SEOUL BIKE DATA DEMAND

INTRODUCTION

Public bike-sharing systems have been gaining momentum only in the last decade. The main purpose other than convenience and easy-to-use service for customers is the mobility. More people are turning to healthier life styles and locations where bike riding can be easily available. There are many benefits in bike riding. Therefore, it is important to have rental bikes available to the customers (in our case, the public) to reduce their waiting time.

In this project, we implement a binary classification problem. It helps to give us the best model selected through experimentation and then evaluate the model for prediction

DATASET

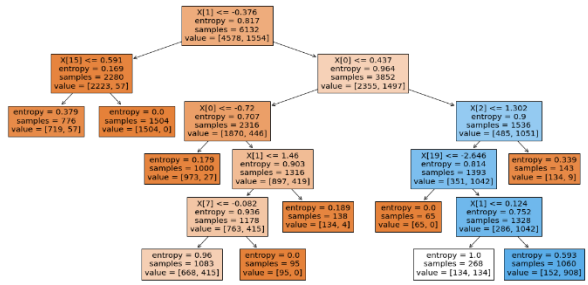
This dataset has 8760 records comprising the details of every hour each day and also 14 columns. The data contains the hourly and daily count of rental bikes. It also contains the weather information (Rainfall, Snowfall, Temperature, Humidity, Visibility etc). Rented Bike Count is the current dependent variable when checked in the dataset.

We plan to implement different algorithms and see if our model generalises well.

DATA CLASSIFICATION PROBLEM

The dependent variable is Rented Bike Count which is a numerical variable. We will change our dependent variable to a binary classification problem by thresholding. Current threshold taken for our analysis is 75^{th} percentile. A good reason for creating binary predictors from numerical predictors is to overcome the problem of linearity. According to linear regression, we assume that X and Y to be in linear relation. If we can't find an appropriate equation to represent relation between X and Y variables , then creating binary predictors might be a way of obtaining some predictors

What interesting is the fact that earlier, we could only get the variation of features that have an impact to the dependent variable, Rented Bike Count. Now, we get a solid yes/no answer to the point where features/independent features give an answer to the dependent variable.



Also, for example, in the case of decision tree algorithm, we can find the most important features that have a huge impact on the dependent variable. Variables Temperature is the one that topped the tree followed by other factors like Hour, Humidity, Summer and Non-Functional Day.

ALGORITHMS

1.SUPPORT VECTOR MACHINE (SVM)

SVM is type of supervised machine learning classification algorithm which separates a dataset into two classes using a line called the maximal margin hyperlane.

We have divided the data into training and testing sets. Scikit-Learn contains the svm library, which contains built-in classes for different SVM algorithms. Since we are going to perform a classification task, we will use the support vector classifier class, in the Scikit-Learn's svm library. This class takes one parameter, which is the kernel type. This is very important. In the case of a simple SVM we simply set this parameter as "linear" since simple SVMs can only classify linearly separable data

There is obviously a trade-off between increasing distance of boundary classes and maximising the no of points correctly classified and it is controlled by C which adds a penalty for each misclassified data point.

If C is small, the penalty for misclassified points is low so a decision boundary with a large margin is chosen at the expense of a greater number of misclassification.

If C is large, SVM tries to minimize the number of misclassified examples due to the high penalty which results in a decision boundary with a smaller margin. The penalty is not the same for all misclassified examples. It is directly proportional to the distance to the decision boundary.

In below figure, we have compared the accuracy score of different SVM models based on the C value and could see that default and polynomial kernel showed the best accuracy score.

	ACCURACY SCORE			
C	**rbf**	**linear**	**polynomial**	**sigmoid**
1	0.9075	0.8668	0.9022	0.7671
100	**0.9243**	0.8657	**0.9212**	0.7508
1000	0.9189	0.8668	0.9186	0.7511

In the below plot, learning curves (Score Vs Training Samples) in regard to the below parameters have been plotted for few kernels. We can see that for rbf kernel, the training score is way different from the cv score, this is a case of overfitting the model. For sigmoid kernel, the performance is very poor compared to other kernels. Linear has shown a decent score but comapred to the non-linear kernels, rbf and kernel, the performance is poor.

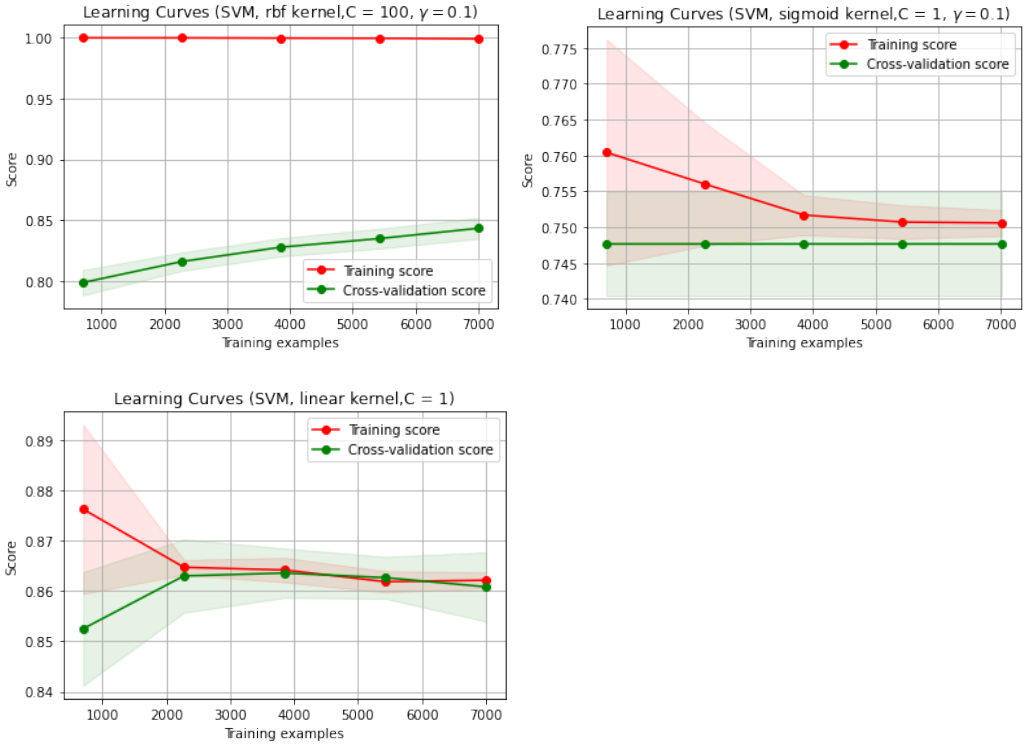
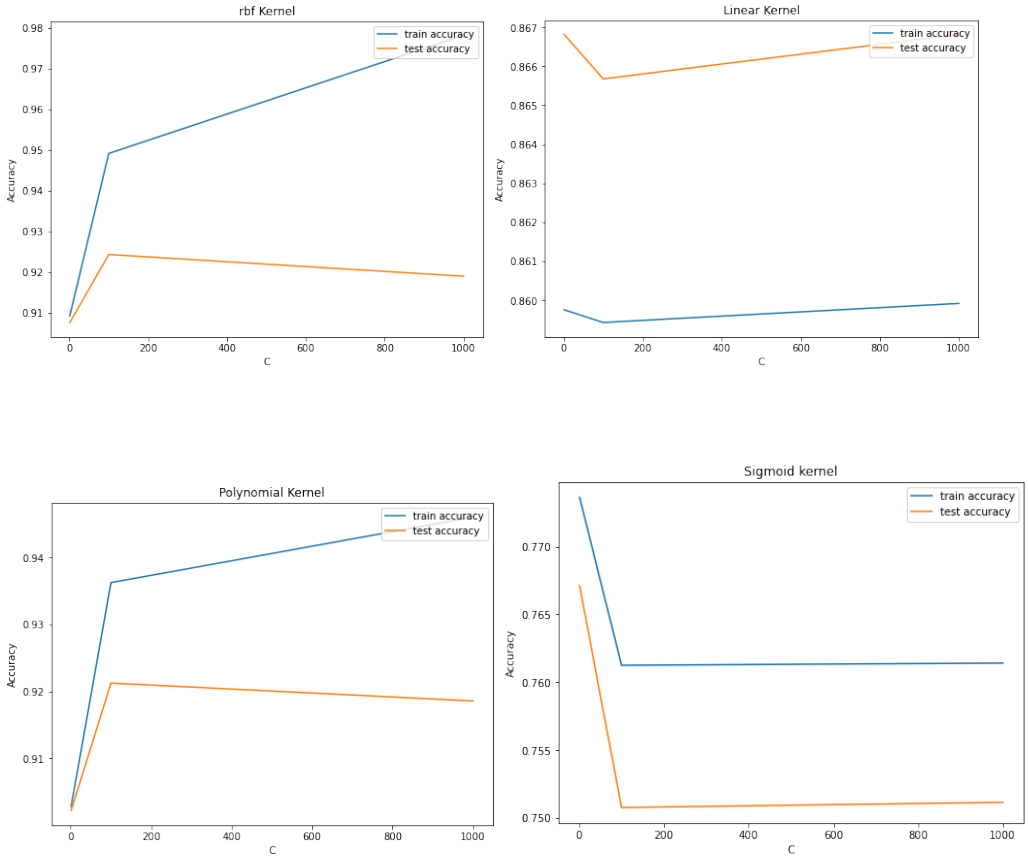


Fig 3.1 Score Vs Training Samples

The plots for training and testing score with respect to C parameter for different kernels. In Fig 3.2, the hyperparameter C is plotted for different training and testing accuracy. The accuracy score is shown least in sigmoid kernel while non-linear kernels have shown the best scores. The test score of linear kernel has mimicked the training score but the score is o=not good when compared to the non-linear kernels

Fig 3.2 Accuracy Vs Graph – Train and Test Data



2.DECISION TREE

The Decision tree algorithm is a simple yet efficient supervised learning algorithm wherein the data points are continuously split according to certain parameters. The algorithm works by dividing the entire dataset into a tree-like structure supported by some rules and conditions. Then it gives predictions based on those conditions. It empowers predictive modeling with higher accuracy, better stability and provides ease of interpretation.

While using decision trees, there is a high chance of overfitting the model as the model gets complex when there are more number of splits and also greater depth. As to this, the variance is also increased which reduces the training error but the prediction on new data points (our test set) is not comparable to that to training performance. Pruning is therefore, considered to avoid this.

ACCURACY SCORE	
Training	Test
1	0.9216

For our dataset, we could see that our data was overfitted for an Unpruned Decision Tree (Fig 4.1) The training score and test score are comparitively different from one another which shows that this model is overfitted.

Attribute Selection Measure

Another primary challenge in the Decision Tree implementation is to identify the attributes which we consider as the root node and each level. This process is known as the attributes selection. There are different attributes selection measure to identify the attribute which can be considered as the root node at each level. *Information Gain* uses the concept of measuring the impurity in the given dataset, that is, entropy while *GINI Index* is used to create split points. The reason to use these metrics is that Gini is intended for continuous attributes and Entropy is for attributes that occur in classes. Gini is to minimize misclassification while Entropy is for exploratory analysis.

In out dataset, we check the accuracy scores for Entropy as well as GINI Index. The accuracy scores of both the training and testing data is compared and we could see from Fig 4.2, the GINI Index gives a better performance when compared to Entropy. Also, Entropy is more complex since it uses logarithms and therefore, the GINI Index is more faster.

Fig 4.2 Accuracy Score – Attribution Selectio Measure

ACCURACY SCORE		
Measure	Training	Test
GINI	0.9223	0.914
ENTROPY	0.878	0.8888

We can see that the training-set score and test-set score is same as above. The training-set accuracy score is 0.9223 while the test-set accuracy to be 0.914. These two values are quite comparable. So, there is no sign of overfitting.

Now, based on the above analysis we can conclude that our classification model accuracy for GINI Index is very good. Model will do a good job in the prediction. We will therfore prefer this metric over the Entropy measure

ACCURACY SCORE		
Method	**Training**	**Test**
Unpruned	1	0.9216
Post Pruning	**0.9274**	**0.9334**

Post Pruning/Backward Pruning

Decision Tree pruning ensures trimming down a full tree to reduce the complexity and variance of the model. It makes the decision tree versatile enough to adapt any kind of new data fed to it, thereby fixing the problem of overfitting. It reduces the size of the decision tree which might slightly increase the training error but drastically decrease the testing error. For this, it become more adaptable.

Backward pruning is where the decision tree is generated first and then the non-significant branches are removed. As in Fig 4.1, our model shows an overfitted model , we prune the branches based on cost_complexity_pruning technique. The parameter used for this technique is ccp_alpha.This gives the minimum value (leaf) of decision tree.

In Fig 4.3, an Accuracy Vs alpha graph has been plotted to check the value of alpha that shows the maximum accuracy. We will choose ccp_alpha as 0.002 as both the training and test accuracy is optimum to show that the model will generalize well for all new data

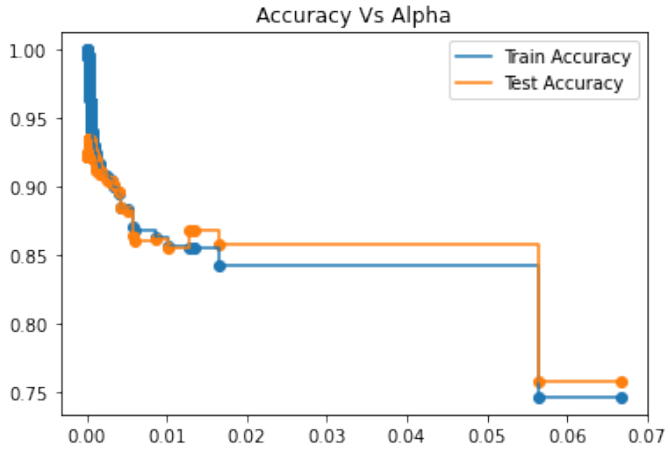


Fig 4.3 Accuracy Vs Alpha

CLASSIFICATION METRICS

Confusion Matrix

Confusion matrix helps to understand the performance of a classification algorithm. It helps to understand the type of errors occurred and also summary of correct and incorrect predictions.

We compared the metrics for the 2 best SVM models selected and Decision tree algorithms. In Fig 5.1, we can see True Positives show high for SVM-Polynomial kernel. Post Pruning shows the bext among the Decision model algorithms in terms of capturing the Truoe Positives and True Negatives.

CONFUSION MATRIX METRICS				
Model	**True Positives**	**True Negatives**	**False Positives**	**False Negatives**
Decision Tree - GINI	1847	555	145	81
Decision Tree - Entropy	1902	434	90	202
Decision Tree - Post Pruning	1892	561	100	75
SVM - Default at C = 100.0	1904	525	88	111
SVM - Polynomial at C = 100.0	1907	514	85	122

Fig 5.1 Confusion matrix metrics

In Fig 5.2 shows the extended version of confusion matrix. They help understand to analyse the model based on performance. In our analysis, accuracy shows highest for default SVM model . However, Sensitivity is the highest for the pruned decision tree.

CLASSIFICATION METRICS					
Model	**Accuracy**	**Error**	**Precision**	**Sensitivity**	**Specificity**
Decision Tree - GINI	0.914	0.086	0.9272	0.958	0.7929
Decision Tree - Entropy	0.8889	0.1111	0.9548	0.904	0.8282
Decision Tree - Post Pruning	0.9334	0.0666	0.9498	0.9619	0.8487
SVM - Default at C = 100.0	**0.9243**	0.0757	0.9558	0.9449	0.8564
SVM - Polynomial at C = 100.0	0.9212	0.0788	0.9573	0.9399	0.8581

Fig 5.2 Extended Classification Metrics

ROC

The ROC is a two-dimensional plot describing the performance of the classifier system using the contingency table and the area under the curve (AUC). The contingency table consists of four conditions based on the model predictions, and the occurrences or non-occurrences of a specific event are categorized as true positive (TP), true negative (TN), false positive (FP), and false negative (FN). The area under the curve (AUC) is used to evaluate the ability of the classifier system to discriminate the actual condition of rented bike counts.

ROC CHARACTERISTICS			
Model	**TPR**	**FPR**	**AUC**
Decision Tree - GINI	0.958	0.2071	0.8999
Decision Tree - Entropy	0.904	0.1718	0.8186
Decision Tree - Post Pruning	0.9619	0.1513	**0.9159**
SVM - Default at C = 100.0	0.9449	0.1436	0.8906
SVM - Polynomial at C = 100.0	0.9399	0.1419	0.8828

Fig 5.3 ROC

A perfect classifier should have high true positive rate and low value of false positive rate. In our analysis, pruned decision tree satisfies the condition(Fig 5.3 and fig 5.4)

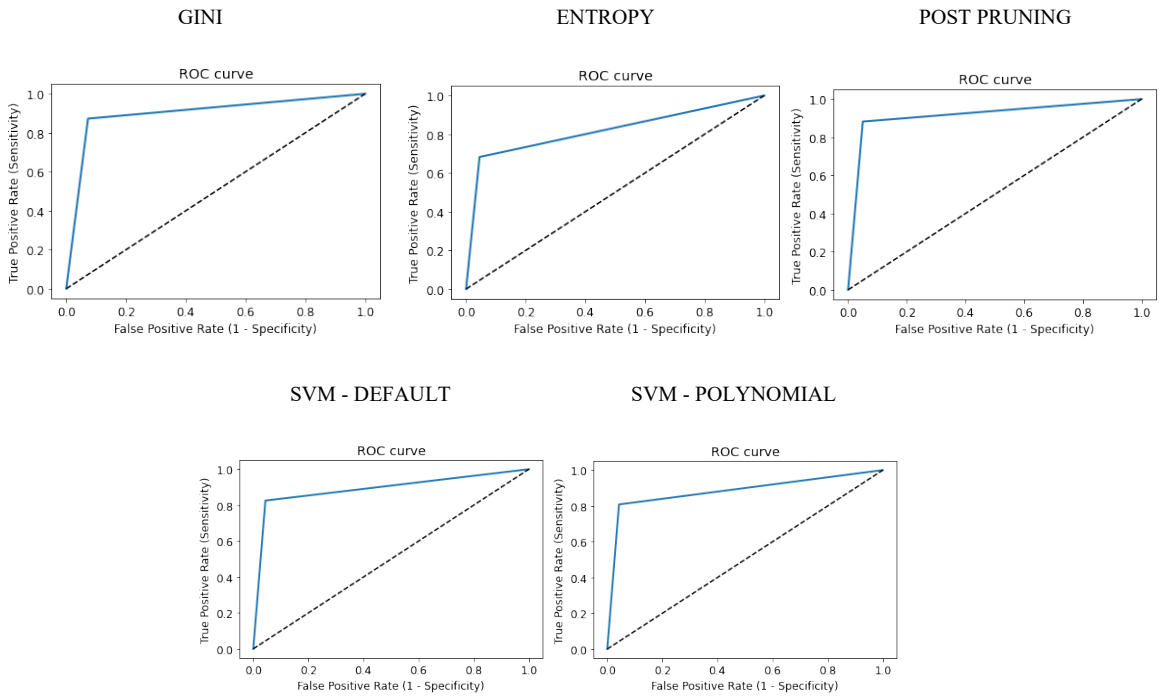


Fig 5.4 Receiver Operating characteristics

CROSS VALIDATION

Cross-validation is a statistical method used to estimate the performance (or accuracy) of machine learning models. It is used to protect against overfitting in a predictive model, particularly in a case where the amount of data may be limited. The number of folds is fixed and analysis on each fold is conducted to get the average overall error. It also helps us better use our data, and it gives us much more information about our algorithm performance

Model	Average Score	Standard Deviation
Decision Tree - Unpruned	0.8534	0.0866
Decision Tree - GINI	0.8479	0.1046
Decision Tree - Entropy	0.8739	0.0813
Decision Tree - Post Pruned	0.8667	0.0996
SVM rbf	0.8286	0.1218
SVM Polynomial	0.8207	0.1255
SVM linear	0.7719	0.1386
SVM Sigmoid	0.6229	0.077

K-Fold Cross Validation has been used for our dataset.The average score and standard deviation for 20 folds have been taken and plotted in Fig 6.1

Fig 6.1 Average Score for CV (k-Fold)

The best hyperparameters have also been checked using cross validation through the GridSearchCV package. This package helps to define the best paramaters, (for SVM: C and gamma; Decision Tree :max_split,max_leaf_nodes,max_samples_split) and as shown in Fig 6.2, validation of these models have been conducted to see if there is any improvement in our score.

kernel	Before Cross Validation	Post Cross Validation
rbf	0.9075	
rbf at C = 100.0	**0.9243**	**0.9189 at C = 100 and gamma= 0.1,**
rbf at C = 1000.0	0.9189	**0.9242 at C = 100 and gamma default**
linear	**0.8668**	
linear at C = 100.0	0.8657	
linear at C = 1000.0	0.8668	**0.8668 at C = 1.0**
polynomial	0.9022	
polynomial at C = 100.0	**0.9212**	
polynomial at C = 1000.0	0.9186	**0.9212 at C = 100**
sigmoid	**0.7671**	
sigmoid at C = 100.0	0.7508	
sigmoid at C = 1000.0	0.7511	**0.7671 at C = 1**

Fig 6.2 Cross Validation Comparison – SVM and Decision Tree

Model	TEST SCORE - Before CV	TEST SCORE - Before CV at max_depth = 6	Best Splits	After Cross Validation
GINI	Overfitted model	0.914	'max_depth': 6, 'max_leaf_nodes': 10, 'min_samples_split': 2	0.8711
Entropy	Overfitted model	0.8888	'max_depth': 7, 'max_leaf_nodes': 10, 'min_samples_split': 2	0.8708
Unpruned	Overfitted model	0.9216(Overfitted model)	'max_depth': 6, 'max_leaf_nodes': 10, 'min_samples_split': 2	0.8711
Post Pruning	0.9334	0.9334	'max_depth': 7, 'max_leaf_nodes': 10, 'min_samples_split': 2	0.8708

On using cross validation, there has been sufficient increase in the model compared to the base model referenced above. The k-Fold helped to identify the best hyperparameters for each kernel

for SVM and provided the best model's hyperparameters, as well as the test accuracy for Decision tree. This model fitting is computationally heavy as it has to check the best possible hyperparameters from the listed one.

CONCLUSION

For the binary classification problem, model has been generalised well based on the two algorithms that has been introduced, SVM and Decision Tree.

The SVM is supervised learning method and different models in variation to hyperparameter C. They help to find the balance between bias and variance. From the analysis for this dataset, one thing noted that ***non-linear kernel functions*** gave more predictive accuracy than the linear which shows that this model in reality can be noisy and is not linearly separable. We create a decision boundary based on ***C parameter*** as this is directly propertional to the direction boundary and will handle misclassifications.

Various kernel functions have been used for the data set and the accuracy score for both linear and non-linear kernal functions have been noted. Polynomial kernel has given an accuracy score of 0.9212 and rbf kernel , an accuracy of 0.9189 which provides a much better model than the linear model of 0.8668. It can be therefore be said, considering the dataset, that for a normalised training data and for a kernel which projects to high dimension and searched for a linear separation, ***polynomial*** and ***rbf*** kernels perform well respectively. Linear kernels are preferred basically for text classification problems.

In decision tree, model accuracy has been validated post pruning and we could see that pruning plays an important role in fitting models using the Decision Tree algorithm and post pruning. ***Post pruning*** has shown the maximum accuracy score of 0.9334 which shows that it is the more efficient one among the models. Selecting the correct value of ***cpp_alpha*** is the key factor in the post pruning process. In addition, decision tree models with GINI Index and Entropy has also been validated with their accuracy model. From our analysis, it is understood that Gini is intended for continuous attributes, and Entropy for attributes that occur in classes. Computationally, entropy is more complex since it makes use of logarithms and consequently, the calculation of the Gini Index will be faster. 0.8999 is the accuracy score for ***GINI Index*** and

0.8889 for ***entropy*** which is almost similar. It is therefore, preferable to choose GINI over entropy criterion as to invest time.

Cross Validation accuracy has been validated in both the algorithms and can be used as a performance metric to compare the efficiency of different models. The CV used here is ***k-Fold*** which generally produces less biased models as every point from the original dataset will appead in both training and testing set. This method is optimal if limited data is used for the analysis. However, as expected, this process might be time-consuming because the algorithm has to rerun k times from scratch. This also means that it takes k-1 times more computation than the holdout method.

We could see that on cross validation, we will able to get the optimum hyperparameters for both SVM and Decision Tree. They are usually fixed before the actual training process begins. These parameters express important properties of the model such as its complexity or how fast it should learn. Therefore, hyperparameter tuning helps to improve the performance of the models.

On comparing both models, ***Decision Tree algorithm*** performance is better than the algorithm Support Vector Machine. Several factors made the Decision Tree algorithm is better than SVM. One of them is the ability to define and classify each attribute to each class simply. The resulting computing time by Decision Tree faster than SVM. Decision Tree is recommended to build or add a parameter. Measurement of the performance of a data mining algorithms can be based on several criteria among others the accuracy, computing speed, robustness, scalability and interpretability. Decision Tree uses a criterion that is based on the accuracy.

We can therefore, have the ***Post Pruning*** method with shows the highest accuracy as the ***best*** model. The model with ***GINI Index*** can also be considered as a model that generalises well for the dataset.This has given a great interpretation of the model results. Based on predictions and based on goodness of fit on the current data helps us to understand that this model can predict new data and can also describes the relations in the current data. Moreover, the confusion metrics along with the ROC curve is a great advantage on validating the best model.

Additionally, best parameter for ***gamma*** hyperparameter can be validated for non-linear kernels.However, compurationally, this is very expensive and should have a good processor.SVM can be conducted with ***L1 regularization*** can be conducted. Another aspect that can be done is conduct the evaluation of the model through methods like ***Log Loss*** and ***Gain and Lift Charts***. Log Loss is very effective to see how likely the model thinks the observed set of outcomes was.The Gain and Lift chart evaluate the model on portion sof the whole population. Also, performance of the model can also be monitored in addition to other criterias. Accuracy can be improved by using techniques like ***bagging*** and ***boosting*** technique.