ASSIGNMENT 2

FIRST DATA SET

This is the energy prediction dataset. The dataset is about usage of appliances with various parameters like temperature, humidity, wind speed etc. The target variable Appliances has been converted to a binary classification variable by taking the median which is 60 and then dividing the values above 60 as Above Average and values below 60 as Below Average. The thresholding is done with the basic assumption that those below 60 will form one half of the distribution and vice versa.

Independent variables I have used from assignment 1:

lights, T1, RH_1, T2, RH_2, T3, RH_3, T4, T6, RH_6, RH_7, T8, RH_8, RH_9, T_out, RH_out, Windspeed

The data set is taken, and mean normalization is done on the selected features. The data set divided into 70 and 30. 70% of the data it's used for training the dataset and the rest is used to test the data and evaluate accuracy of the model. For all the models I have done 5-fold cross-validation which would help us not to overfit the data and choose a good hyperparameter value.

Algorithm 1 - Support Vector Machine (SVM):

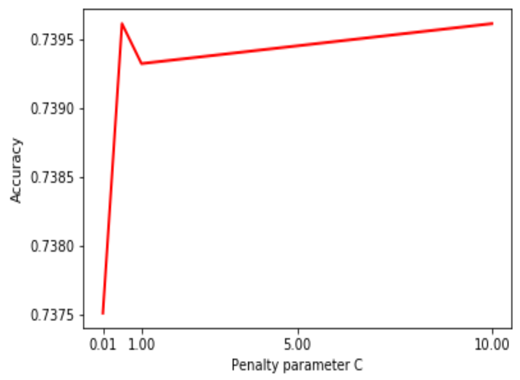
For this, I am choosing to use the sklearn package in Python to implement the algorithm as this allows easy change of kernels. I am choosing to use Linear, polynomial and sigmoid kernels.

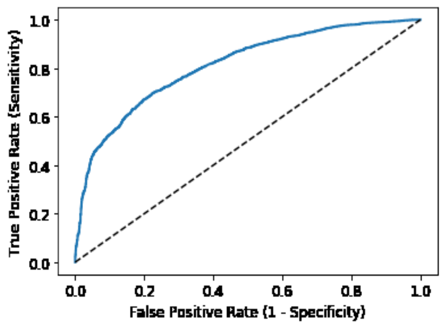
Linear kernel:

Firstly, we need to choose the value for the hyperparameter C as this is the penalty parameter. The C parameter tells the SVM optimization how much you want to avoid misclassifying each training example. For large values of C, the optimization will choose a smaller-margin hyperplane if that hyperplane does a better job of getting all the training points classified correctly. Conversely, a very small value of C will cause the optimizer to look for a larger-margin separating hyperplane, even if that hyperplane misclassifies more points. For very tiny values of C, we should get misclassified examples, often even if your training data is linearly separable. Thus, I use 5-fold cross validation to choose a good value of C. I experiment with values of 0.01,0.5,1 and 10. Below is the graph and we can see that C=0.5 gives the highest accuracy of 0.73961 and also C=10. but I choose the lower value of C=0.5 to test the model. On the right, is the confusion matrix I get when C=10.We get an accuracy of **73.79%** for this model with linear kernel. Below is the Roc curve and AUC for this model.

[1416 795]

[757 2953]]





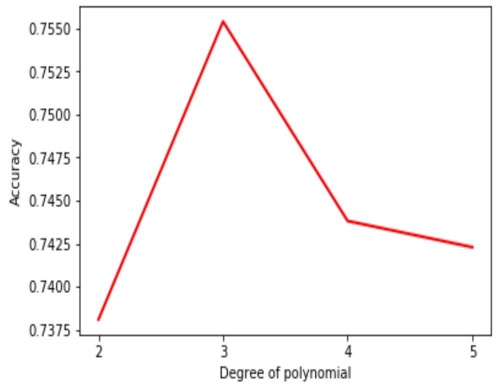
AUC - 0.8125

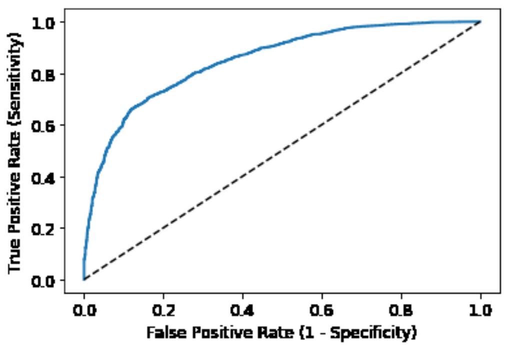
Polynomial kernel:

For polynomial kernel, we need to choose a value of degree. As the degree of polynomial increases, the model tends to overfit and fit the noise and hence the value of test error increases to a large extent. For this, we need to again run the model with different values of degree using 5-fold cross validation with degrees 2,3,4 and 5. We can clearly see that degree=3 gives the best accuracy with 75.54%. On the right is the confusion matrix and the ROC curve when we run the model with degree=3. We get an accuracy of **76.64%.**

[[1198 1013]

[370 3340]]



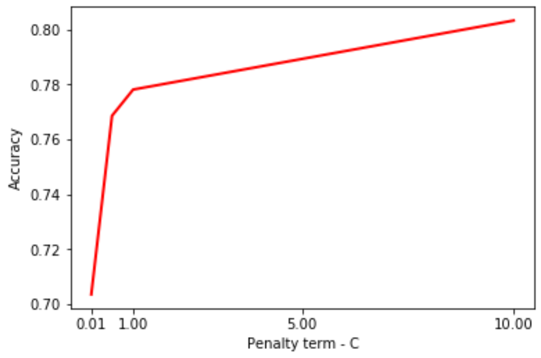


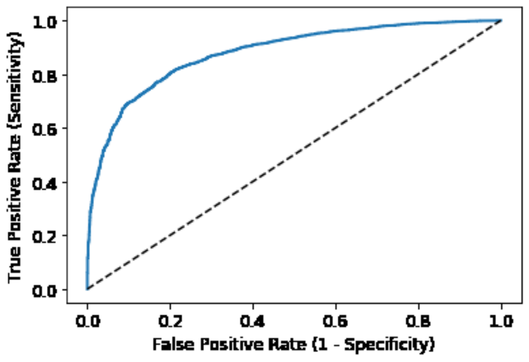
AUC -0.85136

Radial basis function kernel:

Again, we need to choose a good value of C so that the model does not overfit or underfit. We experiment with values of 0.01,0.5,1 and 10 using 5-fold cross validation. The accuracy keeps increasing as the C value is increased. We can clearly see that c=10 gives the highest accuracy of 80.32% and hence we use c=10 for testing the dataset. Below is the confusion matrix for the test dataset and the ROC curve. This gives a high accuracy of **80%**

[[1663 548]
[607 3103]]

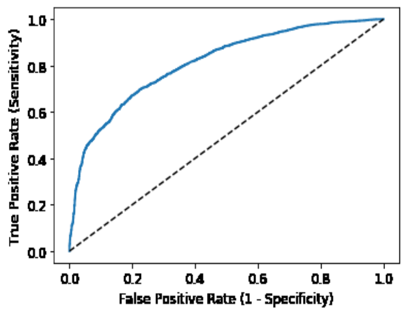


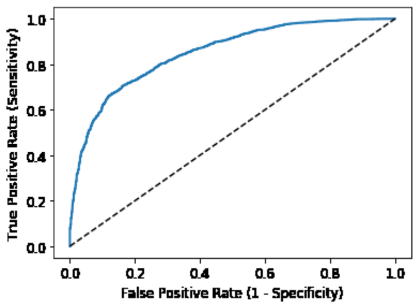


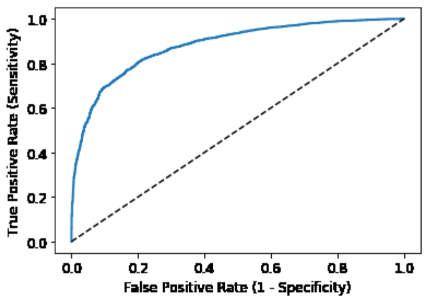
AUC- 0.88014

COMPARION BETWEEN KERNELS:

Across linear, polynomial and radial basis kernels, we see that the test accuracies are 73.79%, 76.64% and 80% respectively. We can obviously conclude that **radial basis kernel gives the best accuracy** for this dataset. Even as seen below the ROC curve is closer to the top left for radial basis kernel and has the highest AUC of 0.88.



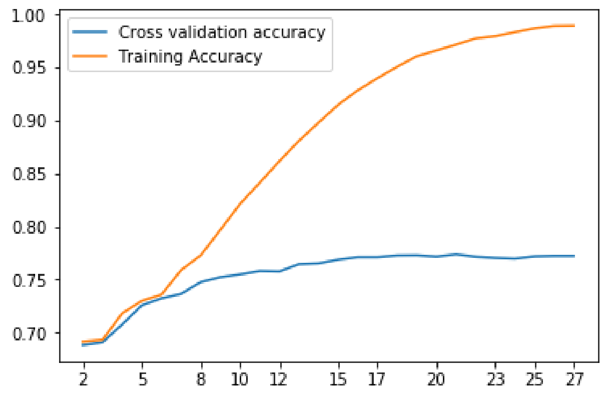




AUC - 0.81251 **AUC -0.85136** **AUC- 0.88014**

DECISION TREE ALGORITHM:

We use information gain for splitting on the variables. In information theory, entropy refers to the impurity in a group of examples. Information gain is the decrease in entropy. Information gain computes the difference between entropy before split and average entropy after split of the dataset based on given attribute values. This will help us to deal with datasets which have high imbalances. When we first run the decision tree with any parameters, we get 77.09% as the train error and 77.52% as the test error. But we need to prune the model to find a better fit for the model. So, we use cross validation and experiment with various depths from 2 to 27 to find the best fit. Below is the plotted graph while pruning the model. This is clearly a case of overfitting. After depth=7, the training accuracy begins to shoot upwards while the cross-validation accuracy is reaching a level of constancy. This means the train error is low while the test error is very high. Hence, we finally fix upon depth=7 as a good parameter for pruning.

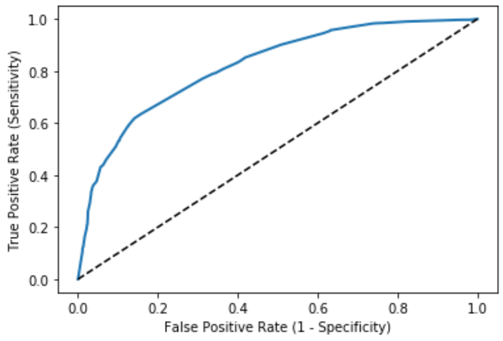


For Depth=7, we get an accuracy of 75.9% as train error and **73.66%** as test accuracy. Below is the confusion matrix for the model followed by the classification metrics and ROC curve

[[1304 907]
[581 3129]]

AUC - 0.81959

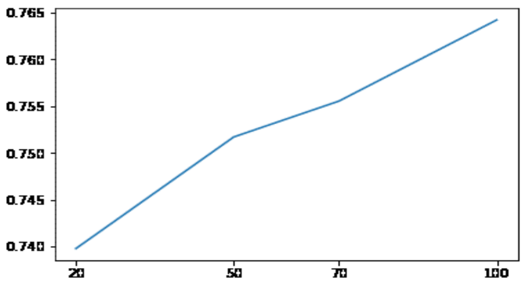
	precision	recall	f1-score	support
Bad	0.69	0.59	0.64	2211
Good	0.78	0.84	0.81	3710
micro avg	0.75	0.75	0.75	5921
macro avg	0.73	0.72	0.72	5921
weighted avg	0.74	0.75	0.74	5921



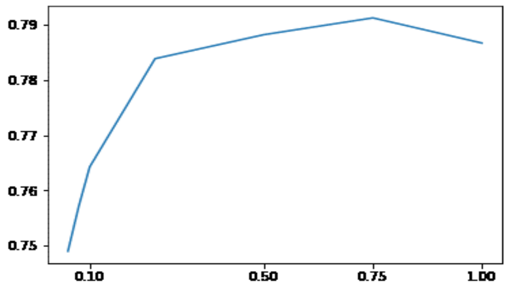
Algorithm 3: Boosted Decision Tree- Gradient Descent:

For boosted version of Decision Tree, we will use Gradient Descent. It uses gradient descent algorithm which can optimize any differentiable loss function.

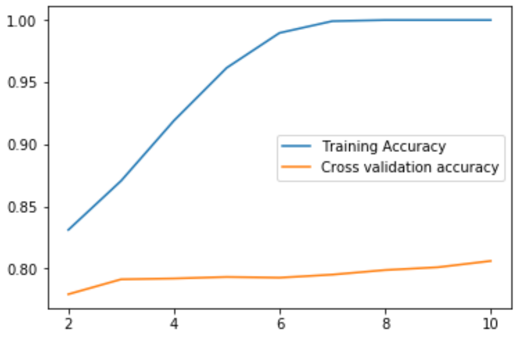
Gradient Boosting= Gradient Descent + Boosting



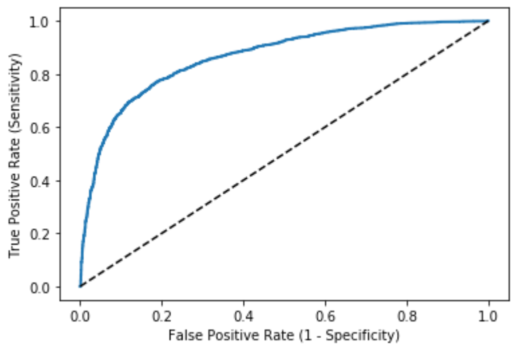
For this, we first need to choose the number of boosting stages. So, we experiment with 20,50,70 and 100 as the different values and perform cross validation. We see that n_estimators=100 gives the best accuracy at 76.5%.



Next, we need to choose the learning rate and hence we experiment with values of 0.05, 0.075, 0.1, 0.25, 0.5, 0.75, 1 and we find that alpha=0.75 gives the highest accuracy at 79.21%.



Finally, we experiment with various values of depth from 2 to 10. We can see a case of overfitting as after depth =3 the training accuracy starts to overshoot and reaches 100% accuracy whereas the CV accuracy reaches a constant. Hence, we choose 3 as the optimal depth value which gives 79.21%. We finally train the model with n_estimators=100, alpha=0.75 and depth=3. We get accuracy of **79.07%.** Below is the confusion matrix, classification metrics and the ROC curve.



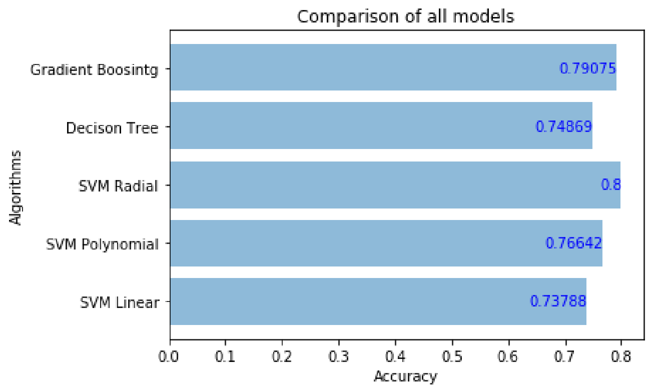
```
[[1518  693]

[ 546 3164]]
```

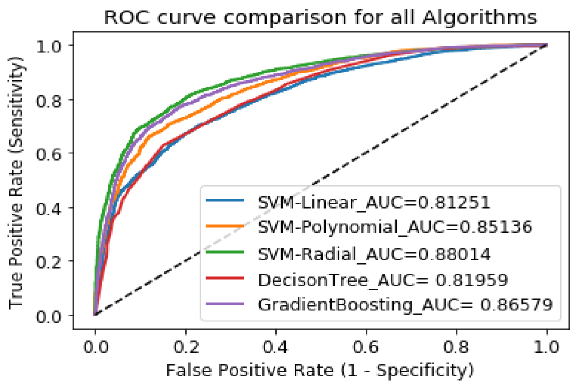
```
precision    recall f1-score   support
   Bad              0.74      0.69      0.71      2211
  Good         0.82      0.85      0.84      3710
  micro avg       0.79      0.79      0.79      5921
  macro avg       0.78      0.77      0.77      5921
weighted avg     0.79      0.79      0.79      5921
```

AUC - 0.86579

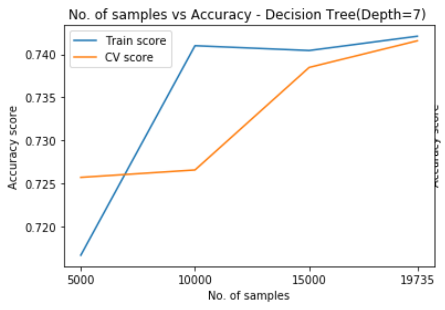
COMPARISON ACROSS ALGORITHMS:

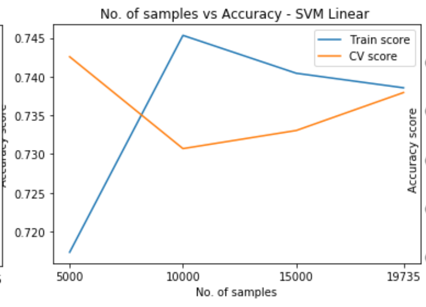
For this dataset, I have plotted the test accuracies and ROC curves for all the models below. We can see that both SVM Radial basis kernel and Gradient Boosting perform well on this dataset. SVM radial gives 80% and Gradient Boosting gives 79.1%. The AUC scores also signify the same with SVM radial giving 0.88 and Gradient Boosting giving 0.866. SVM linear performs the worst for this data because the data is not linearly separable and is nonlinear.

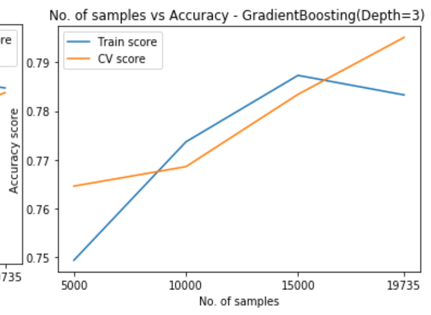




Learning curve as a function of Train size:







From the above graph we can see that as the training size increases, the CV accuracy also increases but the training accuracy keeps overfitting in the first 2 plots. Only when the size is at least 15000, the training and

CV accuracy do not show much difference. But the full dataset size of 19735 seems to be a good size as both CV and training error almost converge

CONCLUSION:

We conclude that **radial basis kernel** does best on this dataset with 80% accuracy.

SECOND DATASET

The objective is to predict whether rain will come tomorrow or not based on the location's weather conditions in Australia. The dataset provides us with various variables such as Wind speed, humidity, temperature, pressure etc. Occurrence of rain is indicated by 1 and non-occurrence is indicated by 0. I have chosen all the continuous variables for my analysis.

Independent Variables: Mintemp, Maxtemp, Rainfall, WindGustSpeed, WindSpeed9am, WindSpeed3pm, Humidity9am, Humidity3pm, Pressure9am, Pressure3pm, Cloud9am, Cloud3pm, Temp9am, Temp3pm

I did random sampling of 20000 observations because the original sample contains 1.4 lakhs. 20000 is a good sample for training the model. Mean normalization is done on the selected features and the data set divided into 70 and 30.

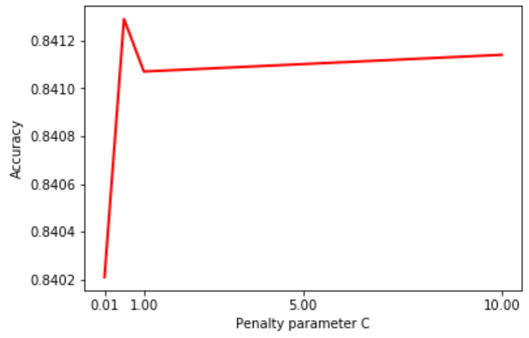
Algorithm 1 - Support Vector Machine (SVM):

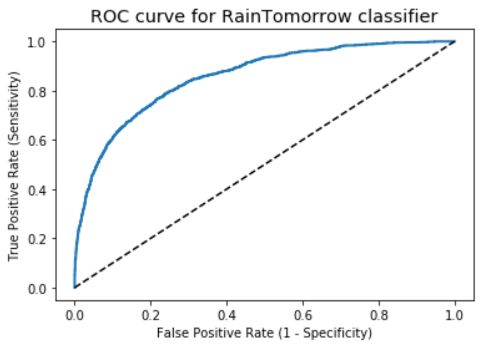
Linear kernel:

We use 5-fold cross validation to choose a good value of C. I experiment with values of 0.01,0.5,1 and 10. Below is the graph and we can see that C=0.5 gives the highest accuracy of 84.13%. Thus, I choose the C=0.5 to test the model. On the right is the confusion matrix I get when C=0.5. We get an accuracy of **84.34%** for this model with linear kernel. Also shown below is the Roc curve and AUC for this model.

[[4522 191]

[746 541]]



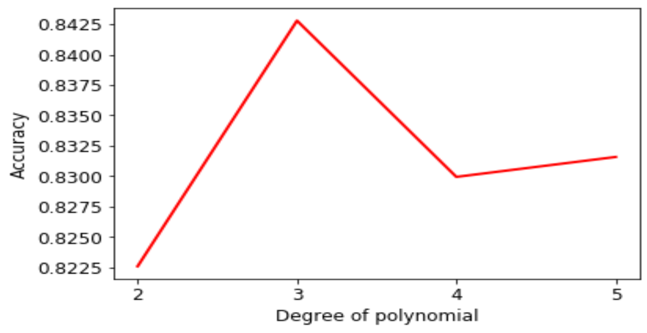


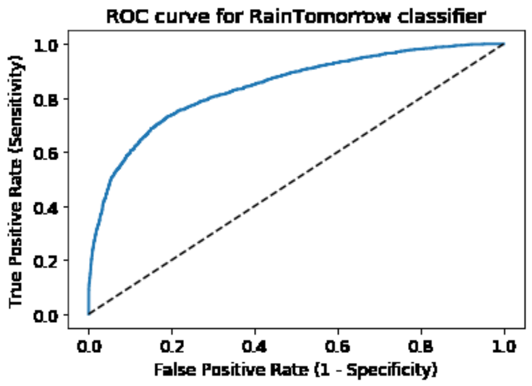
AUC - 0.85673

Polynomial kernel:

For this, we need to run the model with different values of degree using 5-fold cross validation with degrees 2,3,4 and 5. We can clearly see that degree=3 gives the best accuracy with 84.28%. On the right is the confusion matrix when we run the model with degree=3. We get an accuracy of **83.88%.**

[[4592 121]
[846 441]]





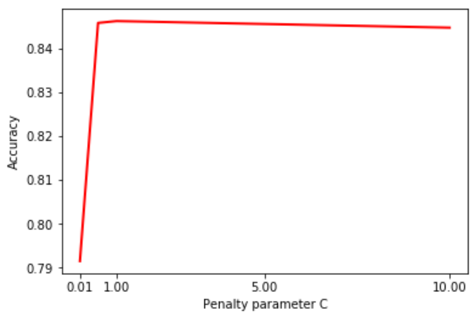
AUC - 0.84083

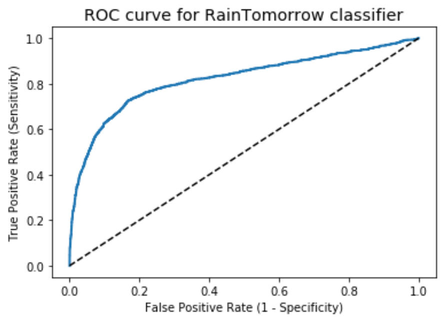
Radial basis function kernel:

The gamma values are chosen by giving auto option. But we need to choose a good value of C so that the model does not overfit or underfit. We experiment with values of 0.01,0.5,1 and 10 using 5-fold cross validation. The accuracy keeps increasing as the C value is increased. We can clearly see that c=0.5gives the highest accuracy of 84.58% and hence we use c=0.5 for testing the dataset. Below is the confusion matrix for the test dataset and the ROC curve. This gives a high accuracy of **84.8%.**

[[4546 167]

[746 541]]

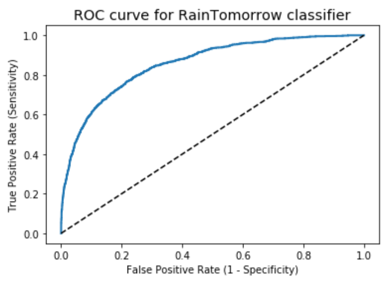


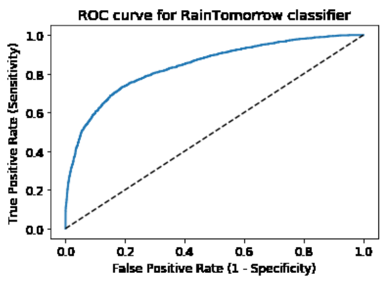


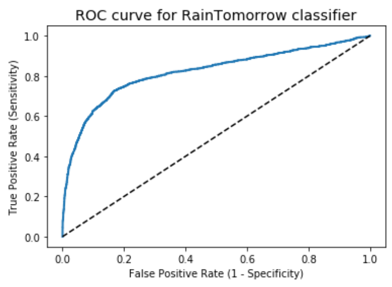
AUC- 0.82043

COMPARION BETWEEN KERNELS:

Across linear, polynomial and radial basis kernels, we see that the test accuracies are 84.38%, 83.88% and 84.8% respectively. We can obviously see that linear and radial kernel both give good accuracies for this dataset. But if we observe the ROC curves, it is closer to the top left for linear kernel and has the highest AUC of 0.857. Thus, we choose **linear kernel** as the best of all kernels

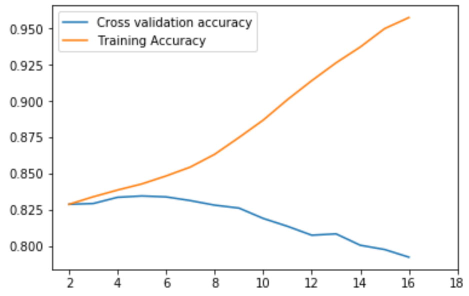






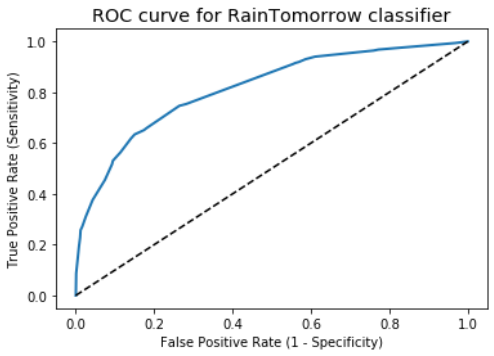
DECISION TREE ALGORITHM:

When we first run the decision tree with any parameters, we get 78.17% as the train error and 78.21% as the test error. But we need to prune the model to find a better fit for the model. So, we use cross validation and experiment with various depths from 2 to 16 to find the best fit. Below is the plotted graph while pruning the model. This is clearly a case of overfitting. After depth=5, the training accuracy begins to shoot upwards while the cross-validation accuracy is reaching a level of constancy. This means the train error is low while the test error is very high. Hence, we finally fix upon depth=5 as a good parameter for pruning. For Depth=5, we get an accuracy of 83.46% as train error and **83.23%** as test accuracy, below is the confusion matrix for the model followed by the classification metrics and ROC curve



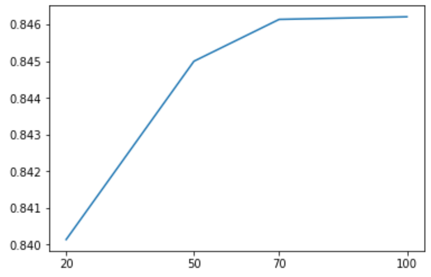
```
[[4510  203]
 [ 803  484]]
```

```
              precision    recall     f1-score  support
 0               0.85        0.96         0.90     4713
 1               0.70        0.38         0.49     1287
micro avg    0.83        0.83         0.83     6000
       macro avg   0.78        0.67         0.70     6000
weighted avg    0.82        0.83         0.81     6000
```

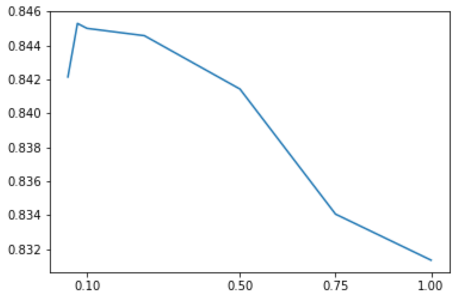


AUC- 0.81558

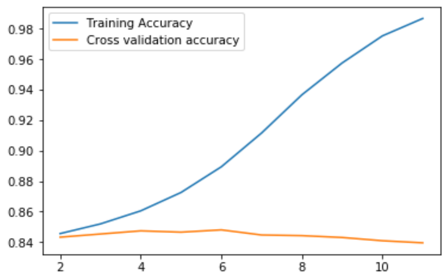
Algorithm 3: Boosted Decision Tree- Gradient Descent:



For this, we first need to choose the number of boosting stages. So, we experiment with 20,50,70 and 100 as the different values and perform cross validation. We see that n_estimators=50 gives 84.5% and almost reaches a constant after that. Hence, we fix n_estimators=50.



Next, we need to choose the learning rate and hence we experiment with values of 0.05, 0.075, 0.1, 0.25, 0.5, 0.75, 1 and we find that alpha=0.075 gives the highest accuracy at 84.53%.



Finally, we experiment with various values of depth from 2 to 10. We can see a case of overfitting as after depth =4 the training accuracy starts to overshoot and reaches 100% accuracy whereas the CV accuracy reaches a constant. Hence, we choose 4 as the optimal depth value which gives 84.74%.

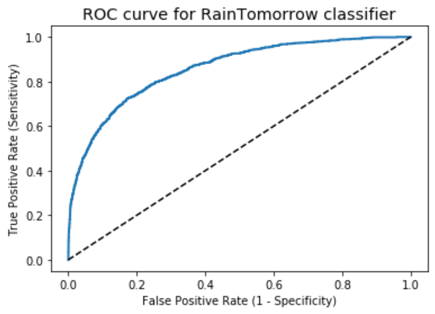
We finally train the model with n_estimators=50, alpha=0.075 and depth=4. We get accuracy **of 84.82%.** Below are the confusion matrix and classification metrics and the ROC curve.

[[4525 188]

[723 564]]

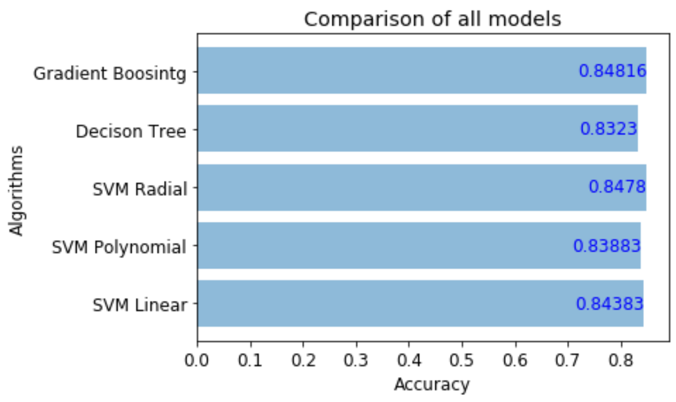
AUC- 0.85725

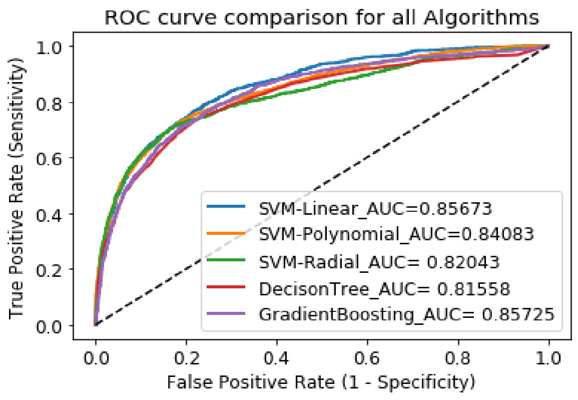
	precision	recall	f1-score	support
0	0.86	0.96	0.91	4713
1	0.75	0.44	0.55	1287
micro avg	0.85	0.85	0.85	6000
macro avg	0.81	0.70	0.73	6000
weighted avg	0.84	0.85	0.83	6000



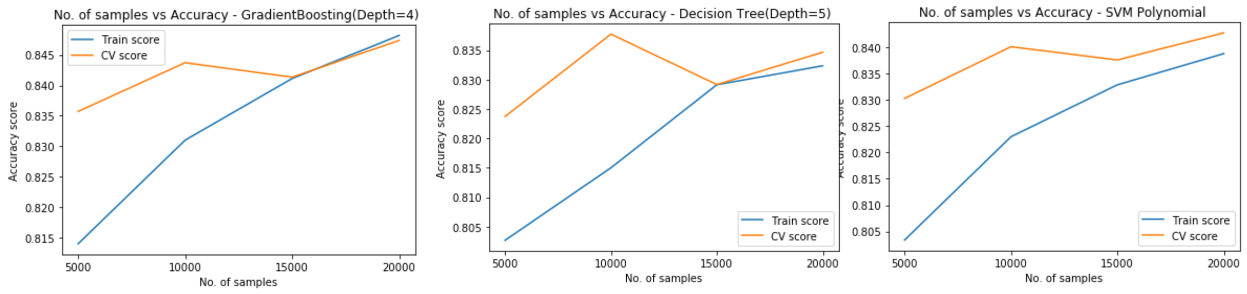
COMPARISON ACROSS ALGORITHMS:

For this dataset, I have plotted the test accuracies and ROC curves for all the models below. We can see that almost all the algorithms perform well on this dataset. SVM Linear gives the best among SVM kernels. Decision Tree though has a good accuracy rate has the lowest AUC score of all. **Gradient Boosting** with 84.82% and AUC score of 0.85725 performs best on this dataset. The reason is that this algorithm learns from the weak classifiers and improvises each stage and hence gives a better accuracy. Decision Tree performs worst for this data.





Learning curve as a function of Train size:



From the above graph we can see that as the training size increases, the training accuracy keeps increasing while the CV accuracy increases and then drops again before increasing again. Only when the size is at least 15000, the training and CV accuracy have a lesser difference and almost reach same levels. Hence, we conclude that at least 15000 points would be good for this data.

CONCLUSION:

We conclude that **Gradient Boosting** does best on this dataset with 84.82% accuracy.

THINGS THAT COULD HAVE BEEN DONE:

1) Feature selection could have been done by using forward and backward selection. Regularization could have been done to avoid overfitting.
2) GridSearchCV in Scikit-learn package could have been used to do hyperparameter tuning to select optimum values of C and gamma.

EFFECT OF CROSS VALIDATION:

Cross validation was really helpful to avoid overfitting and underfitting. For this assignment, I used CV for choosing an optimal value of C, degree of polynomial, for pruning (choosing depth) and for choosing hyperparameters of Gradient Boosting. This really helped because training the model with only training set and not using validation leads to models with high variance or high bias. Cross validation rectified this issue and as a result I obtained good models with a good bias-variance trade-off.