Assignment 10

Swapnil Asawa, swa12

Problem 1

a)

Submitted Fisher_Score function.

Code for part 1a: hw10_1a

Top 20 dims with their fisher_scores(sorted according to Fisher score)

48	0.319180730643819
25	0.214007577983443
21	0.190969964698631
70	0.189213760572332
65	0.169320028160465
40	0.167344782823831
29	0.165045125172703
19	0.140196441194676
57	0.125453639029847
20	0.121208166902893
24	0.0995248083025423
30	0.0950242089262554
12	0.0858179354076155
47	0.0846436482183233
61	0.0606773370724610
10	0.0579159433174404
34	0.0526784350453484
27	0.0462159428948702
39	0.0461136373408911
41	0.0421851727680418

Or sorted according to the dimension:

10	0.0579159433174404
12	0.0858179354076155
19	0.140196441194676
20	0.121208166902893
21	0.190969964698631
24	0.0995248083025423
25	0.214007577983443
27	0.0462159428948702
29	0.165045125172703

30 0.0950242089262554
34 0.0526784350453484
39 0.0461136373408911
40 0.167344782823831
41 0.0421851727680418
47 0.0846436482183233
48 0.319180730643819
57 0.125453639029847
61 0.0606773370724610
65 0.169320028160465
70 0.189213760572332

b)
Submitted AUROC_score function.
Code for part 1b: hw10_1b

AUROC (sorted according to AUROC score):
25 0.733986318407961
48 0.713308457711443
40 0.688743781094527
29 0.683690920398010
21 0.683302238805970
67 0.672963308457711
70 0.670747823383085
11 0.669465174129353
47 0.666083644278607
65 0.662002487562189
12 0.645872201492537
24 0.643229166666667
39 0.641169154228856
6 0.638254042288557
19 0.631452114427861
57 0.626982276119403
20 0.620841106965174
34 0.617381840796020
5 0.616759950248756
14 0.609025186567164

Or sorted according to the dimension:

5 0.0461136373408911
6 0.0846436482183233

11	0.140196441194676
12	0.0995248083025423
14	0.0421851727680418
19	0.0606773370724610
20	0.0526784350453484
21	0.169320028160465
24	0.0950242089262554
25	0.319180730643819
29	0.189213760572332
34	0.0462159428948702
39	0.0858179354076155
40	0.190969964698631
47	0.125453639029847
48	0.214007577983443
57	0.0579159433174404
65	0.121208166902893
67	0.167344782823831
70	0.165045125172703

More than 50% of the dimensions are same. Both are measuring how separable the data is, so I expect them to have similar results. If means are near and SD is high, TPR and TNR should also be low and hence AUROC should be low as well as Fisher scores should be low. However, the algorithm used in both are different, so the results could be little different. AUROC doesn't consider mean or variance explicitly. It just moves a threshold and finds TPR and TNR.

Problem 2

For different curves the errors are coming different each time for all the following parts.

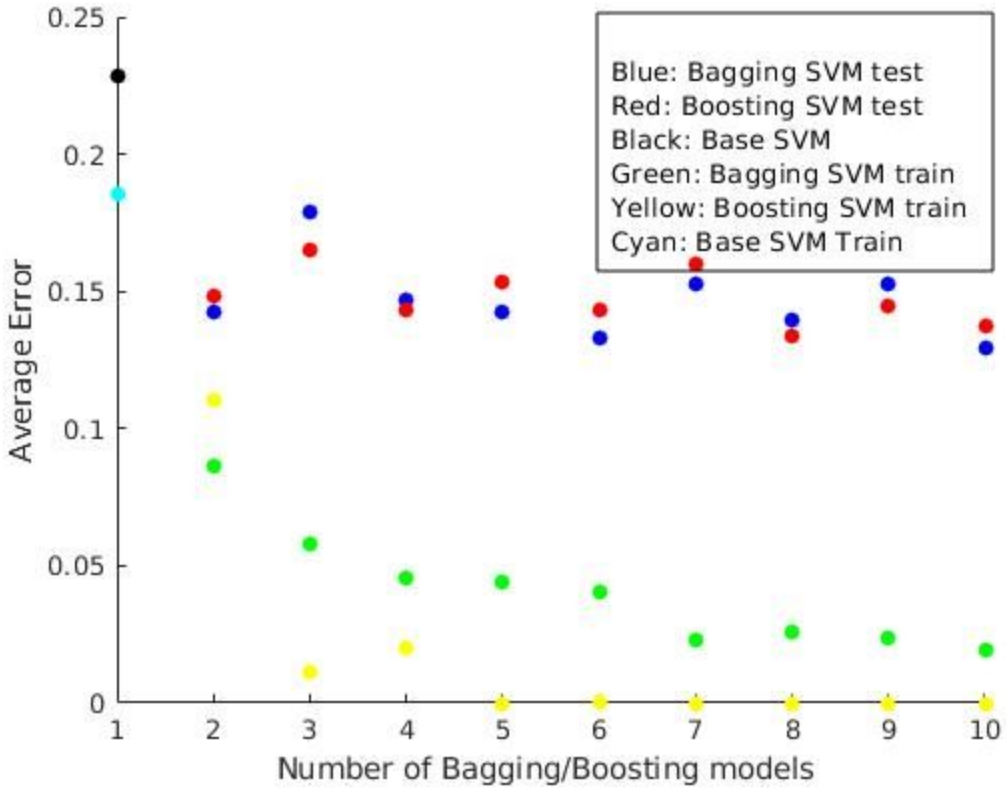
a) Code submitted

Test Error_5 =0.185714285714286,

Test Error_base =0.200000000000000,

training_Error_5 =.114285714285714,

training_Error_base =0.300000000000000



Plotted are the errors for bagged and boosted for SVM

Bagging and boosting gives better result than Base.

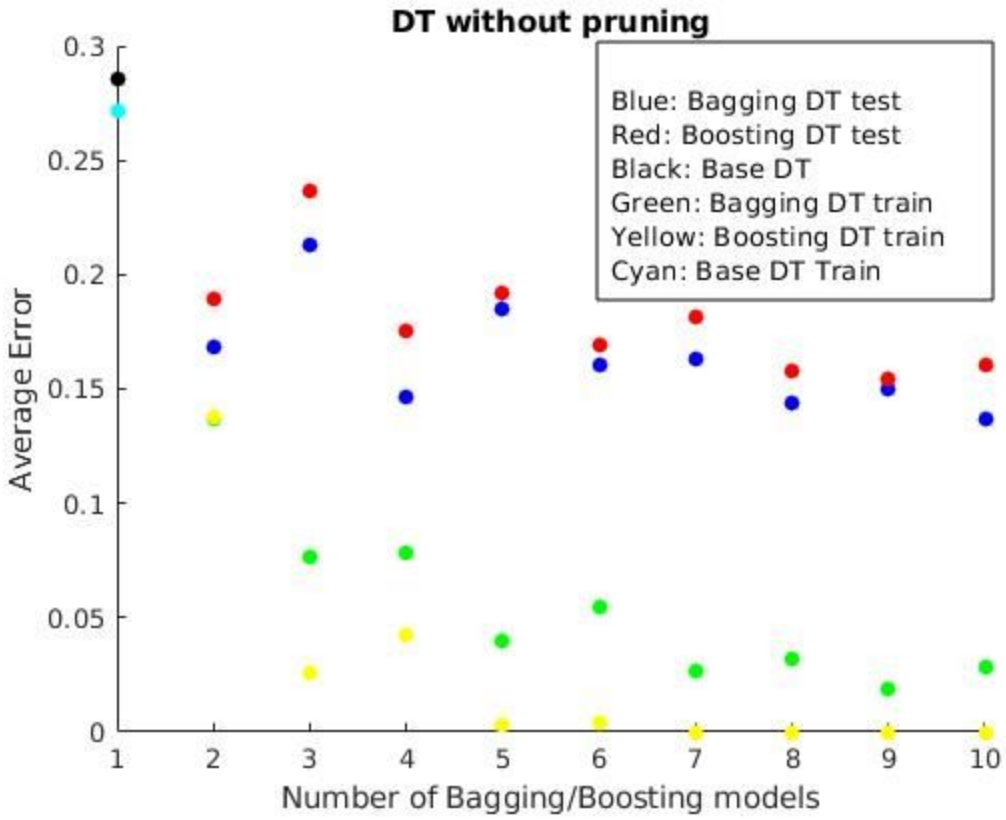
The SVM is overfitted with boosting as the training error is 0 and test is high.

The error decrease with more number of models bagging and boosting generally.

After T=6, training is decreasing but testing reached plateau bagging. Same is applicable for boosting for T=2.

b)Code submitted

Decision Trees
Plotted errors
Bagging and boosting gives better result than Base.
The error decrease with more number of models bagging and boosting generally. SVM results are better than Decision Tree. Again, The DT like SVM is overfitted with boosting as the training error is 0 and test is high. Bagging train and test both are decreasing although there is big gap between them. Bagging + regularization should help.



c)Code submitted
Decision Trees of only one split.
Plotted errors

Bagging and boosting gives better result than Base.

The error is little decrease with more number of models bagging and boosting generally. Very interesting trend it shows that test error is less than training. Here Base model has very high training error. It maybe due to less vrience in test as compared to training(the test data is easier compared to train). The algorithm **does not** overfit for bagging. Boosting training error is decreasing but increases after some steps(7 in this run) showing overfit. The gap between training and testing is lower as compared to SVM and Full Decision Trees.