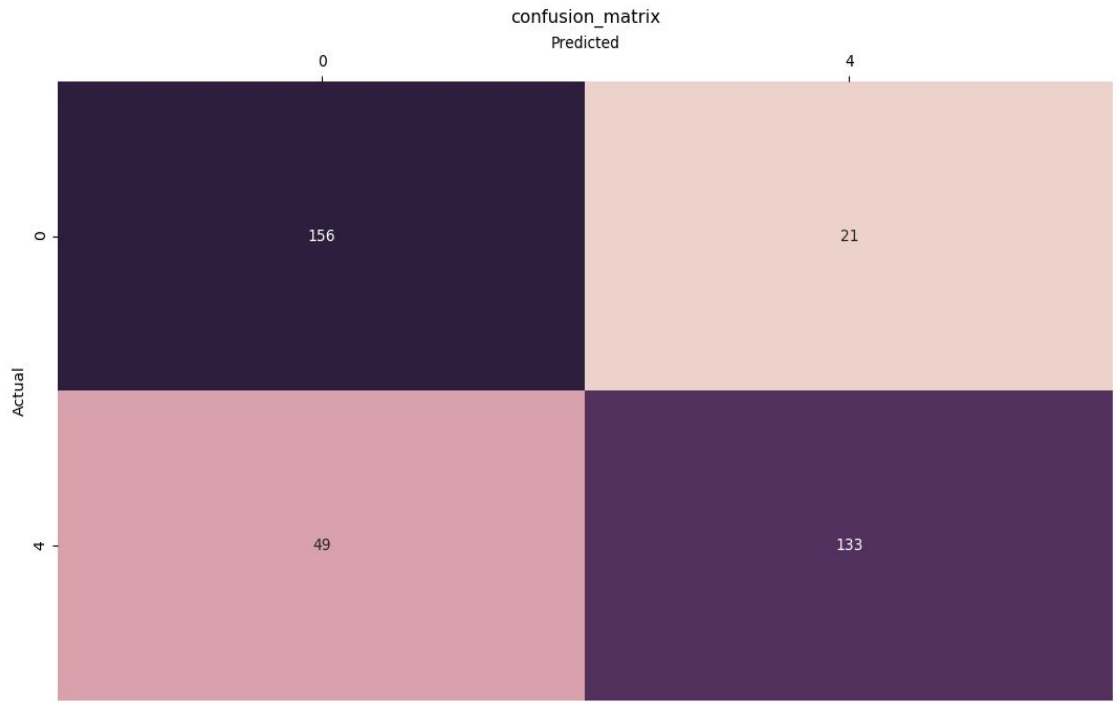
The accuracy of the test set on random prediction is 50%.

This was expected because Random prediction predicts one of the possible classes with a probability of 1/(number of classes), which is 0.5 in this case (as number of classes is 2 in the first part).

There is no major class as training data is equally distributed in two classes. If, however, either class is taken as majority one by one then the test accuracy comes out to be 0.51%(for class 4) and (0.49%) for class 0. Indicates equal distribution of data in the training and testing sets.

My Naive Bayes model gives 30% better accuracy than probability predicted by random or majority predictions.

3. Confusion Matrix:



4. Stemming:

Accuracy = 82.45%

Removed stop words, special characters and extra white spaces using regex and nltk. Converted to lower case, tokenized and stemmed using nltk.

5. Feature Engineering:

5.1. Bigram:

Accuracy on the test set = 77.5634

5.2. Feature Engineering:

In the feature, I tried two different sets of the feature. In the first part, I tried to use both bigram and each word which was what done originally. This method gives a test set accuracy of 78.9346.

In second method I tried to include emoticons as features too which gives a test set accuracy of 84.3269.

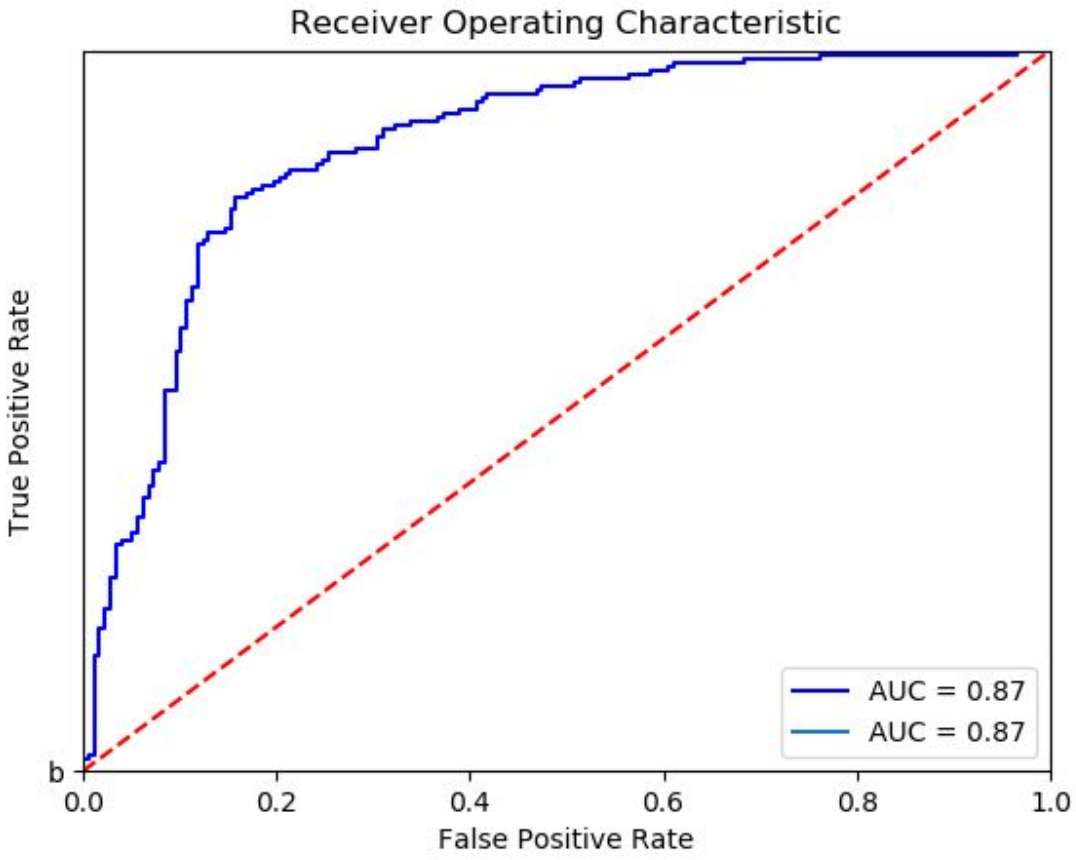
6.TFIDF:

No. of features selected = 200
Minimum document frequency = 1000
Accuracy = 72.4233983287

On decreasing the number of features the accuracy decreases. Furthermore, no more than a selection of 200 features was supported on my system showing memory error. To get rid of this I tried to use 10 batches and partially fit training cases into the NB model.

7.ROC Curve:

Below ROC curve is plotted for part 1 of this assignment.
When using normalized units, the area under the curve (often referred to as simply the AUC) is equal to the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one (assuming 'positive' ranks higher than 'negative').



PART-B (SVM Classifier):

1. Binary Classification:

The last digit of my entry number is 9 so I have done the training and testing on dataset labeled 0 or 9. The following are the results obtained for various classifying techniques used. Accuracy given is calculated on file test.csv

1.1 Linear SVM using CVXOPT:

C = 1
Accuracy = 100.0 %
Time = 419 s
b = -0.7637326861487987
Number of support vectors = 173

1.2 Gaussian Kernel SVM using CVXOPT:

C = 1
Gamma = 0.05
Accuracy = 98.6 %
Time = 416s
b = -0.8344916943993754
Number of support vectors = 844

1.3 Linear SVM using SVC:

C = 1
Accuracy = 100.0 %
Time = 6.26s

1.4 Gaussian kernel SVM using SVC:

C = 1
Gamma = 0.05
Accuracy = 99.8 %
Time = 13.2452s

2.Multi-Class Classification:

2.1 SVM with gaussian kernel using CVXOPT:

C = 1
Gamma = 0.05
Accuracy = 0.856
Time = ~4 hours

2.2 SVM with gaussian kernel using SVC:

C = 1

Gamma = 0.05

Accuracy = 0.8808

Time = 604 s

2.3 Confusion matrix:

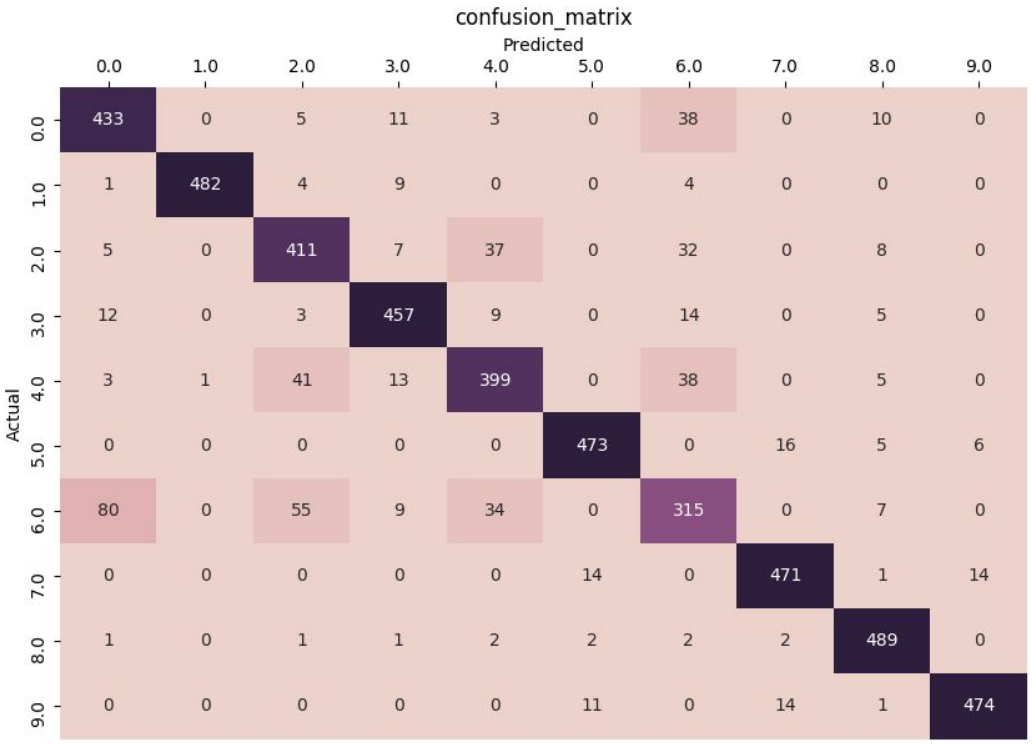
Plot of confusion matrix for the SVM classifier using SVC.

Parameters:

C= 1

Gamma = 0.05

Plot:



Analysis:

In the plot we can see there are certain irregularities and we are going to analyze them below:

Between classes:

0,6 : 0 represents t-shirt and 6 represents shirt which is very similar

2,4,6: 2 represents pullovers, 4 represents coats and 6 represents shirts, therefore, they are very similar to each other and can be misidentified by the classifier

2.4 K-fold Classifier:

K = 5

1. C = 10^{-5} :

Accuracies on k-fold : 0.0946,0.0934,0.0922,0.0953,0.09333

Average accuracy = 0.093766

Accuracy on test set = 0.092

2. C = 10^{-3} :

Accuracies on k-fold: 0.09555,0.09603,0.09704,0.0962,0.0974

Average accuracy = 0.09644

Accuracy on test set = 0.0957

3. C = 1 :

Accuracies on k-fold: 0.88,0.883,0.878,0.881,0.879

Average accuracy = 0.8803

Accuracy on test set: 0.879

4. C = 5 :

Accuracies on k-fold: 0.890,0.883,0.884,0.8720.877

Average accuracy = 0.882

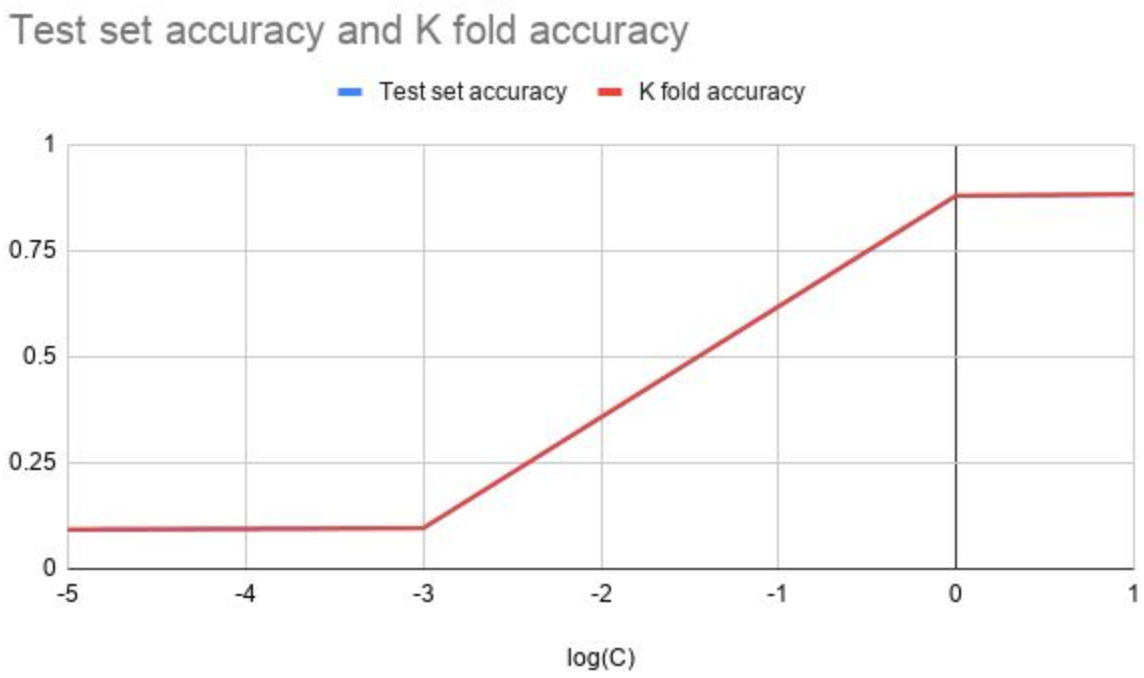
Accuracy on test set = 0.8817

5. C = 10 :

Accuracies on k-fold: 0.8815,0.8817,0.886,0.8817,0.8875,

Average accuracy = 0.8838

Accuracy on test set =



The test and k-fold accuracy lines can't be discriminated because of very similar values. The same is true for very small C's and very large C's.

We will choose C = 1 because it's giving an acceptable accuracy and the computation time is low comparatively. As seen from the trend the accuracy increases with increasing C but also the number of support vector will also increase with increasing C which in turn will increase the computation time.