SEOUL BIKE SHARING PREDICTION

Team member: Mohd Sharik

Contents

1 INTRODUCTION......3

2 DATA OVERVIEW......3

3 DATA CLEANING......4

4 EXPLORATORY DATA ANALYSIS......4

4.1 Univariate analysis......4

4.2 Bi-Variate analysis......6

4.3 Correlation......9

5 FEATURE ENGINEERING......12

6 MODELLING......12

6.1 Train test split......13

6.2 Regression......14

6.3 Decision tree......14

6.4 XGBoost......16

7 Evaluation of model......17

7.1 MSE......17

7.2 MAE......18

7.3 R2......18

7.4 Adjusted R2......19

8 Data Visualization......19

9 Libraries......20

10 Summary...**21 & 22**

1 INTRODUCTION

Ddareungi (Korean: 따릉이) is Seoul's bike sharing system, which was set up in 2015. It is also named **Seoul Bike** in English.

Ddareungi was first introduced in Seoul in October 2015 in select areas of the right bank of the Han River. After a few months, the number of stations reached 150 and 1500 bikes were made available. In 2016, the number of stations has increased steadily to cover new districts. As of July 2016, there were about 300 stations and 3000 bikes available, and Seoul mayor Park Won-soon confirmed his intention to increase the number of bikes available to 20,000.

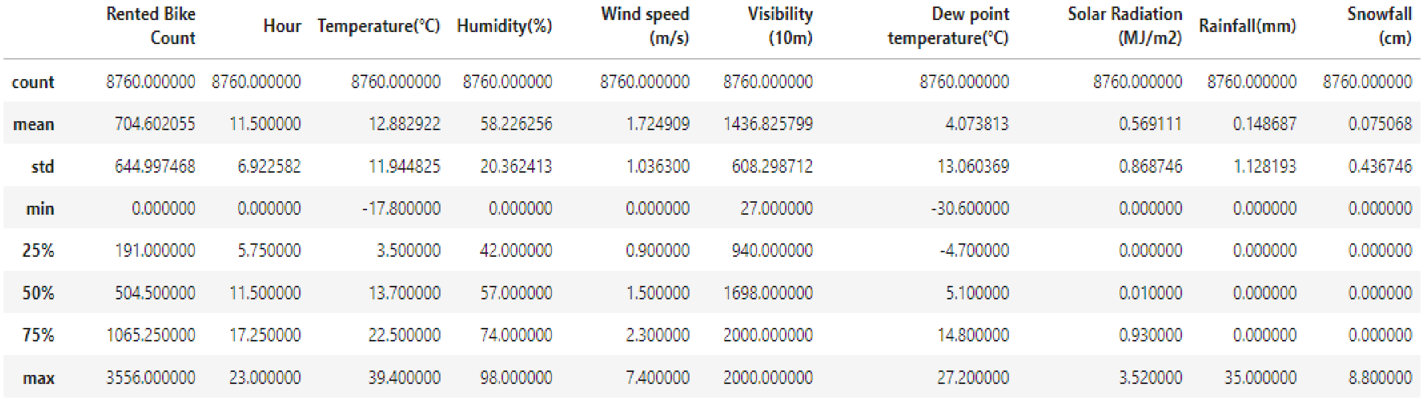
Bike sharing has been gaining importance over the last few decades. More and more people are turning to healthier and more liveable cities where activities like bike sharing are easily available. there are many benefits from bike sharing, such as environmental benefits.

2. Data overview

This dataset contains the hourly and daily count of rental bikes between years 2017 and 2018 in Capital bike share system with the corresponding weather and seasonal information. The dataset contains 8760 rows and 14 columns (the features which are under consideration). Attribute Information:

- Date – it’s a simple data column that contains information such as year-month-day.
- Rented Bike count – This column contains Count of bikes rented at each hour and our target column
- Hour – this column contains Hour of the day ranging from 0 to 23
- Temperature-this column contains information about Temperature (in Celsius) at the time of booking
- Humidity – This column contains humidity level at the time of booking
- Wind speed – this column contains information about wind speed in m/s
- Visibility - this column explains the visibility at the time of booking
- Dew point temperature – temperature of dew point in Celsius
- Solar radiation - MJ/m2 (information about solar radiation)
- Rainfall – mm (information about rainfall)
- Snowfall – cm (information about Snowfall)
- Seasons – this column contains for unique weather seasons and these values are Winter, Spring, Summer, Autumn
- Holiday - Holiday/No holiday (whether a day is a holiday or not)
- Functional Day - NoFunc(Non Functional Hours), Fun(Functional hours)

A summary of numeric features can be seen below .



	Rented Bike Count	Hour	Temperature(°C)	Humidity(%)	Wind speed (m/s)	Visibility (10m)	Dew point temperature(°C)	Solar Radiation (MJ/m2)	Rainfall(mm)	Snowfall (cm)
count	8760.000000	8760.000000	8760.000000	8760.000000	8760.000000	8760.000000	8760.000000	8760.000000	8760.000000	8760.000000
mean	704.602055	11.500000	12.882922	58.226256	1.724909	1436.825799	4.073813	0.569111	0.148687	0.075068
std	644.997468	6.922582	11.944825	20.362413	1.036300	608.298712	13.060369	0.868746	1.128193	0.436746
min	0.000000	0.000000	-17.800000	0.000000	0.000000	27.000000	-30.600000	0.000000	0.000000	0.000000
25%	191.000000	5.750000	3.500000	42.000000	0.900000	940.000000	-4.700000	0.000000	0.000000	0.000000
50%	504.500000	11.500000	13.700000	57.000000	1.500000	1698.000000	5.100000	0.010000	0.000000	0.000000
75%	1065.250000	17.250000	22.500000	74.000000	2.300000	2000.000000	14.800000	0.930000	0.000000	0.000000
max	3556.000000	23.000000	39.400000	98.000000	7.400000	2000.000000	27.200000	3.520000	35.000000	8.800000

3. Data Cleaning and outliers handling

In the given data we have seen there is no missing value and all the values are reasonable and also based on the model selection we can avoid this step as the model which we have decided are either tree based or ensemble technique due to this fact even if there are few outliers we can avoid this step and still we will get equivalent results.

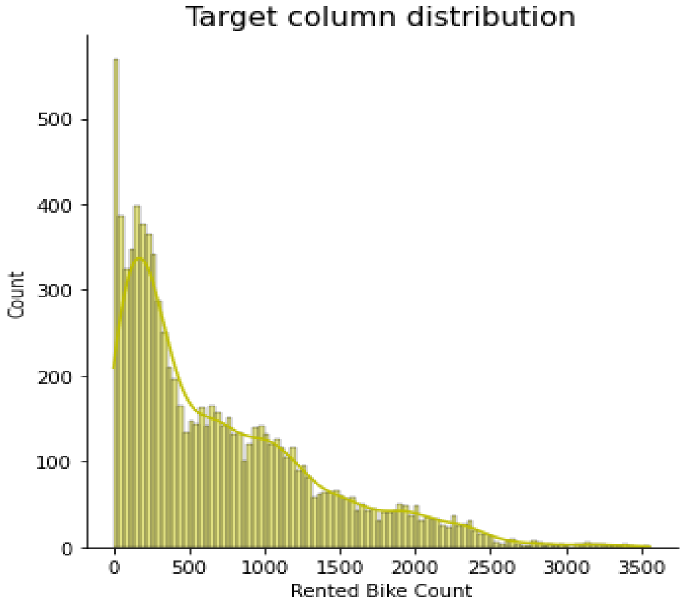
4. EXPLORATORY DATA ANALYSIS

Univariate analysis

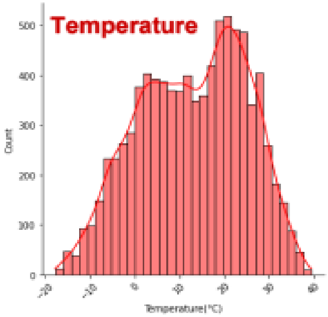
In this section of the project we deep dive into analysis of single feature hence calling it as Univariate analysis. then will explore distributions of features and try to understand which transformation should we apply and what is the reason behind it.

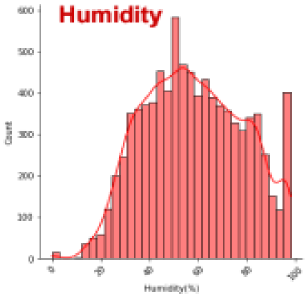
Before we start modeling, let us first get an idea on how the number of bike rentals depend on the various features provided to us one by one.

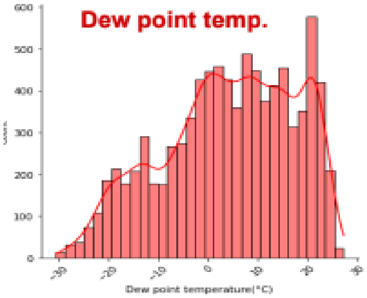
1. Bike count distribution itself

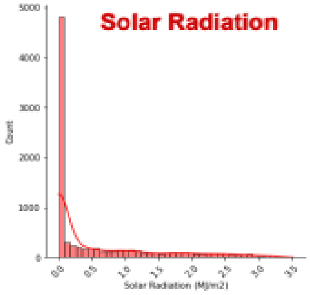


By plotting the distribution of Bike count itself we found that Distribution of the bikes is positively skewed and m Booking per hour are between 500 to 1000 and exactly m Is 704 which can be taken from above table.







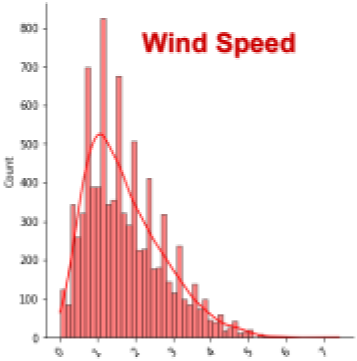


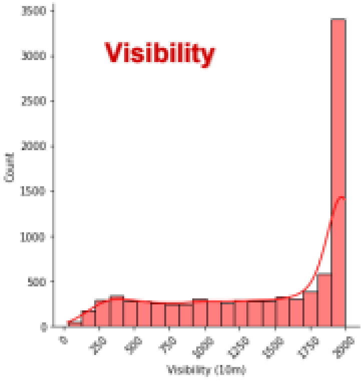
Temperature: We have noticed that temperature was normally distributed in certain range.

Humidity: We noticed that Humidity was somewhat negatively skewed and mean of the same was somewhere around 58 at the scale.

Dew point temp: Dew point temp distribution was negatively skewed means higher value of temperature was more likely to appear in the data.

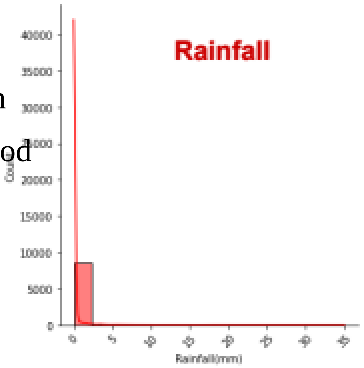
Solar radiation: this feature has recorded as 0 on the scale very high number of times and there were only few non zero values.

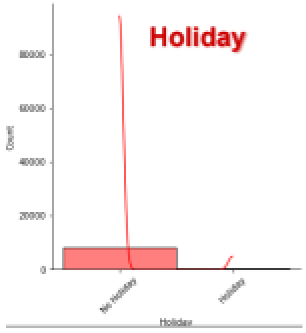




Wind Speed: Distribution of wind speed explains that it Was a positively skewed which means that higher the Value at scale lower the probability to be appeared in Dataset.

Visibility: we noticed that when visibility was less the Distribution appear to be somewhat uniform. However, When visibility scale was somewhere around 2000 then The count was very high which indicate that we had good visibility throughout the period.





Rainfall: most of the we have recorded almost no rainfall which is natural as per the geographic views.

Holiday: we have seen that the data that we were given contains more non-holidays than holidays which is a real practice.

This all information can be summaries as –

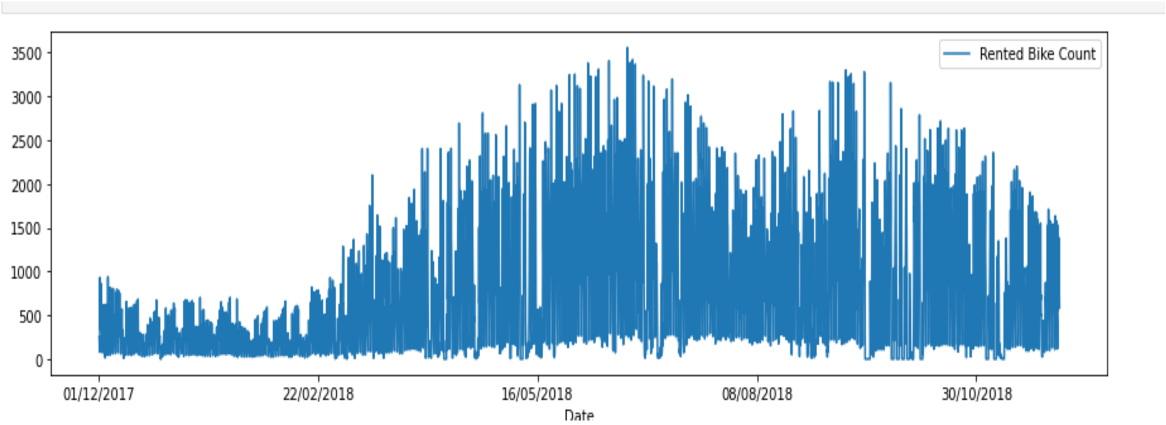
In this Section we have explored distributions of each feature of data set and have found interesting distribution as given below.

1. most of the features are not normally distributed.
2. Hour- somewhat uniformly distributed with some exceptions.
3. wind speed is positively skewed
4. visibility is negatively skewed
5. dew point feature is negatively skewed
6. solar radiation, rainfall, snowfall, functioning day, holiday, year. all these columns are biased towards a particular value

Since our data is randomly distributed hence we are pretty sure that we cannot apply any parametric algorithm here hence we will have to go for a non-parametric regression algorithm.

Bi-Variate analysis

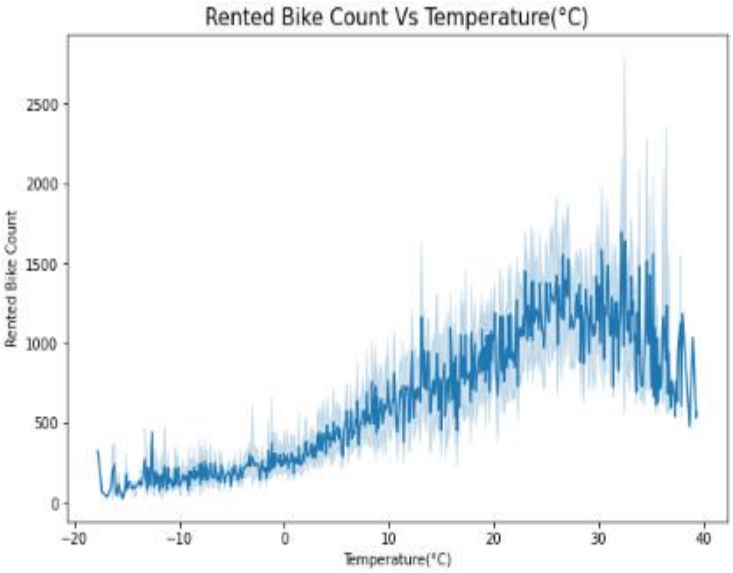
in this Section we will look at bi-variate analysis of features we will focus on the relation between target column and other features relation. since we have different type of value in dataset hence we will plot different plots for different datatypes. first we will look at continuous features against bike count. then we will see how categorical variables are behaving in reference of bike count and and then we will go for the correlation matrix where will find see correlation between features and then will remove unwanted features and build out final data frame that is to be feed to out ML algorithm.



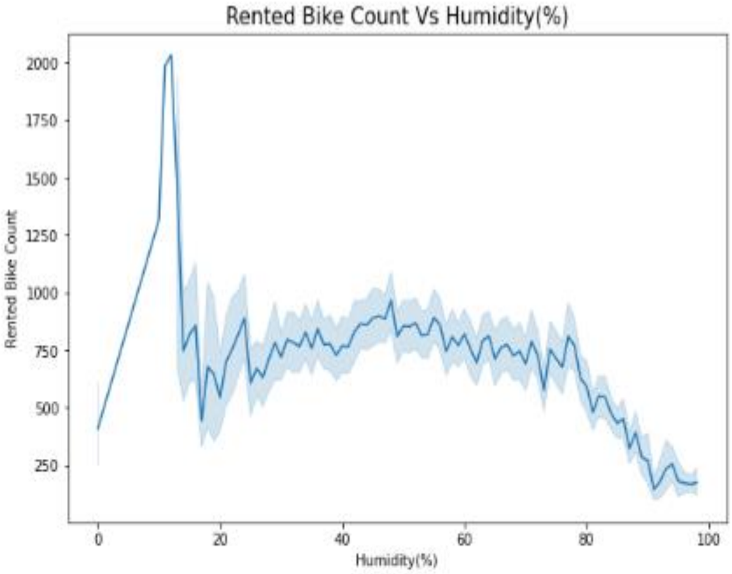
Date Vs Rented Bike counts :

We have seen that the number of rented bike counts was very less in Dec 2017 and then the number increased and finally we observed that the number was getting decreased in the end 2018

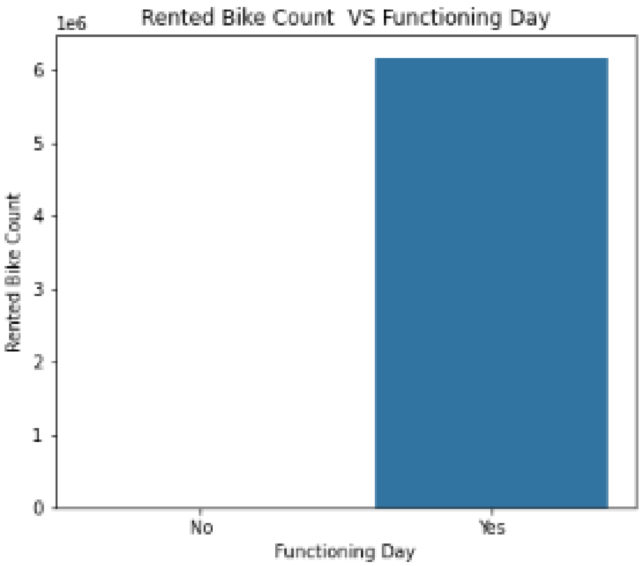
We found that there is a linear relationship between temperature and no. of bike counts up to a certain temperature (around 33C) then this relation starts in negative direction.

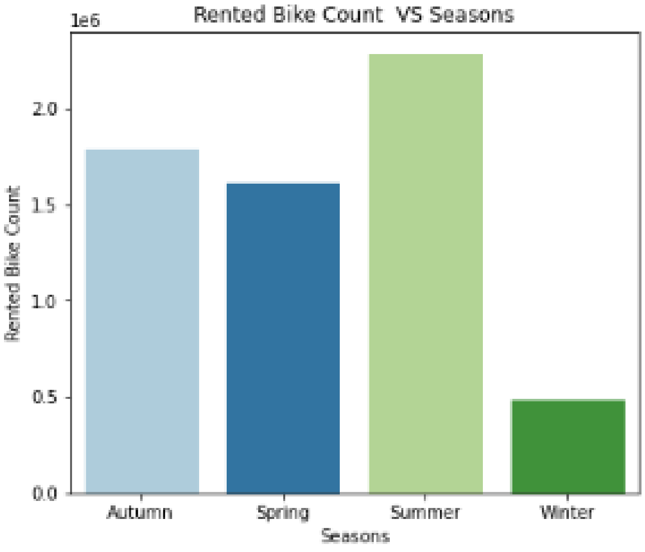


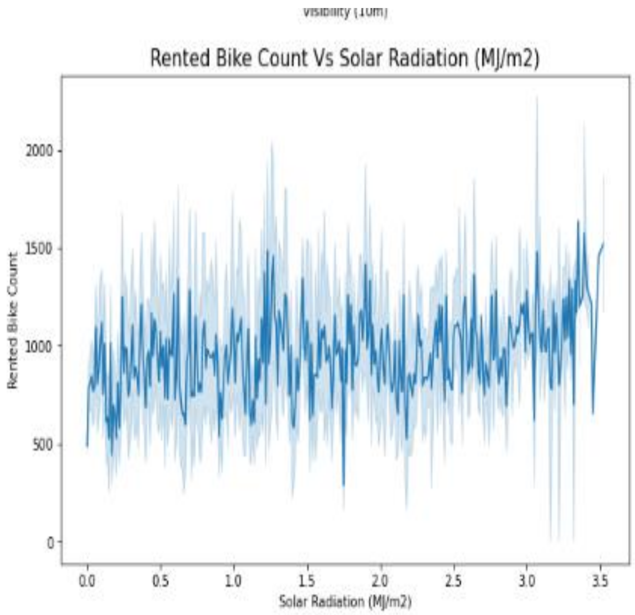
When humidity was less then count is very high however it suddenly decreases at 20 and remain somewhat constant then starts falling as humidity level increases further than 80

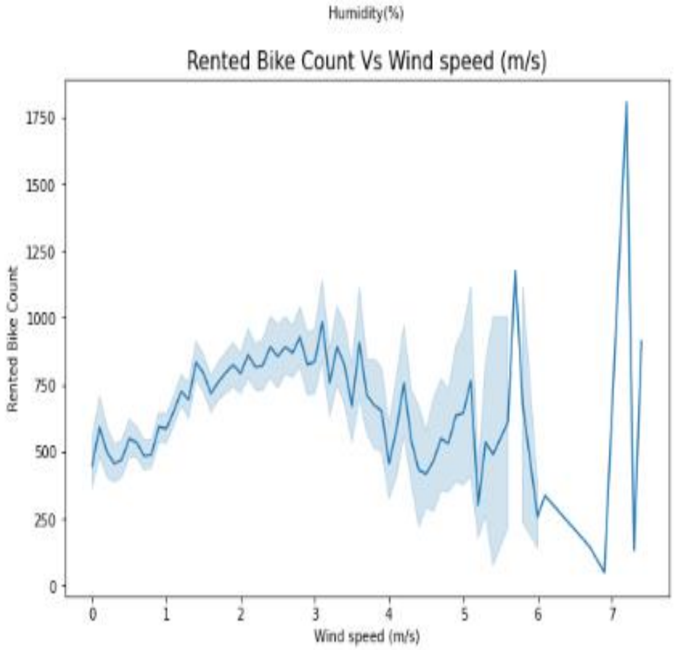


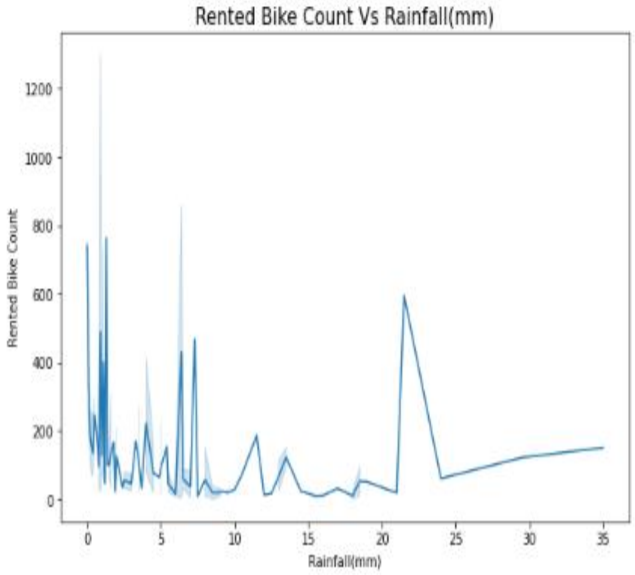
Similarly, other relations can also be seen in below figures.

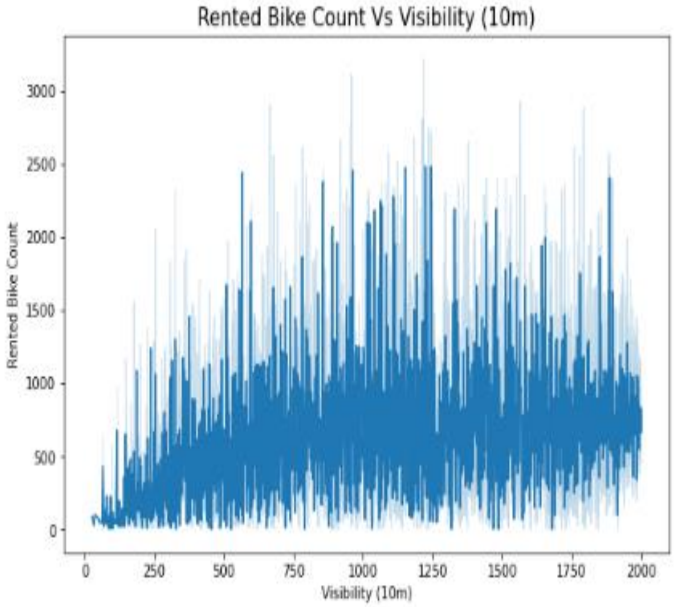


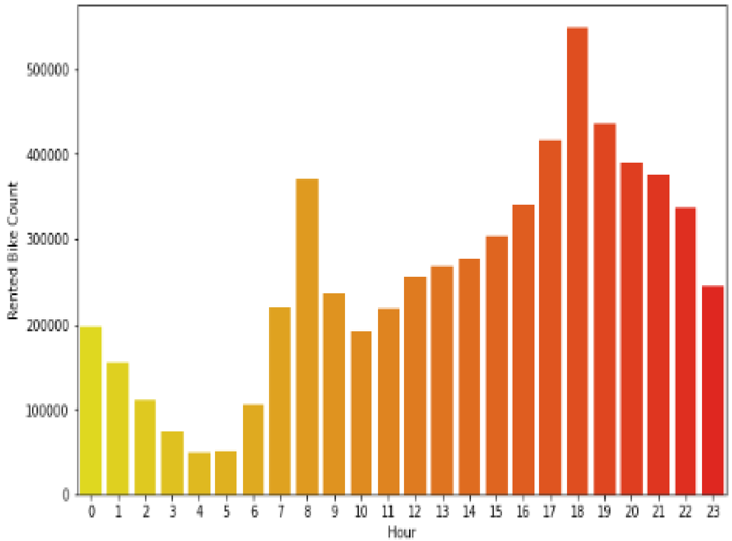


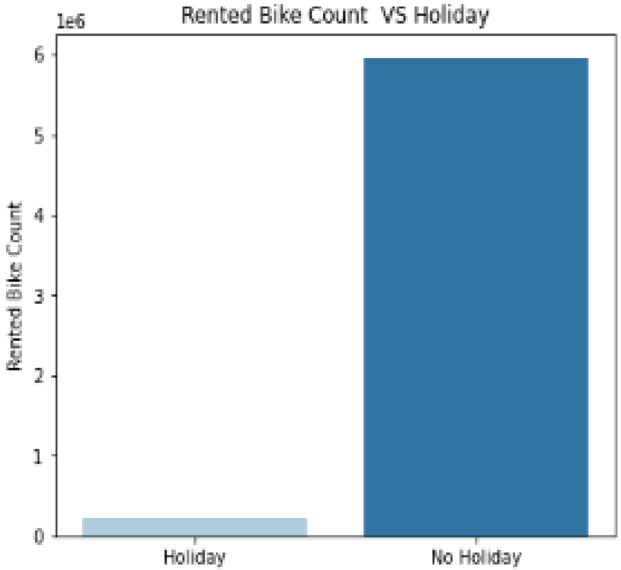

Visibility (10m)
Rented Bike Count Vs Solar Radiation (MJ/m2)
Rented Bike Count
0
500
1000
1500
2000
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
Solar Radiation (MJ/m2)

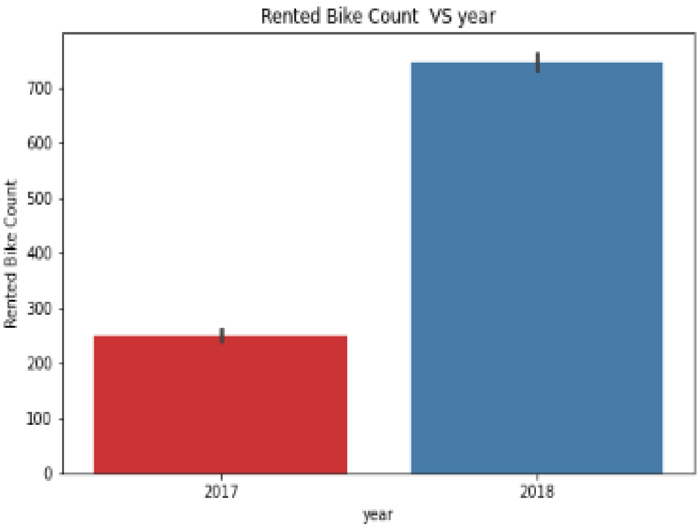

Humidity(%)
Rented Bike Count Vs Wind speed (m/s)
Rented Bike Count
0
250
500
750
1000
1250
1500
1750
0
1
2
3
4
5
6
7
Wind speed (m/s)

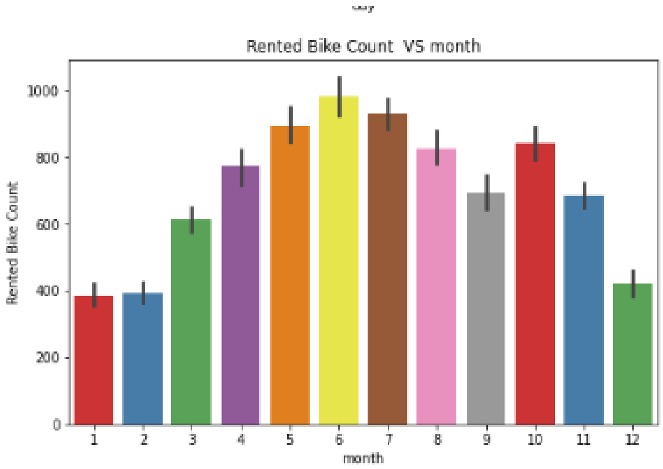

Rented Bike Count Vs Rainfall(mm)
Rented Bike Count
0
200
400
600
800
1000
1200
0
5
10
15
20
25
30
35
Rainfall(mm)

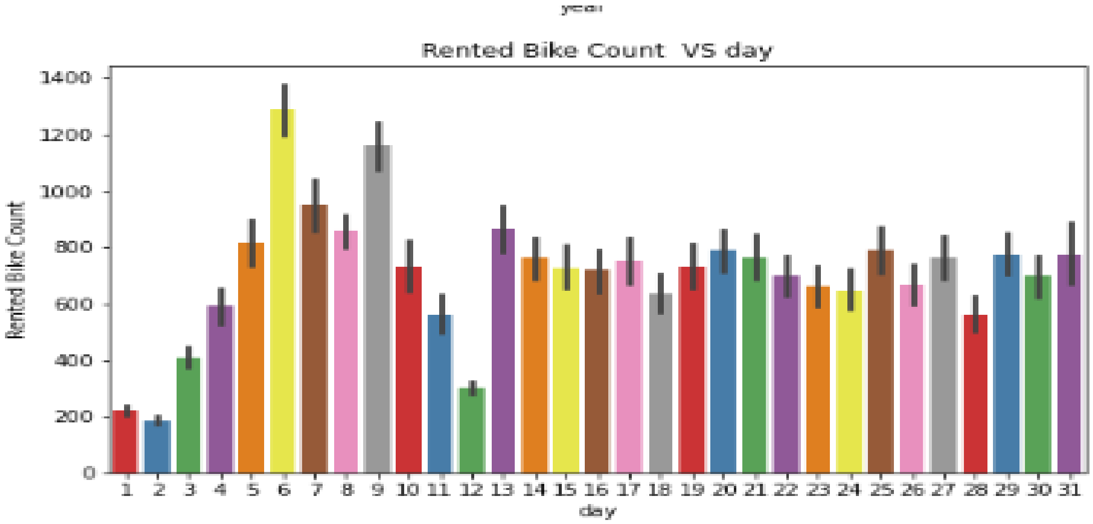

Rented Bike Count Vs Visibility (10m)
Rented Bike Count
0
500
1000
1500
2000
2500
3000
0
250
500
750
1000
1250
1500
1750
2000
Visibility (10m)


Rented Bike Count
0
100000
200000
300000
400000
500000
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
Hour


Rented Bike Count VS Holiday
1e6
Rented Bike Count
0
1
2
3
4
5
6
Holiday
No Holiday







Correlation:

Variables within a dataset can be related for lots of reasons.

For example:

- One variable could cause or depend on the values of another variable.
- One variable could be lightly associated with another variable.
- Two variables could depend on a third unknown variable.

It can be useful in data analysis and modeling to better understand the relationships between variables. The statistical relationship between two variables is referred to as their correlation.

A correlation could be positive, meaning both variables move in the same direction, or negative, meaning that when one variable's value increases, the other variables' values decrease. Correlation can also be neutral or zero, meaning that the variables are unrelated.

- **Positive Correlation**: both variables change in the same direction.
- **Neutral Correlation**: No relationship in the change of the variables.
- **Negative Correlation**: variables change in opposite directions.

The performance of some algorithms can deteriorate if two or more variables are tightly related, called multicollinearity. An example is linear regression, where one of the offending correlated variables should be removed in order to improve the skill of the model.

We may also be interested in the correlation between input variables with the output variable in order provide insight into which variables may or may not be relevant as input for developing a model.

The structure of the relationship may be known, e.g. it may be linear, or we may have no idea whether a relationship exists between two variables or what structure it may take. Depending what is known about the relationship and the distribution of the variables, different correlation scores can be calculated.

Pearson’s Correlation

The Pearson's Correlation Coefficient is also known as the Pearson Product-Moment Correlation Coefficient. It is a measure of the linear relationship between two random variables - X and Y. Mathematically, if (σXY) is the covariance between X and Y, and (σX) is the standard deviation of X, then the Pearson's correlation coefficient ρ is given by:

$$\rho_{X,Y} = \frac{\sigma_{XY}}{\sigma_X \sigma_Y}$$

The Pearson correlation coefficient measures the linear association between variables. Its value can be interpreted like so:

+1 - Complete positive correlation

+0.8 - Strong positive correlation

+0.6 - Moderate positive correlation

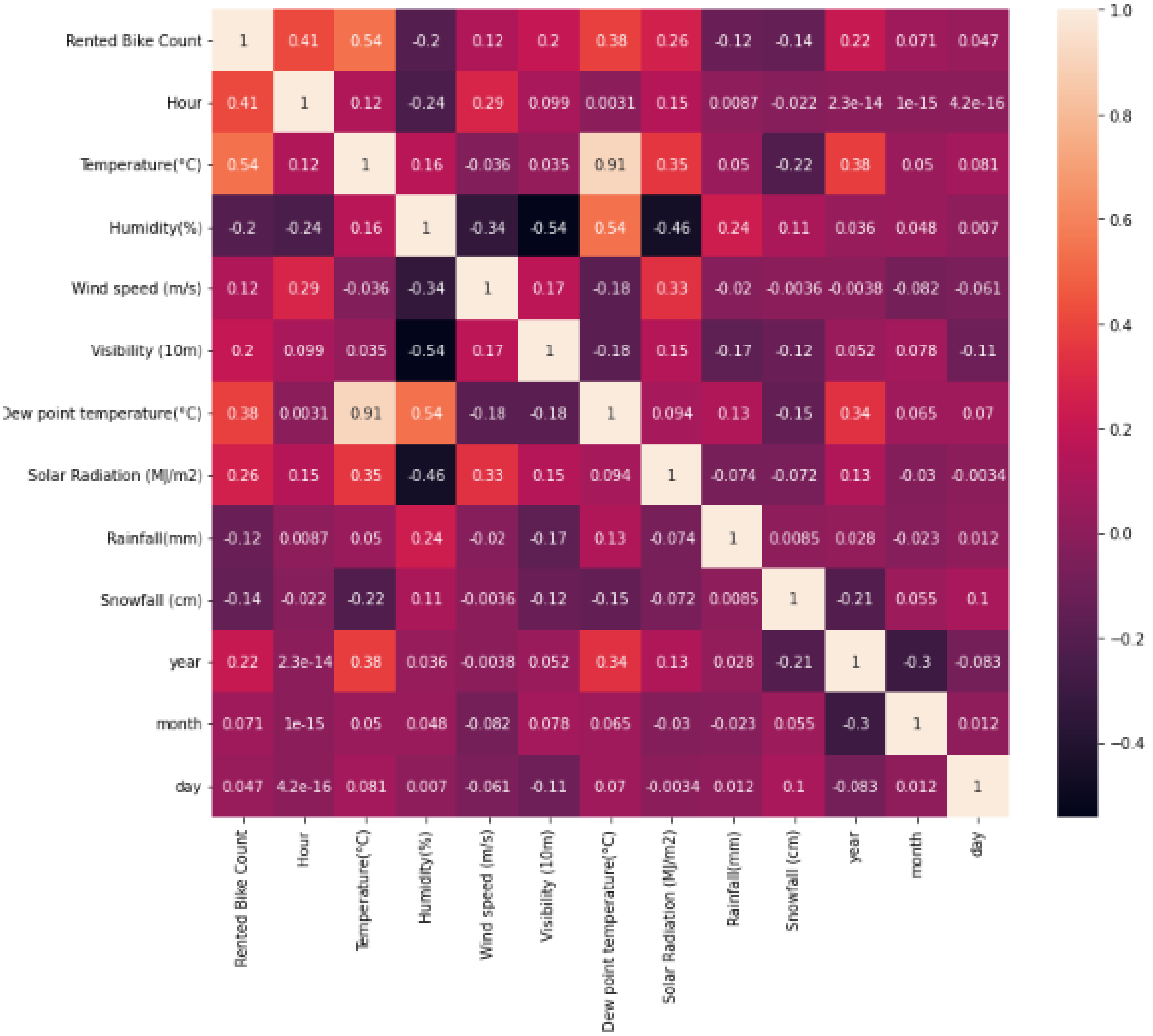
0 - no correlation whatsoever

-0.6 - Moderate negative correlation

-0.8 - Strong negative correlation

-1 - Complete negative correlation

In python we use correlation matrix to understand the correlation between two variables. We have also plotted same correlation matrix to understand the relation between two features.



in this section various interesting information and hence we can summarise the section as -

1. Bike count increases as temperature increases up to ~33C and then decrease.
2. when the humidity is between 20 and 80 bike count distribution is approx. similar and if it goes beyond 80 count goes lower down.
3. Visibility and radiation are directly not affecting the bike count.
4. Rainfall and snowfall are having random distribution with the bike count.
5. more bike counts are on functional and non-holiday days which is quite interesting.
6. from day 5 to day 10 of the month bike are high and then it shows uniform distribution.
7. people are using these bikes in evening in a very high extent.
8. rented bike count is less in winter season hence in comparison of other seasons.
9. we also have found that bike count is very high in 2018 in comparison of 2017
10. Finally, we have plotted correlation matrix and seen correlations between two features.

5. FEATURE ENGINEERING

The provided data in its raw form wasn't directly used as an input to the model. Several feature engineeringwas carried out where few features were modified, few were dropped, and few were added. Below is a summary of the feature engineering carried out with the provided data set

a. The *datetime* column which contained the date-time stamp in 'yyyy-mm-dd format was split into individual *['month', 'date', 'day']* categorical columns.

b. First we have removed highly correlated features from the data as highly correlated features may lead to bad performance. This was done with help of VIF factor which has direct relation with correlation and help to identify columns which may lead to poor results. The columns that we have removed are 'Dew point tempature' and 'year'

c. OneHotEncoding of categorical feature set

   i. *Seasons*: We apply one hot encoding on the seasons feature and as a result we get four columns having values 0 and 1. These columns are season_autumn, season_spring, season_summer, season_winter

   ii. *Holiday*: Split month column to *holiday*, *no_holiday columns and* as a result we get four columns having values 0 and 1.

   iii. Functioning day: We also apply same encoding to functioning day and thus we get fday_yes, and fday_no columns

d. Drop *season* column: This is because we already have new columns based on this column and onehotencoding was done to remove categorical features from the dataset

e. Drop *holiday* and *day* columns: The workingday column had information about holiday embeddedin it. *workingday* = weekday and not a holiday. Since we noticed that there were two kinds of bikerental behaviors - during working days and not a working day, we will retain only the *workingday*column and drop *'day'* and *'holiday'* column.

f. Drop functioning day column: as discussed above we will drop functioning day and fday_no column from the dataset.

6. MODELLING

Modeling Overview

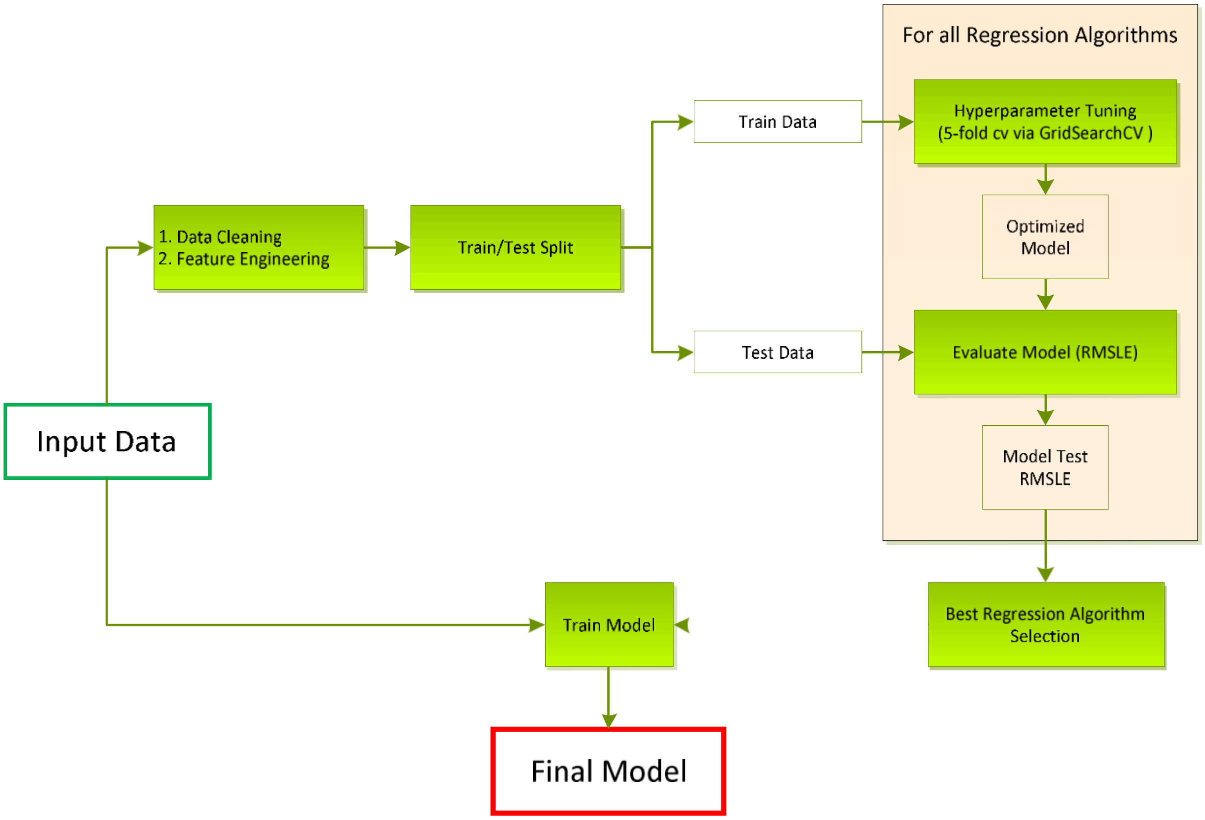


Figure 7-1: Overview of the Modelling procedure

Figure 7-1 depicts the overall procedure followed to obtain the Final Model. The provided data is first cleaned and transformed using Feature Engineering. We then split the data into Train set (for Hyperparameter tuning) and Test set (for Model Evaluation). Using RMSLE as our evaluation metric, we compare various models and select the regression algorithm based on the lowest RMSLE on the Test data.The final model used for submission is then obtained by again training the selected Regression Algorithmon the entire Input Data set.

Train/Test Split

Below figure summarizes the method to split the provided data into training and testing data set. we held out data from 20th to the end of the month (for every month) as test set (no labels, i.e., count values

have been provided for those data). We follow a similar approach to split the provided labelled data set(consisting of 1st to 19th of every month) into two sets

1. Training set
   a. This contains data of from the 1st to 15th of every month
   b. This set is used to train various model and obtain the best set of hyperparameters for thesemodels. We use GridSearchCV to tune the hyperparameters using this training set
2. Test set
   a. This contains data of from the 16th to 19th of every month
   b. This is used to evaluate all our models. The model with the best test score is finally chosen forsubmission

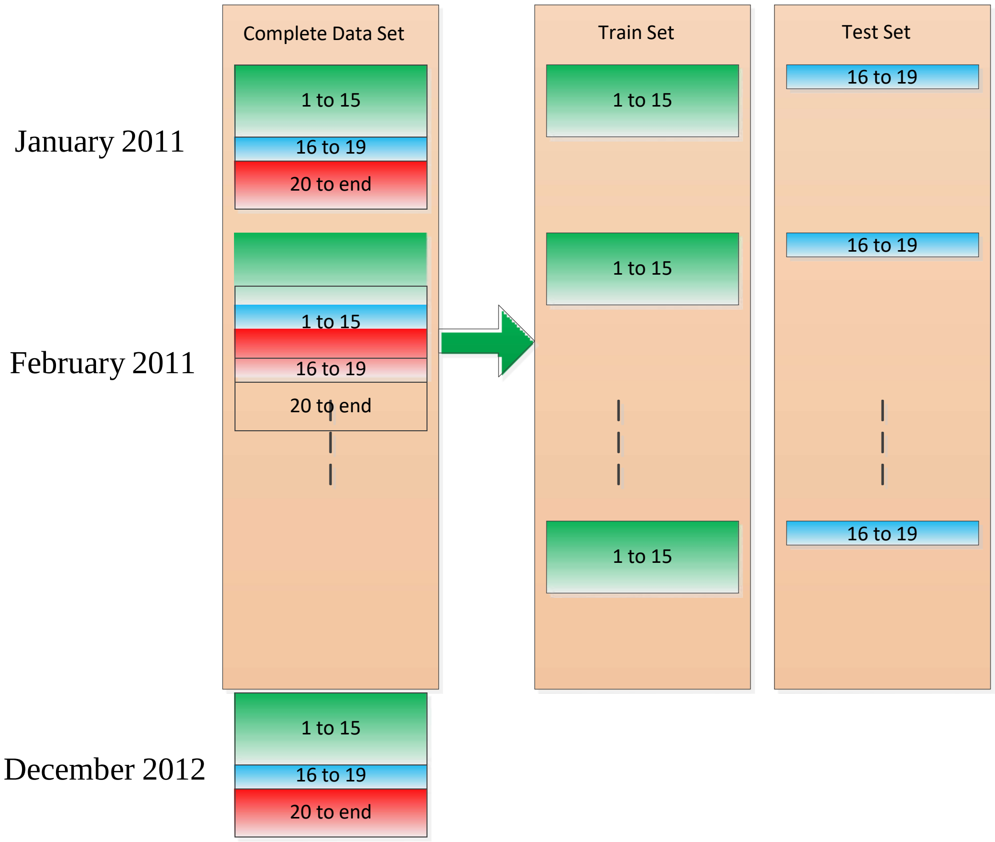


Figure 7-2: Splitting the provided data into training and testing set

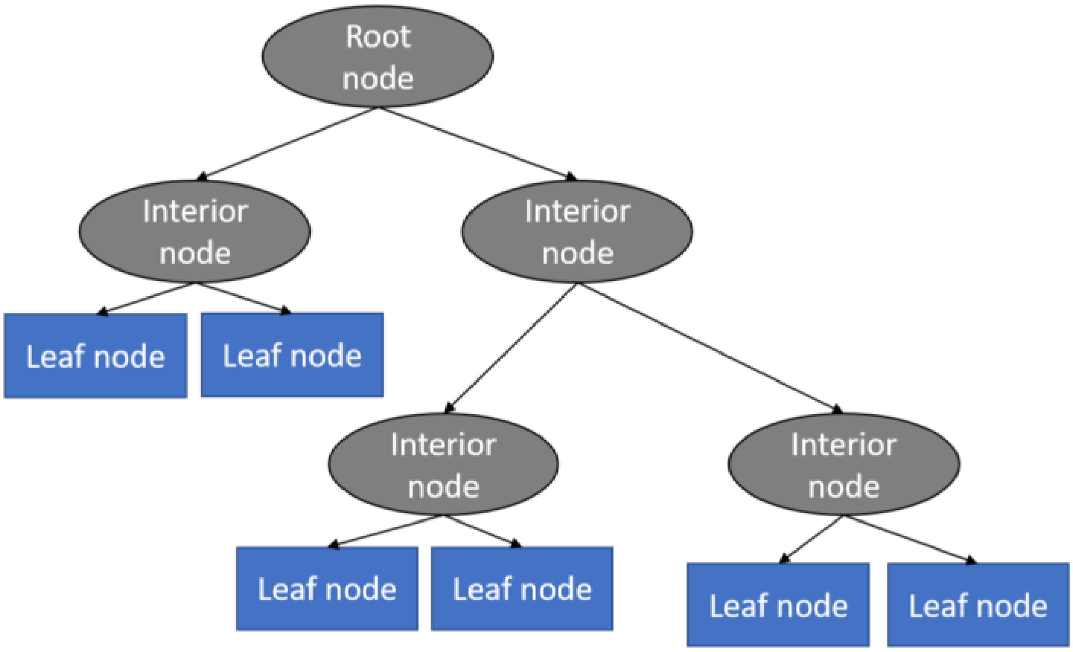
Regression:

Regression analysis is a way of mathematically sorting out which of those variables does indeed have an impact. It answers the questions: Which factors matter most? Which can we ignore? How do those factors interact with each other? And, perhaps most importantly, how certain are we about all of these factors? Regression methods that we used in our model are Decision tree and XGBoost.

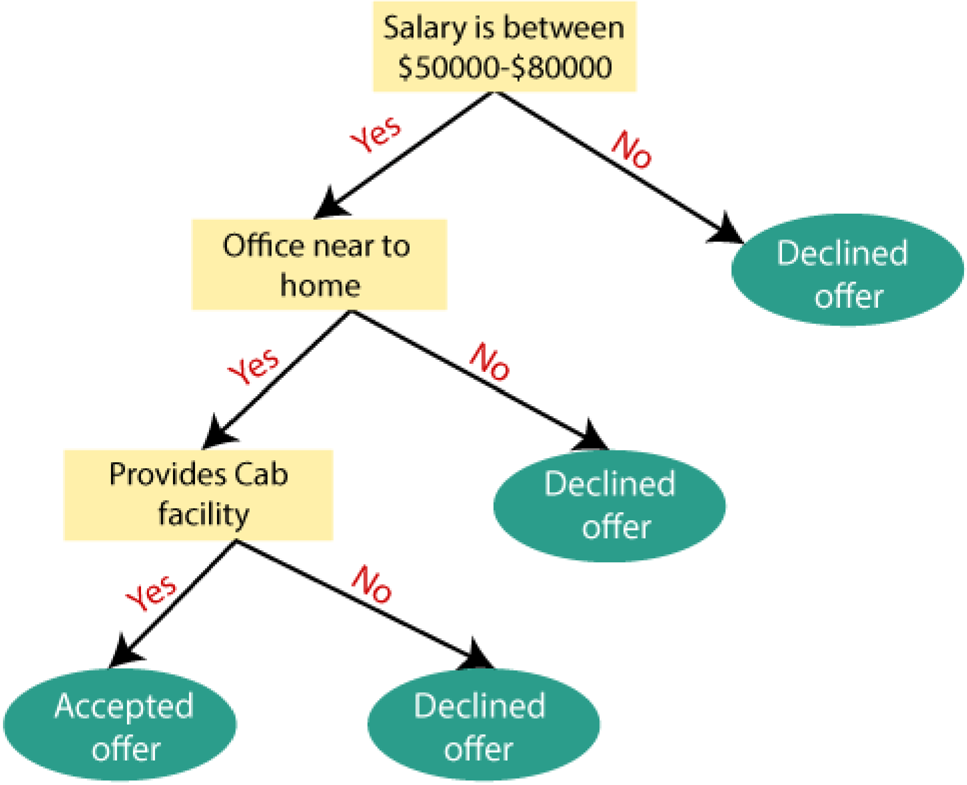
Decision Tree:

Decision Tree is one of the most commonly used, practical approaches for supervised learning. It can be used to solve both Regression and Classification tasks with the latter being put more into practical application.

It is a tree-structured classifier with three types of nodes. The ***Root Node*** is the initial node which represents the entire sample and may get split further into further nodes. The ***Interior Nodes*** represent the features of a data set and the branches represent the decision rules. Finally, the ***Leaf Nodes*** represent the outcome. This algorithm is very useful for solving decision-related problems.



With a particular data point, it is run completely through the entirely tree by answering *True/False* questions till it reaches the leaf node. The final prediction is the average of the value of the dependent variable in that particular leaf node. Through multiple iterations, the Tree is able to predict a proper value for the data point.



An example for Decision Tree Model

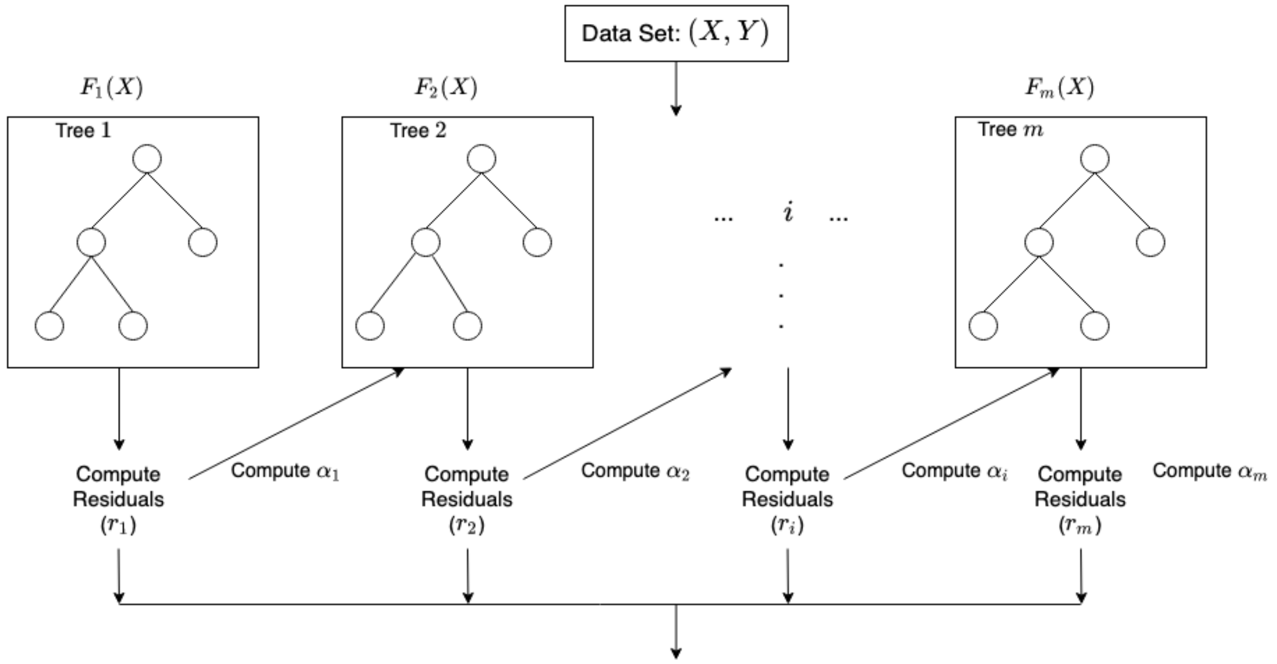
The above diagram is a representation for the implementation of a Decision Tree algorithm. Decision trees have an advantage that it is easy to understand, lesser data cleaning is required, non-linearity does not affect the model’s performance and the number of hyper-parameters to be tuned is almost null. However, it may have an over-fitting problem

XGBoost:

XGBoost is a popular and efficient open-source implementation of the gradient boosted trees algorithm. Gradient boosting is a supervised learning algorithm, which attempts to accurately predict a target variable by combining the estimates of a set of simpler, weaker models.

When using gradient boosting for regression, the weak learners are regression trees, and each regression tree maps an input data point to one of its leafs that contains a continuous score. XGBoost minimizes a regularized (L1 and L2) objective function that combines a convex loss function (based on the difference between the predicted and target outputs) and a penalty term for model complexity (in other words, the regression tree functions). The training proceeds iteratively, adding new trees that predict the residuals or errors of prior trees that are then combined with previous trees to make the final prediction. It's called gradient boosting because it uses a gradient descent algorithm to minimize the loss when adding new models.

Below is a brief illustration on how gradient tree boosting works.



$$F_m(X) \; = \; F_{m-1}(X) + \alpha_m h_m(X, r_{m-1}),$$

where α_i, and r_i are the regularization parameters and residuals computed with the i^{th} tree respectfully, and h_i is a function that is trained to predict residuals, r_i using X for the i^{th} tree. To compute α_i we use the residuals computed, r_i and compute the following: $arg \min_{\alpha} \; = \sum_{i=1}^{m} L(Y_i, F_{i-1}(X_i) + \alpha h_i(X_i, r_{i-1}))$ where $L(Y, F(X))$ is a differentiable loss function.

Extreme Gradient Boosting (XGBoost) is just an extension of gradient boosting with the following added advantages:

1. **Regularization:** Standard GBM implementation has no regularization like XGBoost, therefore it also helps to reduce overfitting. In fact, XGBoost is also known as ‘regularized boosting‘ technique.
2. **Parallel Processing:** XGBoost implements parallel processing and is blazingly faster as compared to GBM. But hang on, we know that boosting is sequential process so how can it be parallelized? We know that each tree can be built only after the previous one, but to make a tree it uses all the cores of the system. XGBoost also supports implementation on Hadoop.
3. **High Flexibility:** XGBoost allow users to define custom optimization objectives and evaluation criteria. This adds a whole new dimension to the model and there is no limit to what we can do.
4. **Handling Missing Values:** XGBoost has an in-built routine to handle missing values. User is required to supply a different value than other observations and pass that as a parameter. XGBoost tries different things as it encounters a missing value on each node and learns which path to take for missing values in future.
5. **Tree Pruning:** A GBM would stop splitting a node when it encounters a negative loss in the split. Thus it is more of a greedy algorithm. XGBoost on the other hand make splits upto the max_depth specified and then start pruning the tree backwards and remove splits beyond which there is no positive gain. Another advantage is that sometimes a split of negative loss say -2 may be followed by a split of positive loss +10. GBM would stop as it encounters -2. But XGBoost will go deeper and it will see a combined effect of +8 of the split and keep both.
6. **Built-in Cross-Validation:** XGBoost allows user to run a cross-validation at each iteration of the boosting process and thus it is easy to get the exact optimum number of boosting iterations in a single run. This is unlike GBM where we have to run a grid-search and only a limited values can be tested.
7. **Continue on Existing Model:** User can start training an XGBoost model from its last iteration of previous run. This can be of significant advantage in certain specific applications. GBM implementation of sklearn also has this feature so they are even on this point.

7. Evaluation of mode

Evaluation metrics are a measure of how good a model performs and how well it approximates the relationship. Let us look at **MSE, MAE, R-squared, Adjusted R-squared, and RMSE.**

Mean Squared Error (MSE)

The most common metric for regression tasks is MSE. It has a convex shape. It is the average of the squared difference between the predicted and actual value. Since it is differentiable and has a convex shape, it is easier to optimize.

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (\hat{y}_i - y_i)^2$$

Mean squared error. Image by the author.

MSE penalizes large errors.

Mean Absolute Error (MAE)

This is simply the average of the absolute difference between the target value and the value predicted by the model. Not preferred in cases where outliers are prominent.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|\hat{y}_i - y_i|$$

Mean absolute error. Image by the author.

MAE does not penalize large errors.

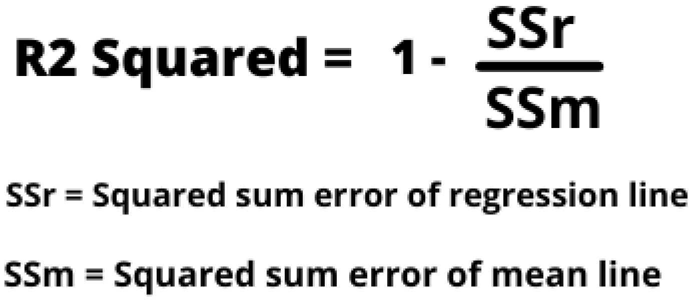
R Squared (R2)

R2 score is a metric that tells the performance of your model, not the loss in an absolute sense that how many wells did your model perform.

In contrast, MAE and MSE depend on the context as we have seen whereas the R2 score is independent of context.

So, with help of R squared we have a baseline model to compare a model which none of the other metrics provides. The same we have in classification problems which we call a threshold which is fixed at 0.5. So basically R2 squared calculates how must regression line is better than a mean line.

Hence, R2 squared is also known as Coefficient of Determination or sometimes also known as Goodness of fit.



$$\text{R2 Squared} = 1 - \frac{SSr}{SSm}$$

SSr = Squared sum error of regression line

SSm = Squared sum error of mean line

R2 Squared

Now, how will you interpret the R2 score? suppose If the R2 score is zero then the above regression line by mean line is equal means 1 so 1-1 is zero. So, in this case, both lines are overlapping means model performance is worst, It is not capable to take advantage of the output column.

Now the second case is when the R2 score is 1, it means when the division term is zero and it will happen when the regression line does not make any mistake, it is perfect. In the real world, it is not possible.So we can conclude that as our regression line moves towards perfection, R2 score move towards one. And the model performance improves.

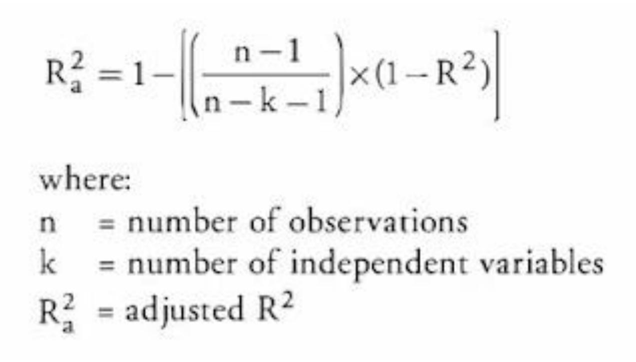
The normal case is when the R2 score is between zero and one like 0.8 which means your model is capable to explain 80 per cent of the variance of data.

Adjusted R Squared

The disadvantage of the R2 score is while adding new features in data the R2 score starts increasing or remains constant but it never decreases because It assumes that while adding more data variance of data increases.

But the problem is when we add an irrelevant feature in the dataset then at that time R2 sometimes starts increasing which is incorrect.

Hence, To control this situation Adjusted R Squared came into existence.



$$R_a^2 = 1 - \left[\left(\frac{n-1}{n-k-1}\right) \times (1 - R^2)\right]$$

where:
n = number of observations
k = number of independent variables
R_a^2 = adjusted R^2

Now as K increases by adding some features so the denominator will decrease, n-1 will remain constant. R2 score will remain constant or will increase slightly so the complete answer will increase and when we subtract this from one then the resultant score will decrease. so this is the case when we add an irrelevant feature in the dataset.

And if we add a relevant feature then the R2 score will increase and 1-R2 will decrease heavily and the denominator will also decrease so the complete term decreases, and on subtracting from one the score increases.

8. Data Visualization

Data visualization is the graphical representation of information using visual elements like charts, graphs, and maps. Data visualization tools provide an accessible way to see and understand trends, outliers, and patterns in data. In the world of Big Data, data visualization tools and technologies are essential to analyze massive amounts of information and make data-driven decisions. Here we make use of the tools matplotlib and seaborn to visualize data. The distributed plot, bar plot and scatter plot are the few graphical representations that are included in our work.

9. Libraries:

1. **NumPy:** NumPy stands for Numerical Python. It is the most fundamental library package in python and lays the foundation of multiple useful libraries in python. NumPy deals with array manipulation, basic mathematical and statistical operations, random number simulation, and much more.

2. **PANDAS:** Pandas is the data scientists go to python package. It is primarily used for data wrangling and analysis. It offers data structures and operations for manipulating numerical tables and time series. It is the most extensively used library in the project.

3. **Matplotlib:** Matplotlib is the plotting library closely associated with NumPy. It offers support for a number of different kinds of visualizations. This is the staple plotting package used by data analysts for basic and exploratory data analysis

1. **Seaborn:** Seaborn is another prominent plotting library used by analysts. Seaborn has extended support and compatibility for pandas and has flexible syntax in comparison to matplotlib. It includes a wider range of visualization types

Scikit-learn is probably the most useful library for machine learning in Python. The sklearn library contains a lot of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction.

10. Summary

About the Dataset:

The dataset that we have chosen is Seoul Bike data and has 8760 and 14 columns.these columns are mixed between numerical and categorical columns**.** In this dataset we will analyze all the possible columns which we can use in the prediction of rented bike counts

1. Introduction: in this section we have introduced the data and about the data that we are given.
2. Data Overview: In section 2 we have seen many details about the data such as feature explanation data summary.
3. Data Cleaning and outliers handling
4. In this section have focused on EDA of features in both way univariate and Bi-variate distributions. We also have seen feature correlations.
5. Feature Engineering: In this section we have prepared our data for ML algorithm.
6. Modeling: In this section we have understood train test split and some regression model
7. Evaluation: In this section we have learnt about evaluation of model.

Challenges faced:

- In order to use a parametric algorithm and upon analyzing features we found that decidingthe transformation was not so easy as we have noticed various distributions.
- Deciding a non-parametric algorithm was a challenge as we were not getting very good results using some of the algorithms.
- Bi-Variate analysis section was a bit difficult as there was lot of importance and we were supposed get best of it.