Prepared by: Jyotsna Nagaprasad Ambekar Date: 07/08/2022

BUAN 6341 Applied Machine Learning

ASSIGNMENT NO 2

Seoul Bike Data

Executive Summary

- **Classification problem with data classified based on median.**
- **Features without high correlation are used to build SVM with Linear, RBF, Polynomial, Sigmoid kernels models. Decision trees algorithm is also implemented**.
- **Implementation of K-fold Cross Validation and GridSearchCV show best kernel for Seoul Bike data is RBF kernel.**
- **Prediction accuracy can be further improved by implementing XGBoost or Random Forest Algorithm that is apt when data has multiple discrete features.**

Context

Currently Rental bikes are introduced in many urban cities for the enhancement of mobility comfort. The goal is to help commuting in the cosmopolitan cities and reduce the pollution amounts caused by cars, individual motorcycles etc. and create green cities altogether. It also is a healthiest way for travelling to work or school. It is important to make rental bike available and accessible to the public at the right time lessening the waiting time. Eventually, providing the city with stable supply of rental bikes becomes a major concern as it can be used as alternative to the commuters in cities. The crucial part is the prediction of bike count required at each hour for the stable supply of rental bikes.

Introduction

In this project, the objectives were to implement Support Vector Machine & Decision Tree algorithms to predict the Rented Bikes count on a given day with given attributes. This report details various experiments conducted using the dataset to understand the effect of using different kernels in SVM and pruning the tree in decision tree algorithm

About the Data

The dataset consists of 14 features and 8760 records with no missing values. The data contains information regarding weather and how much solar radiation is observed on the day, seasons, holiday, functional hours etc. The target variable is the rented bike count per hour.

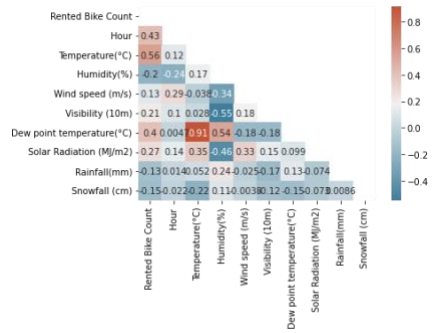
Attribute information

Date : year-month-day
Rented Bike count - Count of bikes rented at each hour
Hour - Hour of the day
Temperature-Temperature in Celsius
Humidity - %
Windspeed - m/s
Visibility - 10m
Dew point temperature - Celsius
Solar radiation - MJ/m2
Rainfall - mm
Snowfall - cm
Seasons - Winter, Spring, Summer, Autumn
Holiday - Holiday/No holiday
Functional Day - NoFunc(Non Functional Hours), Fun(Functional hours)

The independent variables used for training the model are Hour, Temperature(°C), Humidity(%), Wind speed(m/s), Visibility (10m), Solar radiation (MJ/m2), Rainfall(mm), Snowfall, Seasons & Holidays one hot encoded columns. The season column encoded as 0 is Autumn, 1 is spring, 2 is summer, 3 is winter. The Holiday column encoded as 4 is Holiday and 5 is No-Holiday

Data Preprocessing

It is the technique in which data is processed so that it is appropriate to feed the model. Preprocessing steps applied were One hot encoding to encode categorical variables like Seasons & Holiday columns, Splitting the dataset into training and test sets, Feature scaling for standardizing data. Firstly, data exploration was done by understanding the types of data present and description of data. The Date column which was initially string was converted to DateTime format.



The temperature and dew point temperature have correlation value of 0.91. Due to this high correlation, multi collinearity arises if both the features are used to develop the model. Hence dew point temperature is dropped from the independent variables.

Experimentation

Task – 1: The dataset was binary classified based on the median value. There are almost equal number of rows are present in each class. The median was chosen such that there is no imbalance in the dataset. The categorical variables, Seasons & Holiday were OneHotEncoded. Then the dataset was split into training and test sets and feature scaling was done.

Task – 2: The sklearn library in python has SVM module under which we have SVC class and whose attributes can be changed to implement different kernels under SVM. Kernel is a function which transforms input data such that non-linear decision surface can differentiate the data points into different classes in higher dimensional space. Different kernels used were:

1. Linear Kernel – It is one dimensional and is the most basic kernel in SVM. It is used to differentiate linear datasets. As a rule of thumb, linear kernel is always applied on the data to make sure the presence of linearity in the dataset. Equation representing linear kernel is:

$$K(x_i, x_j) = x_i . x_j + c$$

2. Gaussian RBF Kernel – RBF is Radial Basis Function. This is used when there is very less knowledge of the dataset. This can be represented as:

$$K(x_i, x_j) = \exp(-\gamma | |x_i - x_j| |)^2$$

The SVC class has The regularization parameter, C tells the model how much it needs to avoid misclassification. If C is higher, model chooses smaller margin hyperplane and vice-versa. This is one of the parameters that needs to be tuned to make sure the data isn't overfitted. Another parameter which needs to be tuned is Gamma. This shows how far the influence of single training data point reaches. High gamma will look at close data points to decide the hyperplane and vice-versa.

3. Polynomial Kernel – This kernel is used when the dataset is assumed to have non-linearity and certain degree polynomial classifies dataset correctly. Equation is:

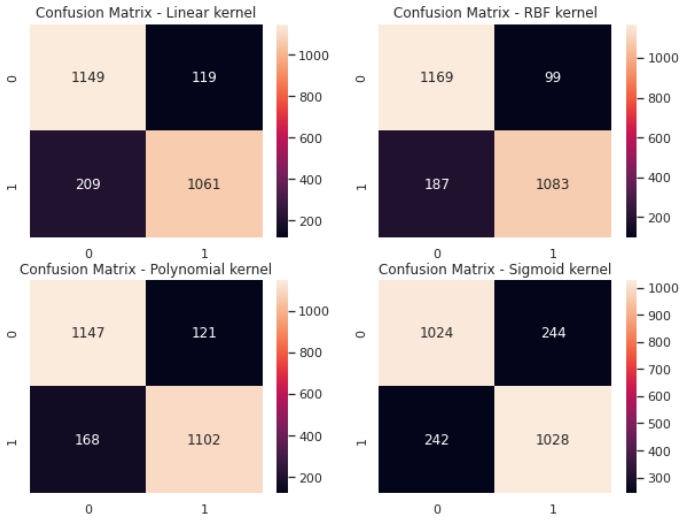
$K(xi,xj) = (xi.xj)^d$ (Homogenous Polynomial kernel function, degree d)
$K(xi,xj) = (xi.xj + c)^d$ (Inhomogeneous Polynomial kernel function)

The degree of the polynomial equation used for this case is 4.

4. Sigmoid Kernel – Equation used to represent sigmoid kernel is:

$K(x_i,x_j) = \tanh(\alpha x^a y + c)$

The Confusion Matrices & Accuracy table generated by implementing various kernels on the training dataset -



Kernel	Accuracy
Linear	87.08%
RBF	88.73%
Polynomial	88.61%
Sigmoid	80.85%

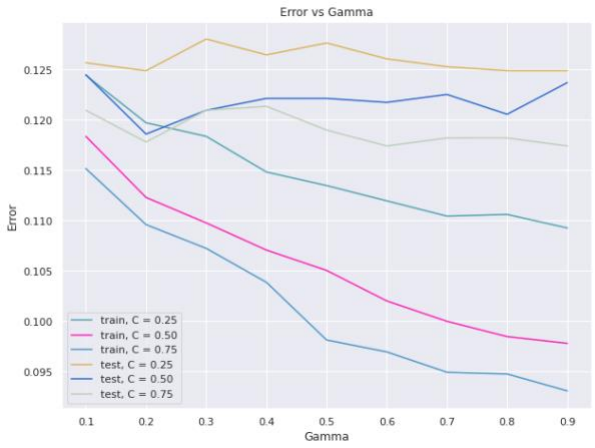
K-Fold Cross Validation for SVM

K-fold cross validation is applied on datasets to remove the plausible overfitting during training due to which we observe high accuracies on training set but very less on test set. So the dataset is divided into k folds and one of the folds is held out during training. The kth fold will be unseen data and as this process is repeated on all the k folds, the mean of all the accuracies obtained is considered along with its distribution. The low standard deviation suggests all the folds have similar accuracies(not obtained by chance) and thus overfitting is avoided. The following table shows the results for different kernels –

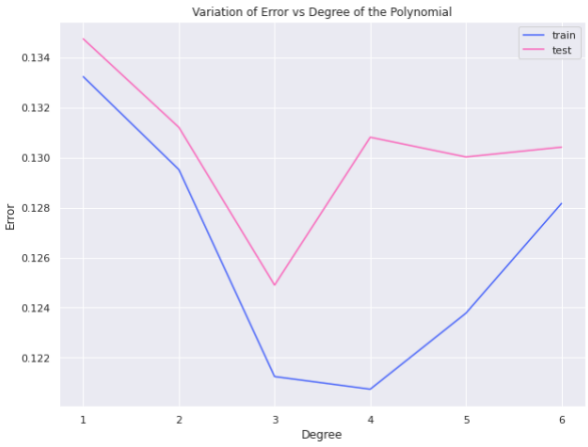
Kernel	Accuracy	Standard Deviation
Linear	86.73%	1.39317
RBF	88.46%	1.22599
Polynomial	87.93%	1.25788
Sigmoid	79.81%	1.89442

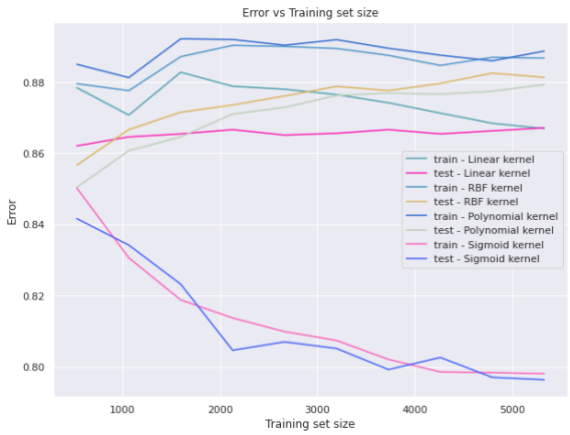
GridSearchCV for SVM

It is a process of exhaustive search with different parameters given as input for the SVM classifiers. This process is done using all the above kernels and hyper parameters.



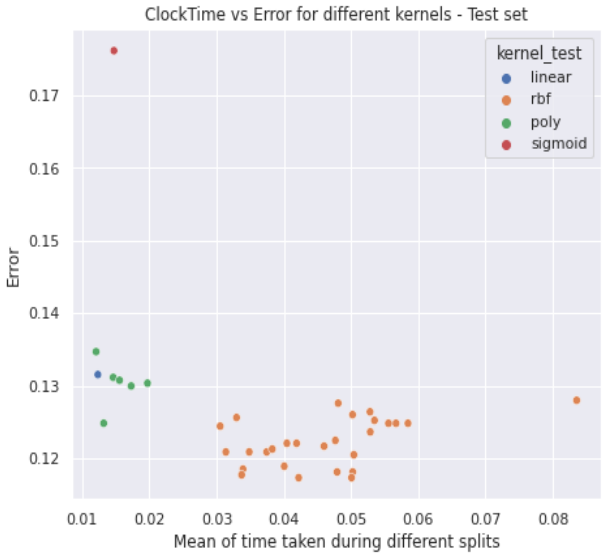
The results are saved in a dataframe. The best parameters for the training set are C = 0.75, gamma = 0.9 with RBF kernel. When the C value is high, the model chose smaller margin for the hyperplane based on the training data and best accuracy is found when we have higher C. The higher gamma means that the model is looking at nearer datapoints to control the decision boundary. Since scoring attribute of GridSearchCV is given input as Accuracy, Accuracy of the model is used to rank the best model. The best accuracy obtained when these parameters are input is 90.70%. The same parameters when applied on test set give best parameters which are C = 0.75, gamma = 0.6 with RBF kernel. With different values of C and gamma, the learning curves are drawn corresponding to the errors. The biggest variation in error is seen when C = 0.75 for the training and test sets. The least error is found when C = 0.75 and gamma = 0.9 for training set. It is observed from the graph that as gamma increases the error decreases for training set, almost similar trend is observed on the test set. In the case of polynomial kernel, degree of the polynomial is used as a parameter and is plotted against the observed errors. The graph shows that when the degree of the polynomial is 3, best accuracy is observed on training & test sets. Though when degree is 4, the error is least for training set, the test set has high error on the same degree.

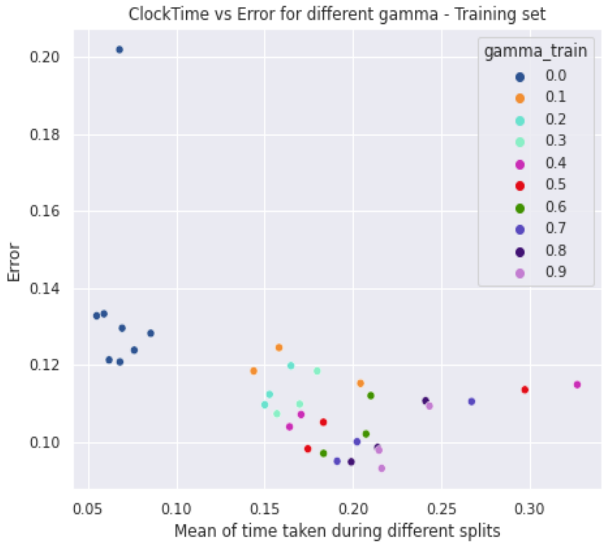


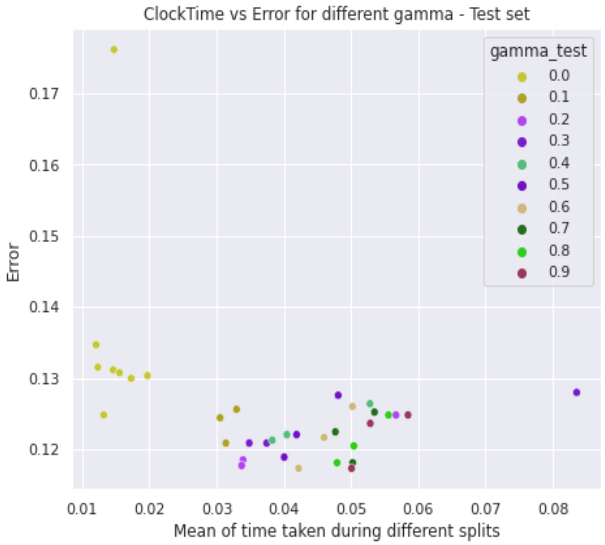


The variation in training set size also varies the error observed. The training set size was varied between 0.1 to 1 times the size of training set, and the error is drawn for different kernels the model was developed for. The sizes of training set vary from 532 to 5329 rows and the rest rows are used for testing the model. Most decrease in errors is found in sigmoid kernel when the training set size increases. The variation in other kernels is shown in the graph. There is slight increase in error as training set size increases in few kernels after certain size.

Another kind of learning curve was plotted showing the variation on time taken for different splits for different gamma values. It shows that in different kernels, the error is in the same range. When similar plot is plotted showing different kernels, sigmoid kernel has highest error when compared to all other kernels though time taken is less. The RBF kernel data points show that time taken is highest in this case with lower errors. This is because RBF uses normal curves around the data points such that the non-linearity of the data points is nicely fit. Hence RBF kernel is best fit for this dataset.

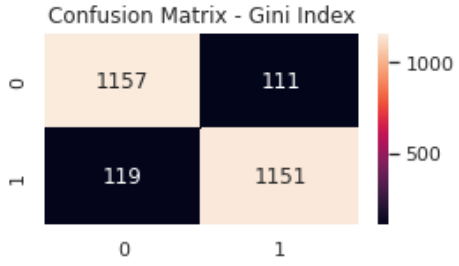




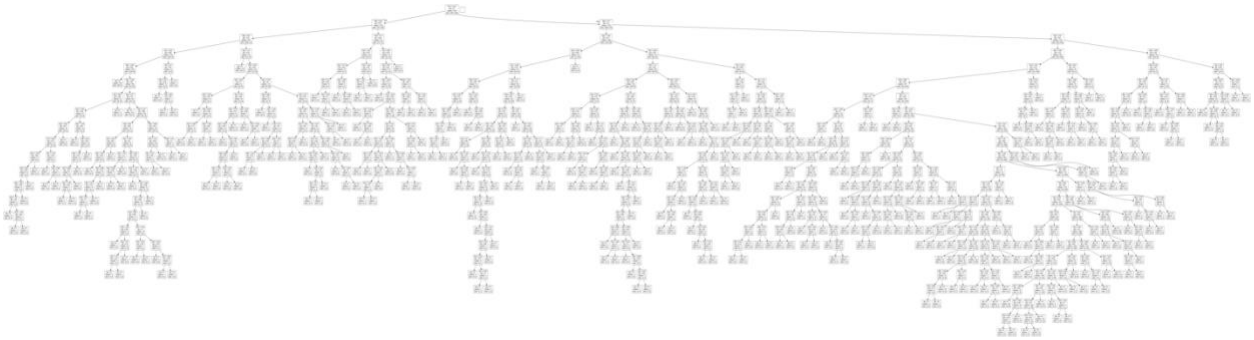


GridSearchCV is sometimes a bad idea to implement because it takes lot of time, and it is more likely to overfit the data. Instead, we can use Halving GridSearchCV which is much faster than the regular one. Halving GridSearchCV is like a competition among all candidates (parameter combinations) In the first iteration, it trains all candidates on small proportion of data. In the next iteration, only candidates which performed best are chosen and they will be given more training samples to compete. The speed of the process can be controlled by attributes – factor & min_samples.

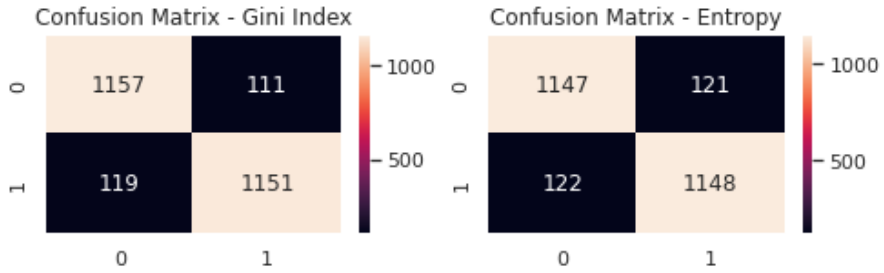
Task – 3: The classification problem was fit using Decision Tree Classifier class of sklearn library. The decision tree includes root node, branches, and leaf nodes. Each internal node denotes an 'if-else' clause, and if the condition on the attribute is met, the data points are on the left branch otherwise on the right. When the node can't be divided any further, such node is called leaf node or terminal node. Then it is said to be a fully grown tree where there can be no further division of data into classes. The tree is let to fully grow with default criterion, Gini index, and the accuracy found on completely grown tree is 90.93% with confusion matrix –



The fully grown tree is depicted as –

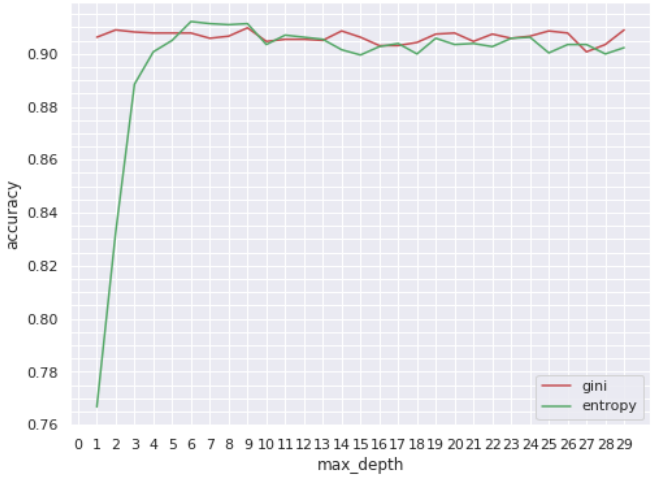


The metrics to calculate which attribute gives best purity of nodes are Gini Index & Information gain. Entropy represents the impurity of the dataset. Information gain is the decrease in entropy. Information gain computes different between entropy before and after split and the attribute with which highest information gain is obtained, is used at the node for splitting. On the other hand, Gini index works well with binary splits. Higher the value of Gini index or information gain, higher is the homogeinity. Accuracy score when Gini index is taken as criterion is 90.94%, and when entropy is criterion, 90.43%. The confusion matrix when each criterion is applied –



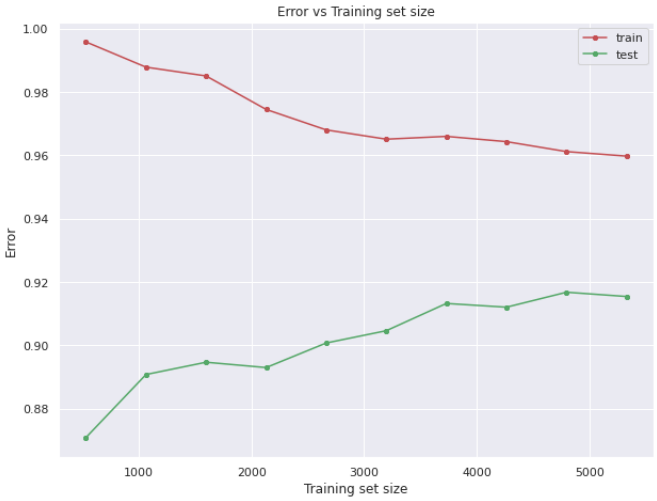
13 data points are wrongly predicted by the model with ‘entropy’ as criterion compared to ‘gini index’.

The graph shows the depth of the tree should be pruned back to 9 steps according to gini index. The max Gini index is found at depth 9. The accuracy of the decision tree is found to be 91.68% when the fully grown tree is pruned back to tree with depth 9. Applying k-fold cross validation on the classifier, with 10 folds, the accuracy is found to be 90.56% with standard deviation 1.70% By applying cross validation, it was made sure that the 91.68% accuracy which was obtained was not by chance and it is in limits of cross



validation mean accuracy +/- its standard deviation. Hence we can use the pruned tree with depth 9 as our best tree for this model with metric as Gini Index because it works well with the binary classification models and also it takes less time for the computation because it doesn't have any log functions involved as entropy.

	Gini Index	Entropy	Max depth = 9	Cross validation
Accuracy	90.94	90.43	91.68	90.56



The learning curve was drawn which depicts the errors for different training set sizes. The training set size ranges from 532 to 5329 rows. The training set has error decreasing as its size increases and for the test set, error increases. This shows that as we increase training set size, after certain extent the change in error is very less. Thus, there is no need to add more data because it increases variance in the model because it overfits the data.

Conclusion

The classification problem had accuracy of 90.66% on the test set with the best parameters obtained by GridSearchCV when fitted on training set. With the best parameters of decision tree, the accuracy obtained was 91.68%. This suggests that Decision tree though not perfect works better than kernel SVM models. Kernel SVM uses kernel trick such that non-linearity of the data is nicely fit whereas decision tree algorithm is based on which criterion to use to achieve best possible homogeneity and what is the least possible depth the tree is to be grown to achieve maximum accuracy. Decision trees involve ID3 algorithm which is termed as 'greedy' selection of the best split point from the dataset at each step. Pruning, thus helps in reducing the overfitting which in turn results in reduction of variance. Cross validation certainly reduced overfitting of the model and the model's best accuracy was in the distribution of the accuracy obtained from Cross Validation.

Better results can be brought by implementing Random Forest algorithm which is suitable for datasets with discrete variables. The high variance in the decision trees can be reduced by developing multiple trees with different samples of the training dataset and by limiting the features which the greedy algorithm can evaluate at each split point when creating a tree. Also

one small change in dataset while developing the decision tree model can have catastrophic effect on the structure of the tree and the features present at each node. Hyperparameter tuning can be done to get best results.