Objective:

The goal of this assignment is to implement 3 learning algorithms: Support Vector Machines, Decision trees and Boosting by using built-in function in python and also implementing cross-validation using the above algorithms. 2 datasets mentioned above are going to be used in this assignment.

Tasks 1:

Dataset 1: Seoul Bike Sharing Demand Data Set

Currently Rental bikes are introduced in many urban cities for the enhancement of mobility comfort. It is important to make the rental bike available and accessible to the public at the right time as it lessens the waiting time. Eventually, providing the city with a stable supply of rental bikes becomes a major concern. The crucial part is the prediction of bike count required at each hour for the stable supply of rental bikes. The dataset contains weather information (Temperature, Humidity, Windspeed, Visibility, Dewpoint, Solar radiation, Snowfall, Rainfall), the number of bikes rented per hour and date information.

Data Pre-processing:

1) Data has been converted into a binary classification by calculating the median value of column "Rented Bike Count" and at point in time, if the count of the rented bikes is greater than the median value (504.5) then, the column MedianBikes is 1 else 0. MedianBikes will act as the target variable.
2) Column Rented Bike Count has been deleted due to its exact collinearity with the new column MedianBikes.
3) All the categorical valued columns have been converted into their respective dummy variables.
4) Feature Scaling has been done for SVM Classification.

Dataset 2: Cardiotocography Data Set

The dataset consists of measurements of fetal heart rate (FHR) and uterine contraction (UC) features on cardiotocograms classified by expert obstetricians. 2126 fetal cardiotocograms (CTGs) were automatically processed and the respective diagnostic features measured. The CTGs were also classified by three expert obstetricians and a consensus classification label assigned to each of them. Classification was both with respect to a morphologic pattern (A, B, C. ...) and to a fetal state (N, S, P). Therefore, the dataset can be used either for 10-class or 3-class experiments.

Data Pre-processing:

1) There are no missing values and categorical data values, so there was no need for dummy variables or data manipulation or dealing with null values.
2) Feature Scaling has been done for SVM Classification.

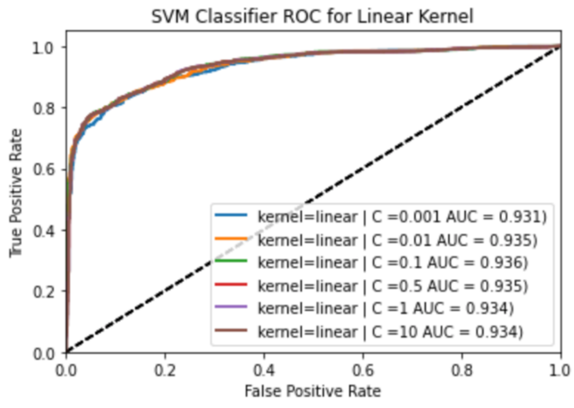
The above dataset is interesting for me because it has some real-life implications. It eliminates the need of an expert obstetrician to take a look at individual cardiotocography results. Also, it reduces the waiting time of the patients and provides an automated way to detect patients with suspect or pathologic conditions as compared to normal condition. In a way, my classification models will do some good to people and also will improve the healthcare industry's highly criticized waiting period for the patients and gives them ample amount of time to start acting on their condition.

Both the datasets have been divided into training and testing datasets in the ratio of 7:3.

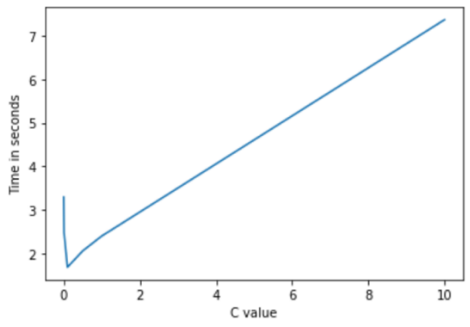
Tasks 2 - 5:

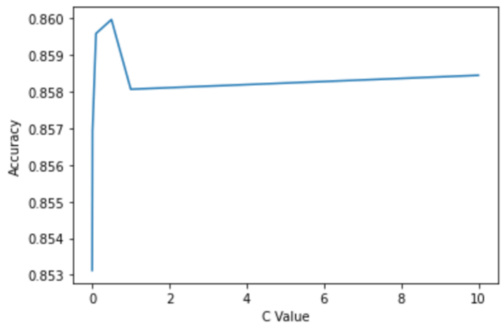
SVM Classification on Dataset 1

Linear Kernel Results:



- The plot here shows the ROC curves and the AUC for linear kernel with their respective C values.
- We can observe that **for C=0.5, the AUC is the highest**. Higher the AUC, better the model.

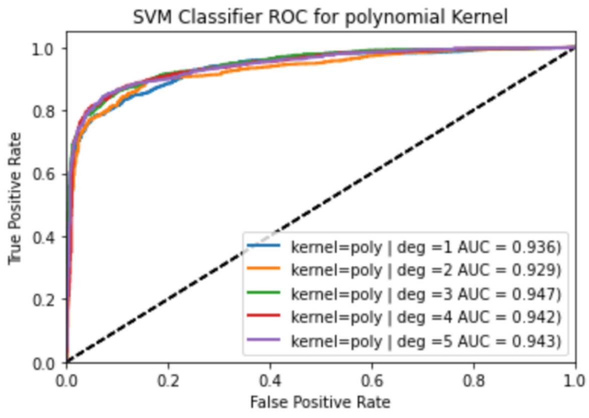




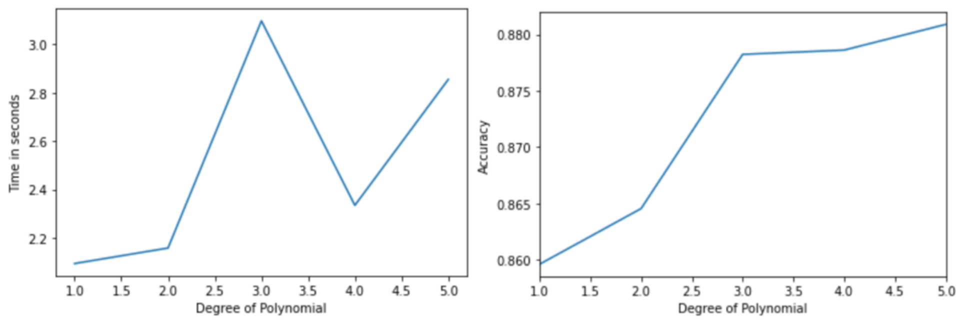
Plot 1: Describes the execution time for different C value. We can observe that as the **C value increases the execution time also increases.** However, execution time purely depends upon the configuration of the environment.

Plot 2: Describes the accuracy with respect to the C-value. We can say that with low C-values the accuracies are low, **as the C-value increases the accuracy also increases in general**. However, the highest accuracy was achieved at C=0.5.

Polynomial Kernel Results:



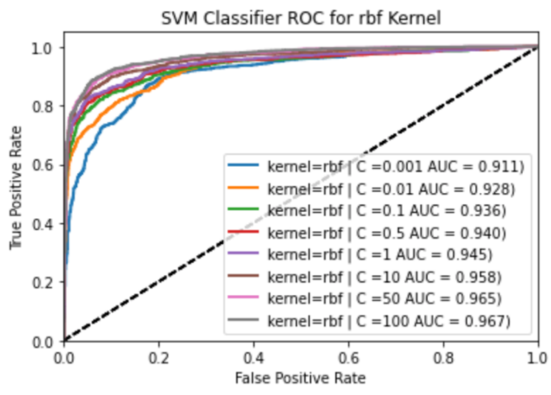
- The plot here shows the ROC curves and the AUC for polynomial kernel with their respective degree.
- We can observe that **for degree=3, the AUC is the highest**. Higher the AUC, better the model.



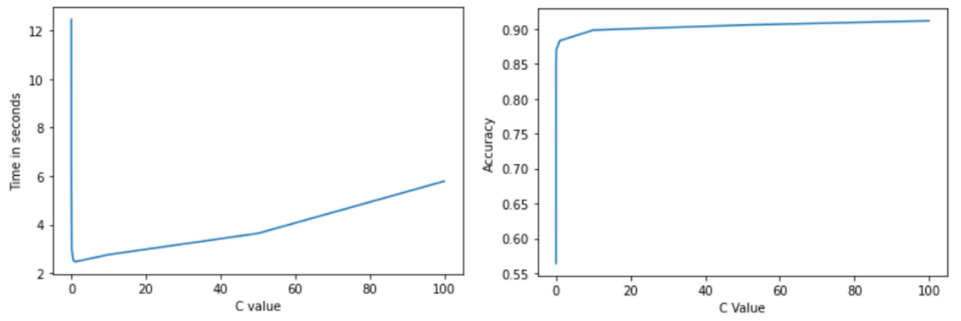
Plot 1: Describes the execution time for different degree of polynomial. We can observe that as the **degree of polynomial increases there is lot of variation in execution time.** Polynomial of degree 3 takes the maximum time.

Plot 2: Describes the accuracy with respect to degree of polynomial. We can say that with low degrees the accuracies are low, **as the degree of polynomial increases the accuracy also increases**. However, the highest accuracy was achieved at degree=5. But, in my opinion this leads to overfitting, I would say **degree =3 has an accuracy of about 0.87 and also has the highest AUC**, so that is the best value for this data set.

RBF kernel Results:



- The plot here shows the ROC curves and the AUC for rbf kernel with their respective C-value.
- We can observe that **for C=100, the AUC is the highest**. Higher the AUC, better the model. We also see that, as the **C value increase the AUC also improves**



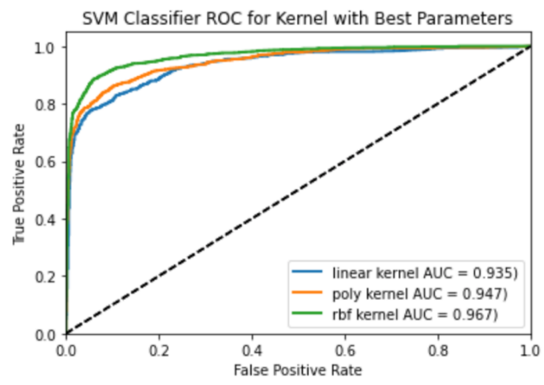
Plot 1: Describes the execution time for different C value. We can observe that as the **C value increases the execution time initially decreases for lower values of C and then gradually increases.** However, execution time purely depends upon the configuration of the environment.

Plot 2: Describes the accuracy with respect to the C-value. We can say that with low C-values the accuracies are low, **as the C-value increases the accuracy also increases in general**. However, the highest accuracy was achieved at C=100.

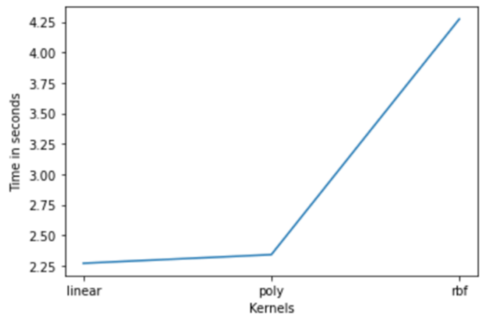
Best Model Parameters

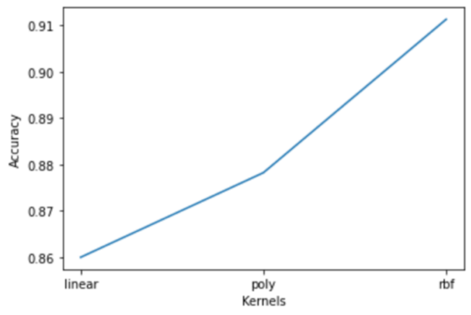
From the above information below are the best parameters for each kernel

Linear Kernel	Polynomial Kernel	RBF kernel
C= 0.5	Degree = 3	C=100



- The plot here shows the ROC curves and the AUC for linear, polynomial and rbf kernels.
- We can observe that **rbf kernel has the best AUC.**

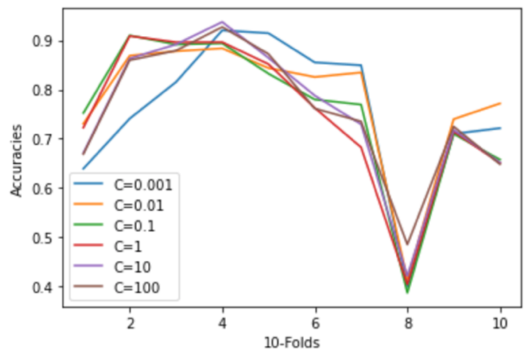




Plot 1: Describes the execution time for all kernels. RBF kernel takes the longest time to execute while linear takes the least.

Plot 2: Describes the accuracy with various kernels with best parameters. **RBF kernel results in the highest test accuracy of about 0.91 and linear kernel results in the least test accuracy of 0.86.**

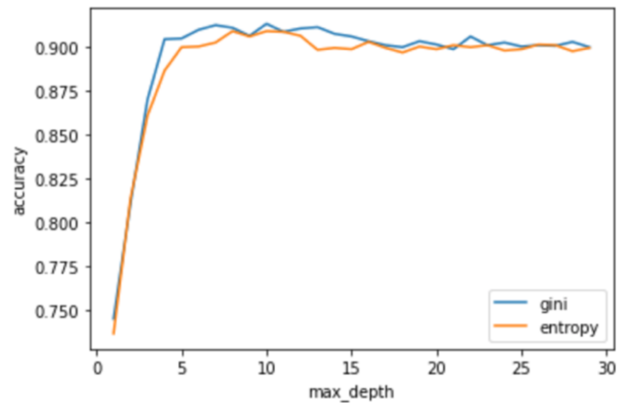
K-Fold Cross Validation:



Mean Accuracy for C-Value 0.001: 0.7565068493150684
Mean Accuracy for C-Value 0.01: 0.7789954337899543
Mean Accuracy for C-Value 0.1: 0.7583333333333334
Mean Accuracy for C-Value 1: 0.7489726027397261
Mean Accuracy for C-Value 10: 0.7535388127853881
Mean Accuracy for C-Value 100: 0.7561643835616437

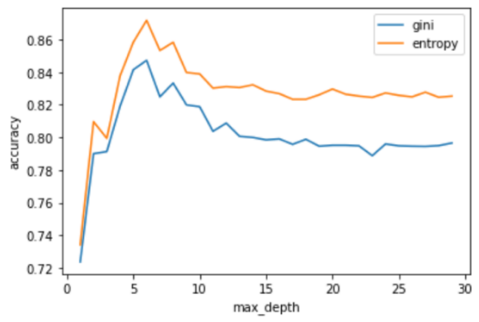
Since It has been established that rbf kernel clearly works the best for this dataset. The mean accuracies of all the 10 folds per each C-value is given above. Cross validation reveals that perhaps **C-value of 0.01 is the best value** with rbf kernel that best generalizes the model and decreases overfitting.

Decision Tree Classification On Dataset 1



- This plot shows the accuracy levels for various tree depths for both gini and entropy information criterion.
- We can see that on the whole gini information criterion has better accuracy levels than entropy.
- At approximate **maximum depth of 7 with gini criterion**, the decision tree classification has the **best accuracy of 0.91,** so we can prune the tree here.

K-Fold Cross Validation:



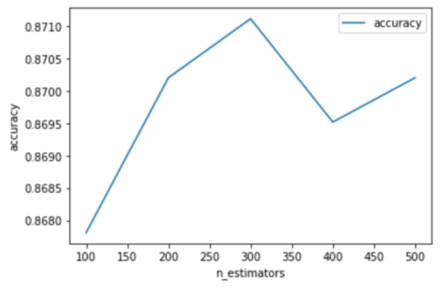
- 10-fold cross validation reveals the mean accuracy levels for various tree depths for gini and entropy-based information criterion.

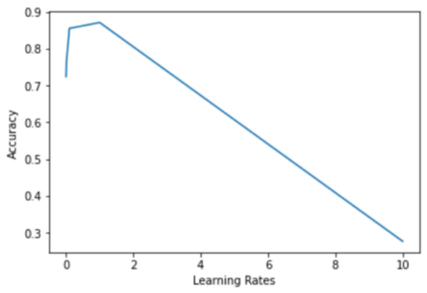
Cross validation reveals that in fact entropy is the best information criterion as opposed to gini in the random train and test split initially. Therefore, **entropy criterion with tree depth of 6 is the best parameters for this model**. This **model** has the mean **accuracy of 0.87** and **reduces overfitting**. We can see that previously we got an accuracy of 0.912 just by random chance.

Boosted Decision Tree using AdaBoost Classification:

The AdaBoost model was tested on the train and test split and the model had the test accuracy of 0.913. There is just 0.1% improvement in the accuracy level by using this model. Learning rate of 1 was used and the estimators used in the model = 100.

Using K-fold Cross Validation for finding the best parameter Values:

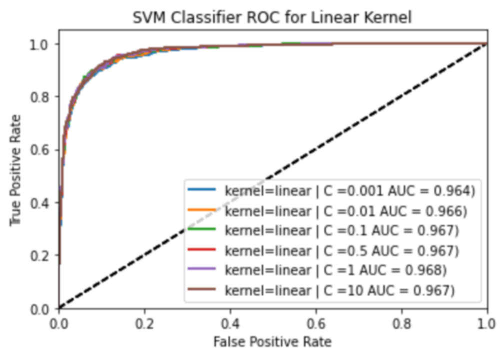




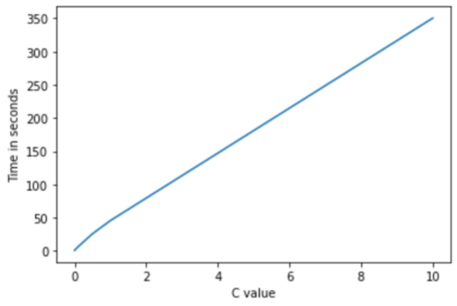
We can see from the above plots that **300 estimators** have overall **best test mean accuracy** of about 0.87 and the **best learning rate is 1** which gives best test accuracy. Therefore, with these best parameters the **average accuracy of the model would be 0.87 that reduces overfitting.**

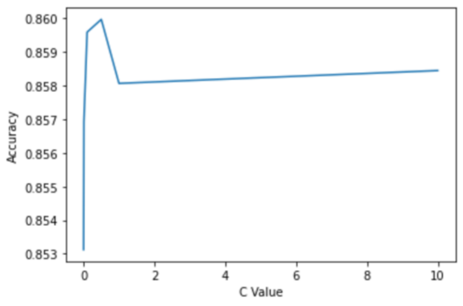
SVM Classification on Dataset 2

Linear Kernel Results:



- The plot here shows the ROC curves and the AUC for linear kernel with their respective C values.
- We can observe that **for C=1, the AUC is the highest**. Higher the AUC, better the model. Here, all the C-values have similar test accuracies.

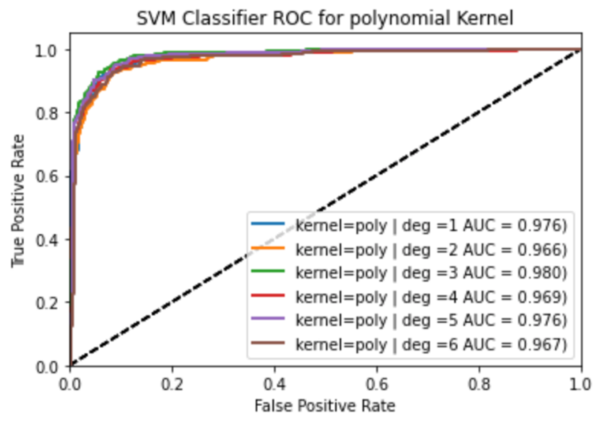




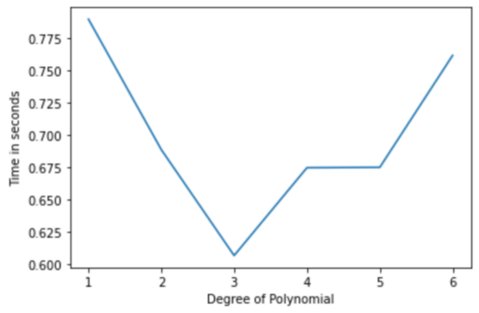
Plot 1: Describes the execution time for different C value. We can observe that as the **C value increases the execution time also increases.** However, execution time purely depends upon the configuration of the environment.

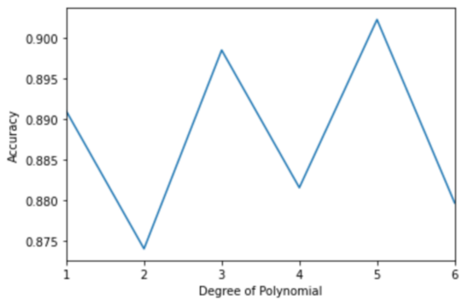
Plot 2: Describes the accuracy with respect to the C-value. We can say that with low C-values the accuracies are low, **as the C-value increases the accuracy also increases slowly**. However, the **highest test accuracy of 0.877** was achieved at **C=0.01**.

Polynomial Kernel Results:



- The plot here shows the ROC curves and the AUC for polynomial kernel with their respective degree.
- We can observe that **for degree=3, the AUC is the highest**. Higher the AUC, better the model.

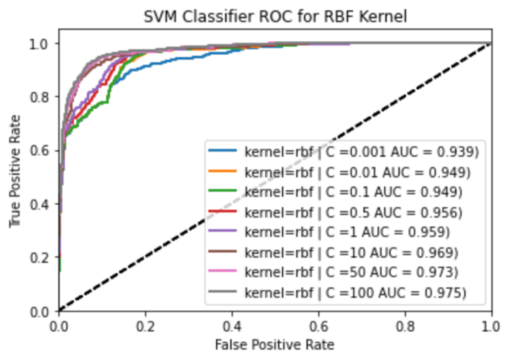




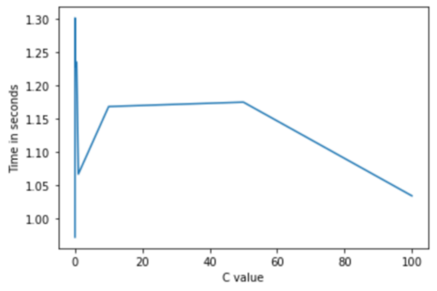
Plot 1: Describes the execution time for different degree of polynomial. We can observe that as the **degree of polynomial increases there is lot of variation in execution time.** Polynomial of degree 3 takes the minimum time.

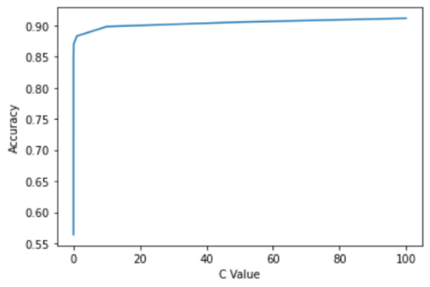
Plot 2: Describes the accuracy with respect to degree of polynomial. In general, the odd degrees have higher accuracy than the even degrees of polynomials. However, the **highest accuracy of 0.90 was achieved at degree=5**. But, in my opinion this leads to overfitting,

RBF kernel Results:



- The plot here shows the ROC curves and the AUC for rbf kernel with their respective C-value.
- We can observe that **for C=100, the AUC is the highest**. Higher the AUC, better the model. We also see that, as the **C value increase the AUC also improves**





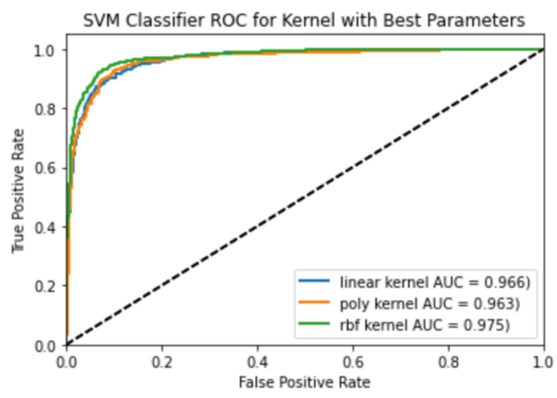
Plot 1: Describes the execution time for different C value. We can observe that as the **C value increases the execution time initially increases for lower values of C and then gradually decreases.** However, execution time purely depends upon the configuration of the environment.

Plot 2: Describes the accuracy with respect to the C-value. We can say that with low C-values the accuracies are low, **as the C-value increases the accuracy also increases in general**. However, the **highest test accuracy of 0.89** was achieved **at C=100**.

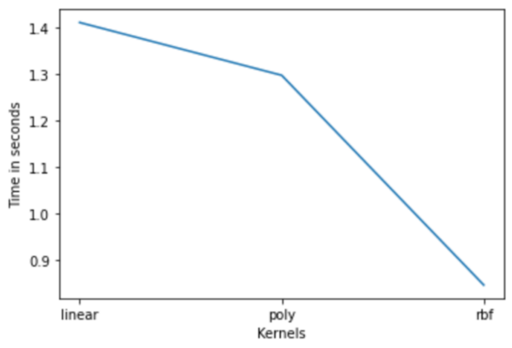
<u>Best Model Parameters</u>

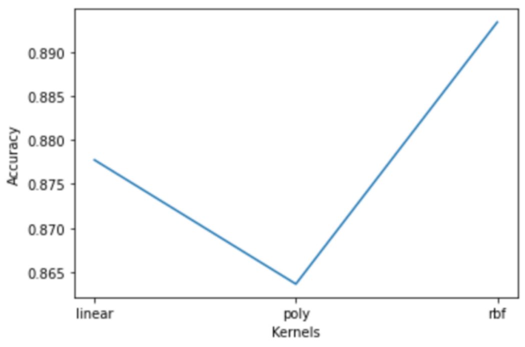
From the above information below are the best parameters for each kernel

Linear Kernel	Polynomial Kernel	RBF kernel
C= 0.01	Degree = 5	C=100



- The plot here shows the ROC curves and the AUC for linear, polynomial and rbf kernels.
- We can observe that **rbf kernel has the best AUC of 0.975.**

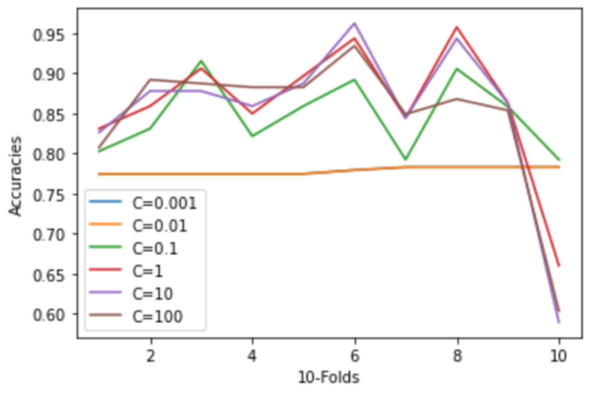




Plot 1: Describes the execution time for all kernels. RBF kernel takes the least time to execute while linear takes the highest.

Plot 2: Describes the accuracy with various kernels with best parameters. **RBF kernel results in the highest test accuracy of about 0.893 and polynomial kernel results in the least test accuracy of 0.863.**

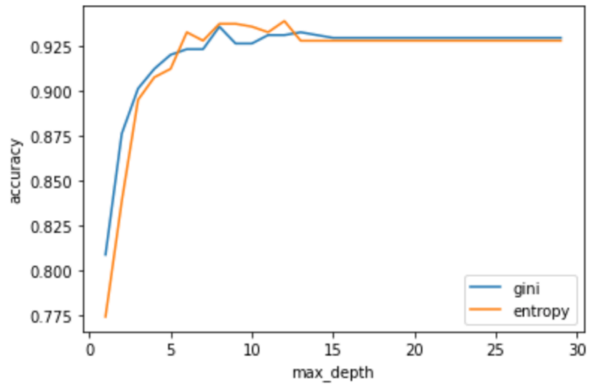
K-Fold Cross Validation:



Mean Accuracy for C-Value 0.001: 0.7784657631322526
Mean Accuracy for C-Value 0.01: 0.7784657631322526
Mean Accuracy for C-Value 0.1: 0.8471122331473115
Mean Accuracy for C-Value 1: 0.8611856674639029
Mean Accuracy for C-Value 10: 0.8531645849942422
Mean Accuracy for C-Value 100: 0.8460913278412614

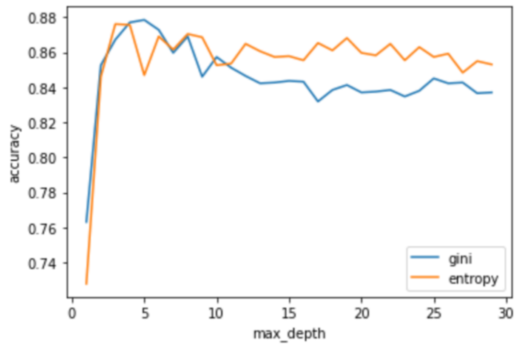
Since It has been established that rbf kernel clearly works the best for this dataset. The mean accuracies of all the 10 folds per each C-value is given above. Cross validation reveals that perhaps **C-value of 1 is the best value with rbf kernel** that best generalizes the model and **decreases overfitting**.

<u>Decision Tree Classification On Dataset 2</u>



- This plot shows the accuracy levels for various tree depths for both gini and entropy information criterion.
- We can see that until 5 depths gini has better test accuracy than entropy. From 5-12 depths entropy achieves higher test accuracy levels.
- At approximate **maximum depth of 12 with entropy criterion**, the decision tree classification has the **best accuracy of 0.938,** so we can prune the tree here.

K-Fold Cross Validation:



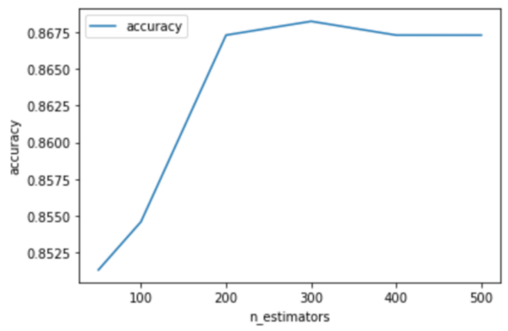
- 10-fold cross validation reveals the mean accuracy levels for various tree depths for gini and entropy-based information criterion.

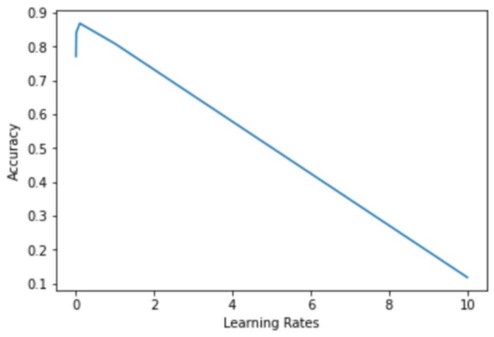
Cross validation reveals that entropy information criterion is doing a better job on the whole but, gini criterion at depths 5 has the highest test accuracy. Therefore, **gini criterion with tree depth of 5 is the best parameters for this model**. This **model** has the mean **accuracy of 0.8766** and **reduces overfitting**. We can see that previously we got an accuracy of 0.938 just by random chance.

<u>Boosted Decision Tree using AdaBoost Classification on Dataset 2:</u>

The AdaBoost model was tested on the train and test split and the model had the **test accuracy of 0.879**. There **6.45% decrease** in the accuracy level by using this model. Learning rate of 1 was used and the estimators used in the model = 100.

Using K-fold Cross Validation for finding the best parameter Values:





We can see from the above plots that **300 estimators** have overall **best mean test accuracy** of about 0.86 and the **best learning rate is 0.1** which gives best test accuracy. Therefore, with these best parameters the **average accuracy of the model with cross validation would be 0.868 that reduces overfitting.**

<u>Comparison:</u>

Test Accuracies of Algorithms with best parameters on the Datasets without Cross Validation

Algorithm	**Dataset – 1 (Test Accuracy)**	**Dataset - 2 (Test Accuracy)**
SVM	91%	89%
Decision Tree	91%	93.8%
AdaBoost	91.3%	87.9%

Accuracies of Algorithms with best parameters on the Datasets with Cross Validation

Algorithm	**Dataset – 1 (Test Accuracy)**	**Dataset – 2 (Test Accuracy)**
SVM	77%	86%
Decision Tree	87%	87%
AdaBoost	87%	86.8%

<u>Conclusion</u>

1) Both the datasets do well with RBF kernel in SVM classification with around 90% test accuracy in the random split.
2) However, decision tree seems to be a good classification model for both the datasets according to the random split that I had, with just over 90% test accuracy levels in both the datasets.
3) In dataset 1, we could see that there was 0.1% improvement in the test accuracy but there was about 6% decrease in accuracy in dataset 2 when the boosted version of the decision tree was used [Adaboost Algorithm]
4) The above results are specific to the random test and train split I had initially, however these results do not conclude that Decision tree is the best model for both the datasets. It is because, the information from the test data has been leaked into the training set while trying to find the best parameters for the models and I chose only those parameter values that gave best test accuracies in all the models. This also means that the models are overfitting.
5) We could observe that the test accuracy levels decrease when all the algorithms are run using cross validation. This is bound to happen because the best generalized model would have a good Bias and Variance tradeoff. This point has been clearly established with the experiments that I had conducted in my assignment.
6) Therefore, it is important to use cross validation for hyperparameter tuning rather than trying it on the same train and test data splits. This reduces overfitting in the model.
7) From the cross-validation results from all the algorithms, I conclude that Decision Tree and AdaBoost algorithms perform better in terms of test accuracy compared to SVM Classification. However, If I have to choose the best model it would be Decision Tree Algorithm since it is doing slightly better in classifying correctly in Dataset -2.