MACHINE LEARNING
COL-774

Assignment 2
Report file

Submitted by:
Anshu Bansal
2018MCS2142

Question. 1 (Naive Bayes)

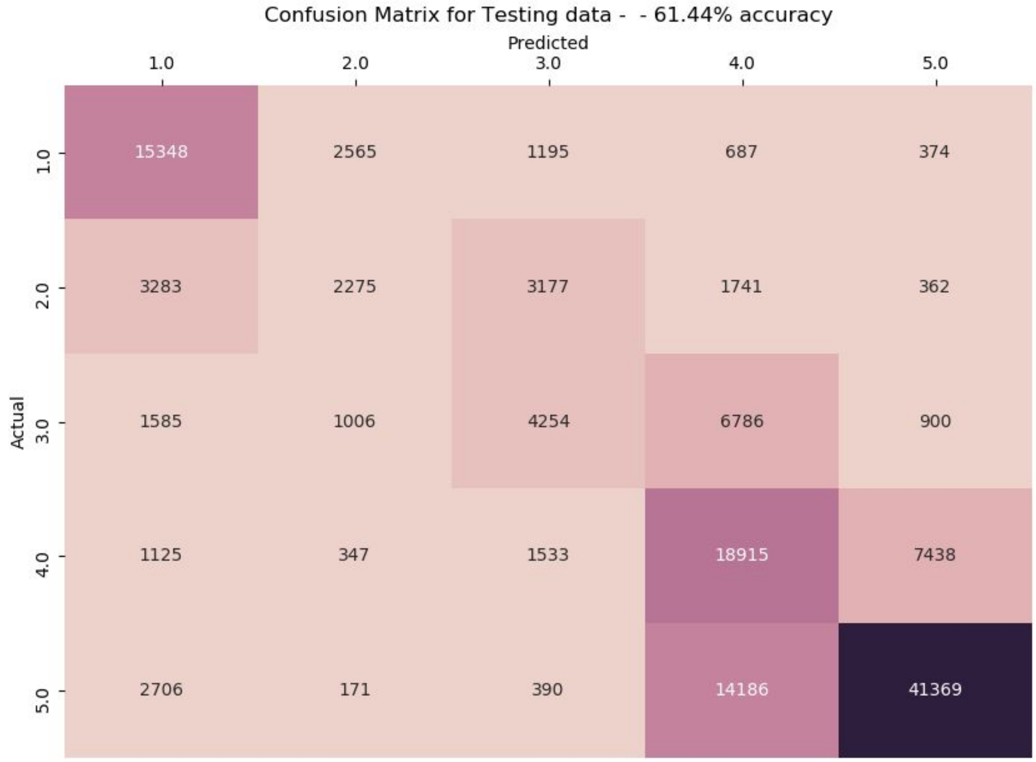
a) No Preprocessing done except removing any extra spaces

Training accuracy: 69.48727919950942 %
Testing accuracy: 61.44348554420497 %
F-score:
[0.69422833 0.26450413 0.33923445 0.52781382 0.75722326]
Macro -f -Score : 0.516600799293

b) Random Testing Accuracy : 19.87540944375477 %
Majority Testing Accuracy : 43.9895900327555 %

Improvement Over Random Prediction : 41.56 %
Improvement Over Majority Prediction : 17.45 %

c) Training accuracy: 69.48727919950942 %
Testing accuracy: 61.44348554420497 %
F-score:
[0.69422833 0.26450413 0.33923445 0.52781382 0.75722326]
Macro -f -Score : 0.516600799293
Category 5 has the highest diagonal entry in the confusion matrix and category 2 has the lowest diagonal entry in the confusion matrix. Which implies that category 5 has been classified correctly maximum number of times and category 2 has been classified minimum number of times .
Confusion matrix shows the number of times a class has been classified into any of the class. It is the very clear and unambiguous way to represent classifiers.
Below is the confusion matrix based on the test data :



d) Two major transformations in the test as well as train data:

(i) Stopword removal

(ii) Stemming

I performed both these transformations on both training as well as testing data simultaneously using the script provided. But these transformations didn’t turn out to be useful as the time taken to train the model increased while the accuracy on the testing data fell down.

Training accuracy: 63.84144243856474 %

Testing accuracy: 59.893955937121405 %

e) The two alternative features which has been used for the training and testing as below:

1) Bi- Grams :I tried to construct bi-grams since the probability of occurring of 2 words together may help us predict the class

better. The overall improvement in accuracy wasn't visible enough. Moreover the training time increased significantly.
Accuracies using **bigram without stemming data** -
Testing accuracy: 64.28453910468299
Accuracy using **bigram with stemmed data** -
Testing accuracy: 63.91809629219701

2) TF-IDF : I have tried this feature with and without stemmed data too. The results are as below :
Accuracies using **TF-IDF without stemming data** -
Testing accuracy: 58.61738883321617
Accuracy using **TF-IDF with stemmed data** -
Testing accuracy: 58.693668765611214

f) F-score : $0 \leq$ F-score ≤ 1

If F-score is 1, it is a perfect classification.
F-score is harmonic mean of Precision and Recall, where Precision and Recall are:
Precision = True Positive/ (True Positive + False Positive)
Recall = True Positive/ (True Positive + False Negative)
True Positive: A positive category correctly predicted as positive.
True Negative: A negative category correctly predicted as negative.
False Positive: A negative category wrongly predicted as positive.
False Negative: A positive category wrongly predicted as negative.
In classification, accuracy may not always give us the best insight into the model because if it is some sensitive data then we can't afford to predict false negatives.
F-score gives us the balance between precision and recall, thus, it can be a better performance measure of a model in various situations.
For our dataset, since it is not a highly sensitive data like cancer case prediction or something, we can go along with accuracy. However, F-score will provide us better insight into our model.

F-score for my best model for every class:
[0.74842006 0.16583773 0.23029951 0.50872989 0.80008431]
Macro -f -Score : 0.490674298702

Question. 2 (SVM: Binary Classification (2 &3))

I performed normalisation on training as well as test data data by dividing every entry by 255 to get the data in the range [0, 1].
To use the package cvxopt, we need to transform our dual problem in the form of matrices. I converted it as follows:

Min $_x$ $\frac{1}{2} x^T Px + Q^T x$

Subject to: $Gx \leqslant H$

$Ax \equiv B$

$P = y_i . y_j . x_i . x_j = K * Y$

$A = [y_1 , y_2 , y_3 , ..., y_m]$

B = 0.0

Q = [[-1],
[-1],
[-1],
....[-1]]

G = [[-1 0 0 0 .... 0]
[0 -1 0 0 .... 0]
.
.
.
[1 0 0 0 .... 0]
[0 1 0 0 .... 0]......[0 0 0 0 ...... 1]]

H = [0
0
.
.
.
C

C]

a) Parameter b = 0.874962480678

w = (alpha * y).T X trainData for all i

Time for training = 1546.7453303337097 seconds

Accuracy on testing = 96.62095984329089 % (1973 / 2042)

b) Parameter b: [0.00100077]

$W = \sum_i \alpha_i y_i \Phi<x_i>$ for all i in support vectors

Time for training = 2768.910334587097 seconds

Accuracy on testing = 96.89666993143977 % (1976 / 2042)

c) using LIBSVM :

(i) Linear kernel :

Test set accuracy = 97.0127% (1981/2042)

Intercept = -1.1624707095500089

Training time = 2.0189993381500244 seconds

(ii) Gaussian kernel :

Test set accuracy = 99.2654% (2027/2042)

Intercept = -0.18614008430101237

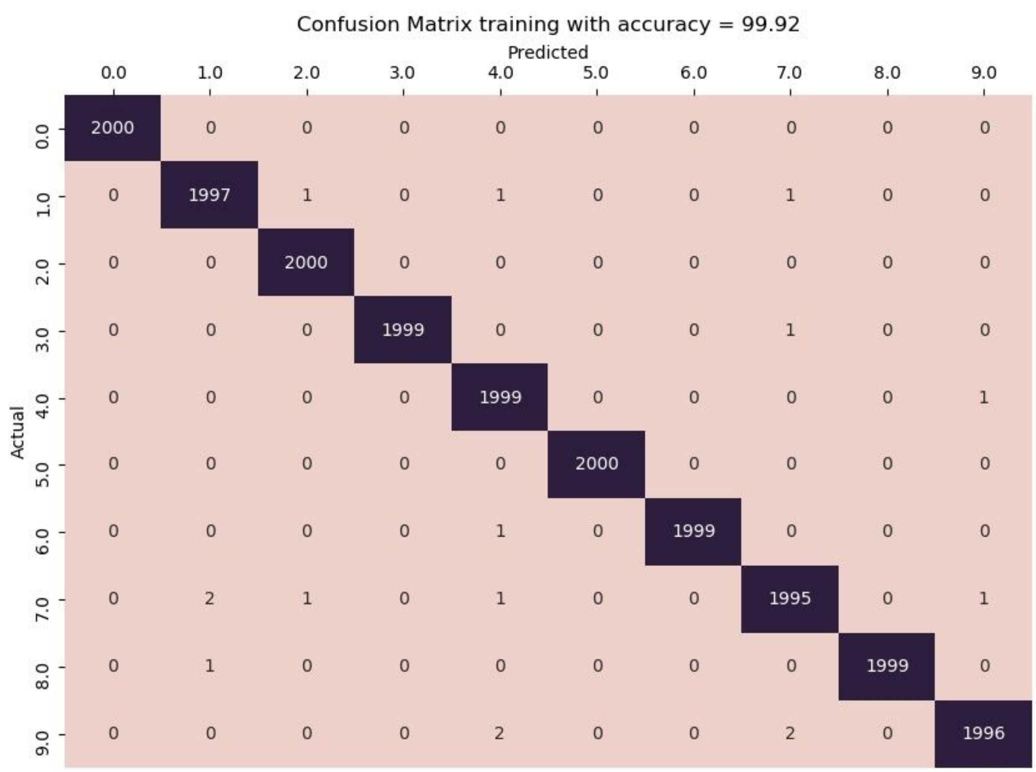
Training time = 9.476181507110596 seconds

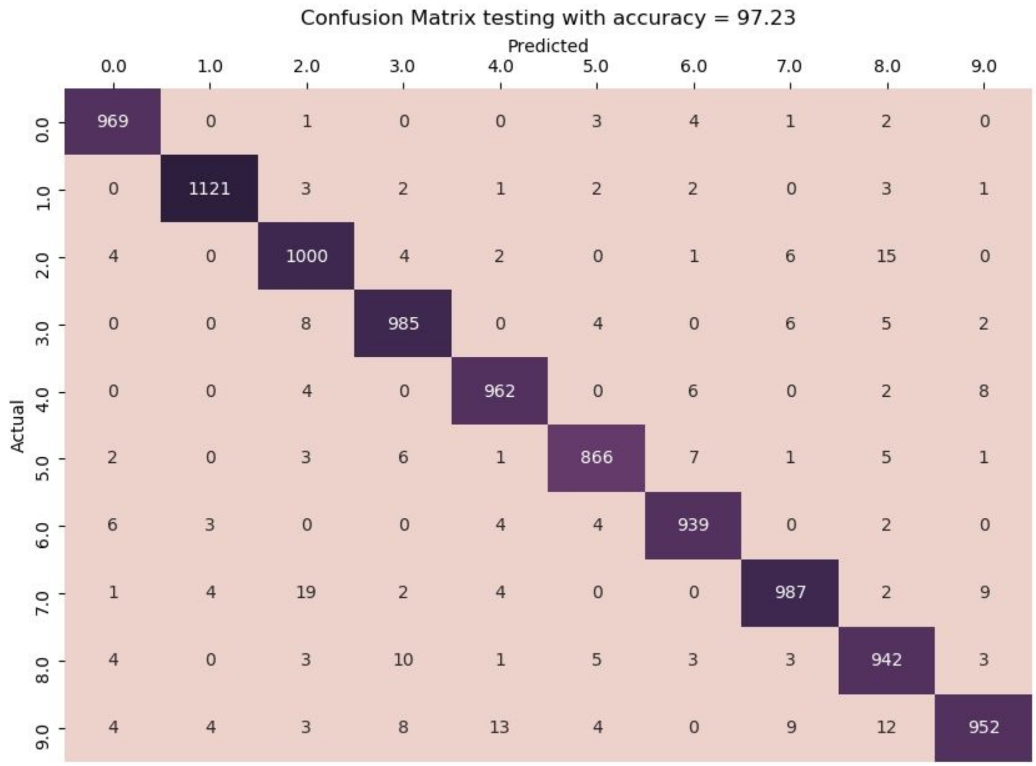
LibSVM is quite faster than cvxopt. Moreover, it seems to give better accuracy in both linear as well as gaussian kernels.

Question. 2 (SVM: Multi Classification)

b) Using LIBSVM Gaussian Kernel →
With gamma = 0.05 and c = 1.0
Training time = 218.67938089370728 seconds
Training Accuracy = 99.92% (19984/20000)
Testing Accuracy = 97.23% (9723/10000)

c) Using LIBSVM Gaussian Kernel →
Training time = 241.52894949913025 seconds
Training Accuracy = 99.92% (19984/20000)
Testing Accuracy = 97.23% (9723/10000)
Confusion Matrix as below :





The confusion matrix looks good as the diagonal entries this time are quite large as compared to what they were last time. Which means that prediction has almost been accurate. Though there are some misclassifications but not much.

7 has been misclassified as 2 maximum number of times.

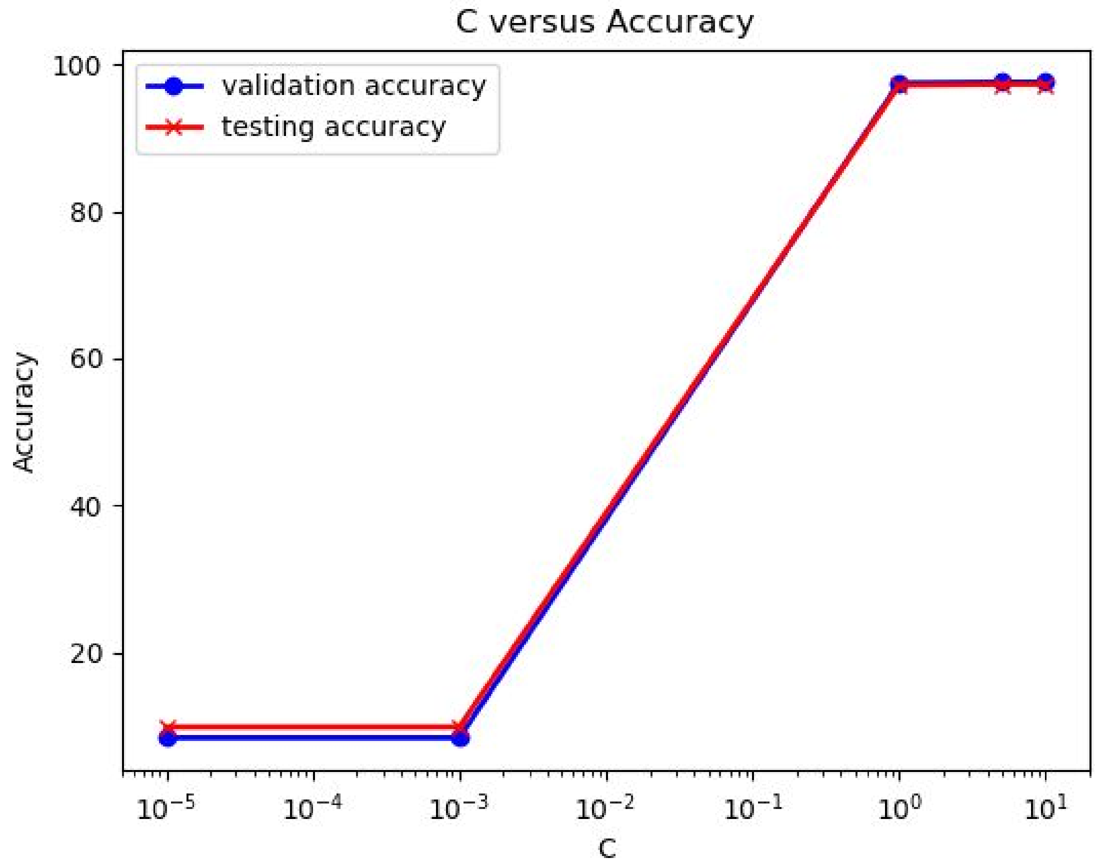
This is so because the MNIST images of these numbers are quite similar which leads to wrong prediction as they have the similar structure.

d) C = {0.00001, 0.001, 1.0, 5.0, 10.0}

I sampled last 10% of training data as validation set.

Validation_accuracy = [8.35, 8.35, 97.45, 97.55, 97.55]

Test_accuracy = [9.8, 9.8, 97.11, 97.25, 97.25]



The accuracy over validation and test set are almost similar. Test set accuracy is slightly higher than validation set.
For the values of C,
as C is low, the accuracies are low and are similar over the range of C. For C = 1 the accuracy value rises suddenly and stays
almost same for C=5 and higher values of C.
After certain value of C, accuracy kinds of get saturated and remains the same.