# Topic : Seoul Bike Sharing Demand Prediction

**For capstone project using machine learning(regression)**

**Kundan Lal, Abhijeet Kulkarni, Pankaj Ganjare, Akshay Auti**

**Data Science Trainees**

**Alma Better**

**Abstract:**

# Our study encompasses the findings done on Seoul Bike (bike sharing demand prediction) csv data file.

# In this we tackles the problem of predicting the number of bikes which will be rented at any given hour, henceforth referred to as the problem of ‘Bike Sharing Demand’. The date set consists of 8760 rows and 14 columns .

# In this vein, we investigates the efficacy of standard machine learning techniques namely Regression, Random Forests, Boosting by implementing and analyzing their performance with respect to each other.

**Problem statement:**

### 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.

**Introduction to Seoul bike Data set.**

There is 1 .csv file we have in our field of study namely ‘seoul bike’ data .csv we have delt with 14 columns namely Date, Rented bike count, Hour, Temperature, Humidity, Windspeed, Visibility, Due point temperature, Solar radiation, Rainfall, Snowfall, Seasons, Holiday, Functioning day .

The dataset contents weather information such as (temp, humidity, windspeed, visibility, dewpoint, solar radiation, snowfall, rainfall) the number of bikes rented per hour and date information

### **Attribute Information:**

### **Date : year-month-day**

### **Rented Bike count - Count of bikes rented at each hour**

### **Hour - Hour of he 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)**

Below is the original actual info about the data set using df2.info() method

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 8760 entries, 0 to 8759

Data columns (total 14 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Date 8760 non-null datetime64[ns]

1 Rented\_Bike\_Count 8760 non-null int64

2 Hour 8760 non-null int64

3 Temperature 8760 non-null float64

4 Humidity 8760 non-null int64

5 Wind\_speed 8760 non-null float64

6 Visibility 8760 non-null int64

7 Dew\_point\_temperature 8760 non-null float64

8 Solar\_Radiation 8760 non-null float64

9 Rainfall 8760 non-null float64

10 Snowfall 8760 non-null float64

11 Seasons 8760 non-null object

12 Holiday 8760 non-null object

13 Functioning\_Day 8760 non-null object

dtypes: datetime64[ns](1), float64(6), int64(4), object(3)

### memory usage: 958.2+ KB

### 

# Importing the libraries

import numpy as np

import pandas as pd

from numpy import math

from sklearn.preprocessing import MinMaxScaler

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import r2\_score

from sklearn.metrics import mean\_squared\_error

import seaborn as sns

import matplotlib.pyplot as plt

%matplotlib inline

import warnings

from sklearn.preprocessing import LabelEncoder,OneHotEncoder

warnings.filterwarnings('ignore')

#Mounting drive

from google.colab import drive

drive.mount('/content/drive')

#Reading the csv file and sorting in variable

df2 = pd.read\_csv('/content/drive/MyDrive/Almabetter/ML project supervised Regression/SeoulBikeData.csv',encoding =('ISO-8859-1'),low\_memory=False)

**Processing of dataset :**

* Changing date-time string to date time format
* Splitting timestamps into days, months, years and day of the week.
* Converting season, holiday, working day and weather into categoric variables or factors.
* Converting hours into a factor.

These transformations allowed us to extract certain key features of the dataset namely, the day of the week and the year, which proved to be pertinent in further analysis.

Such as null values, unique values, rename of columns for easier analysis

**Exploratory Data Analysis**

**Objective**

After cleaning of the data set ,we first wanted to do exploratory data analysis on the dataset to find out,

* The type of correlation of all independent variables vs dependant variables, to check whether they are positively correlated or negatively correlated.
* These analysis we have done for categorical variables and numerical variables separately.

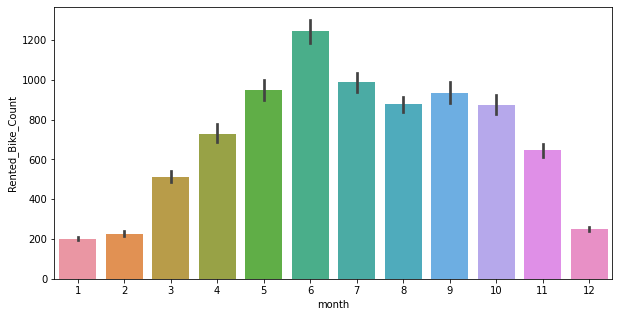
**Findings**

On categorical variables

#Analysing on Month Basis

plt.figure(figsize=(10,5))

sns.barplot(data=df2,x='month',y='Rented\_Bike\_Count')



From above bar graph we can analyse that from April to October the Demand is high of Rented bike count being June as highest.

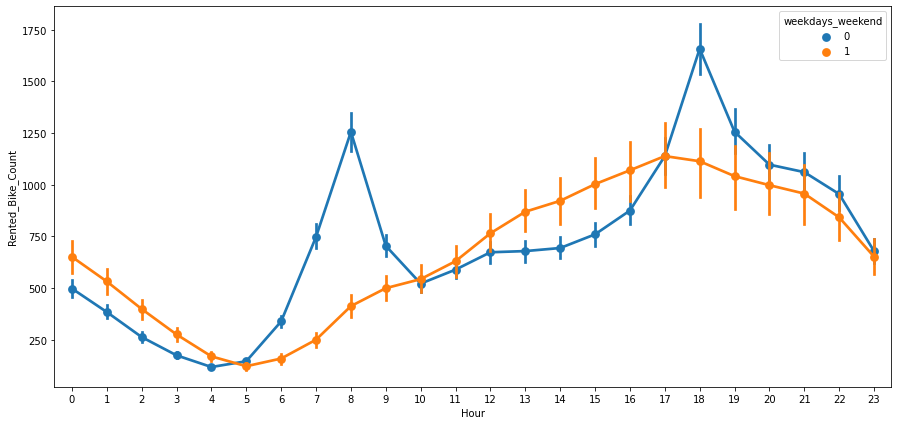
**Findings & Recommendations**

So it is our finding that as there is high no. of Rented bike count from May to October so during this period 1200 bikes should be made available to meet with the demand & supply requirement.

#Count of Rented Bike on Hour Basis

plt.figure(figsize = (15,7))

sns.pointplot(data=df2,x='Hour', y='Rented\_Bike\_Count',hue='weekdays\_weekend')



From the above point plot it can be analysed that the usage of rented bikes are more during weekdays than weekends.

Also that during weekdays from 5 am to 10 am and evening from 4 pm to 8 pm the renting is highest .

looking at orange line it can be said that during weekends the renting is very low during morning hours but as the evening comes closer gradually the rented numbers increases being maximum around 5 pm .It can be inferred that may be these must be getting used by people for leisure or family /personal use , as it’s a non-official day ,being weekend.

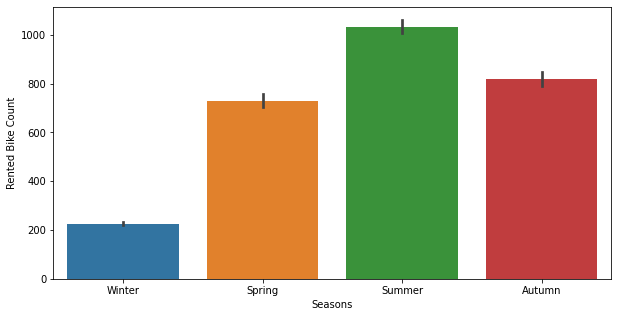
**Findings & Recommendations**

Our Findings are that concentration of more available bikes should be made in and around commercial office sectors during weekdays and can be shifted to residential areas during weekends.

# Analysing Bike rented with regards to Seasons

plt.figure(figsize=(10,5))

sns.barplot(data=df, x='Seasons', y='Rented Bike Count')



From Above barplot it can be seen easily that that Winters are the down season for renting , the number of rented bikes increases in Spring and Autumn and are nearly equal .

Also it can be seen that during summers the rented bike count is highest , concluded earlier also the average demand during summers is 1000 no. and peak requirement reaches during June which is around 1200 no.

Also the renting of Bikes during Summers is approximately 30% higher than Spring and Autumn .

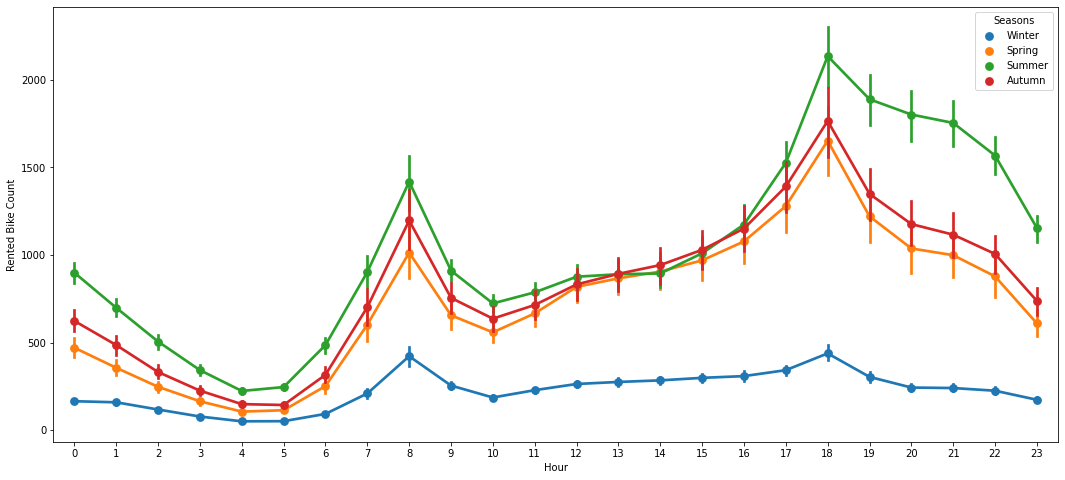
**Findings & Recommendations**

The infrastructure required of the rented bike company and the no. of available rented bikes should be at its best during summers , to cater to approximate 1000 -1200 numbers per day.

# Analysing the bike rented according to seasons and hour

plt.figure(figsize=(18,8))

sns.pointplot(data=df, x="Hour", y='Rented Bike Count', hue='Seasons')



Above point plot shows that the trend of booking is similar for Summer , Autumn and Spring , which shows peak renting from 6 am to 9 am & from 4 Pm to 10 Pm.

Also it can be observed that at seoul , people prefer to use rented bikes more during evening hours than morning hours.

The renting is lowest in Winter season.

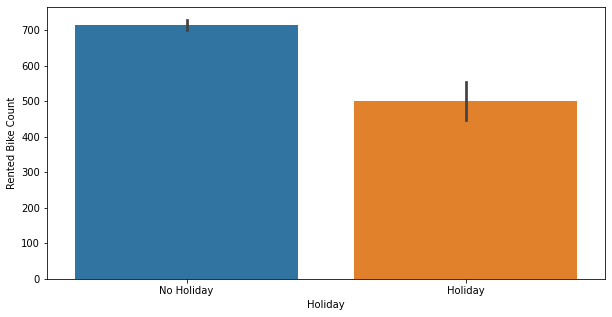
**Findings & Recommendations**

Evenings of summers are most crucial part as the demand is highest reaching 1900 no. of bike requirement. The arrangement should be accordingly to cater to the demand.

#Analysing according to Holiday

plt.figure(figsize = (10,5))

sns.barplot(data=df, x='Holiday', y='Rented Bike Count')



It can be easily seen from above that the higher number of Renting is done on weekdays and lower on holidays

**Findings & Recommendations**

The demand during Holidays is 40% lesser than working days , so if the Bike numbers are enough for working days , Holidays demand numbers can be managed easily .

**Findings done on numerical variables**

Filtering all numerical columns

numerical\_variables = list(df.select\_dtypes(['int64','float64']).columns)

Used for loop for plotting histograms to show distribution of all independent variables .

#displaying displots to analyze the distribution of all numerical features

for col in numerical\_variables:

  fig = plt.figure(figsize = (10,5))

  ax = fig.gca()

  feature = df[col]

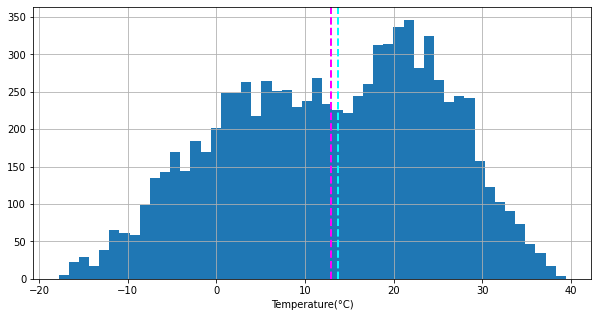
  feature.hist(bins=50, ax=ax)

  ax.axvline(feature.mean(), color='magenta', linestyle='dashed', linewidth=2)

  ax.axvline(feature.median(), color='cyan', linestyle='dashed', linewidth=2)

  plt.xlabel(col)

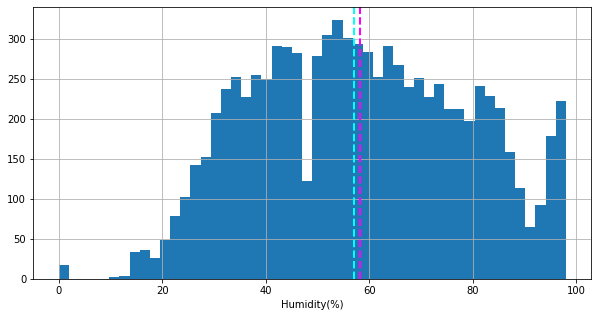
  plt.show()



Here we have used histogram to find the distribution of data for ‘Temperature’ variable , also to check its mean and mode ,

By looking at it we can say that the distribution resembles Normal distribution .

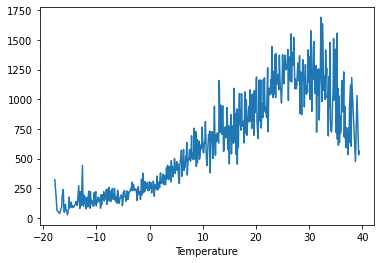
We checked such distributions for all independent variables .



‘Humidity’ variable also shows close to normal distribution.

## **Analysing Rented Bike Count (Dependent Variable) Vs ('Temperature' , 'Snowfall' ,'Rainfall' using their mean value)**

df2.groupby('Temperature').mean()['Rented\_Bike\_Count'].plot()



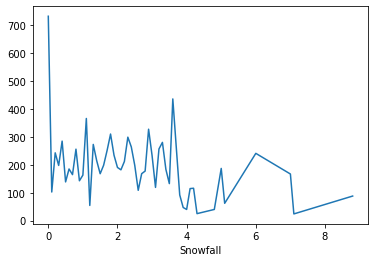
From Above it can be inferred that People prefer to rent a bike most , when the temperature is around 25-33 degree Centigrade.

**Findings & Recommendations**

Using the weather forecast the estimate of normal to high renting of bikes can be made as if the days temperature is between 25-33 degrees the demand can be higher than normal and to be prepared accordingly.

#print the plot to analyze the relationship between "Rented\_Bike\_Count" and "Snowfall"

df2.groupby('Snowfall').mean()['Rented\_Bike\_Count'].plot()



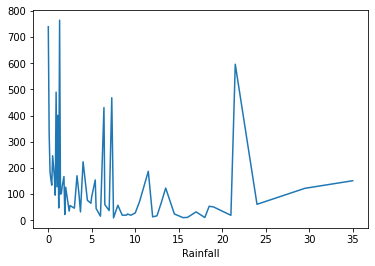
It can be analysed that renting of the bikes are maximum when there is no Snow but it decreases as snowfall starts and drastically slows down after 4 cm of snowfall.

**Findings & Recommendations**

During winters and areas where snowfall is likely to take place can expect downfall in demand by around 40-45%.

#print the plot to analyze the relationship between "Rented\_Bike\_Count" and "Rainfall"

df2.groupby('Rainfall').mean()['Rented\_Bike\_Count'].plot()



It can be seen that opposite of the expectation, there is little decrease in the renting of the bikes even if its raining, intermittently there are surges in the renting numbers.

**Findings & Recommendations**

The Rain does not slows down the Renting rate as much as Snowfall does.

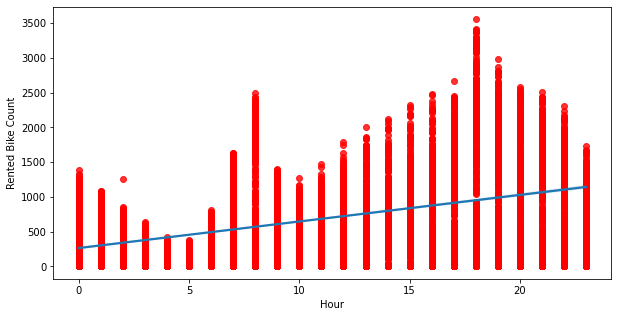
## **We plotted regression Plot to show their Linear Relationship with Target Variables.(+tive or –tive)**

# the regression plot for all the numerical features

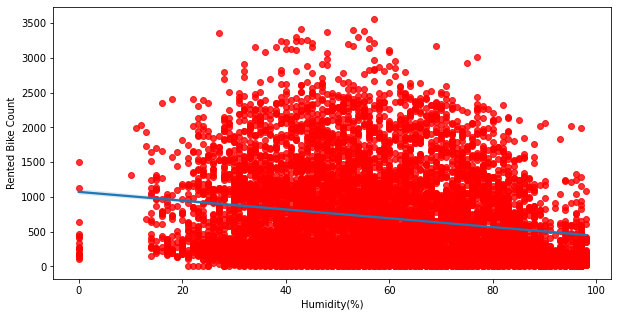
for col in numerical\_variables:

  plt.figure(figsize=(10,5))

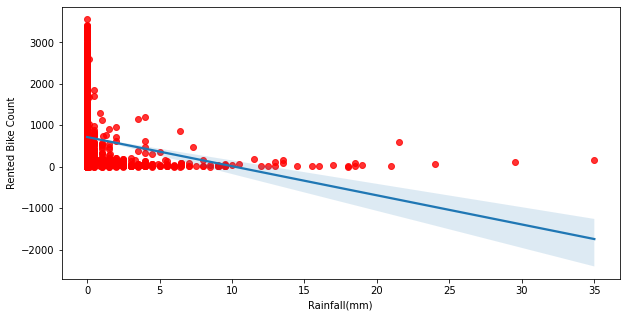
  sns.regplot(x=df[col], y='Rented Bike Count',data=df, scatter\_kws ={'color':'red'})



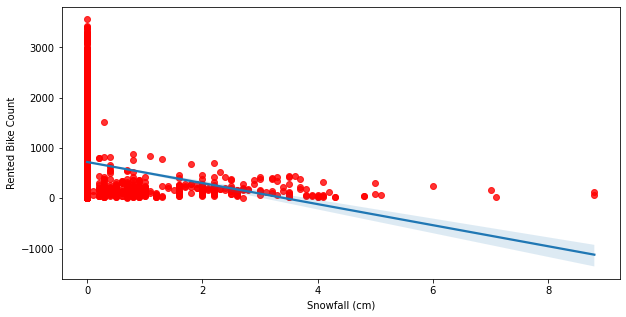
By applying regression plot and by looking at the best fit line we can clearly say that ‘Hour’ variable is Positively correlated with rented bike count variable.



‘Humidity’ variable is Negatively correlated with Rented Bike Count dependant variable ..,



‘ Rainfall ‘ is negatively correlated variable to Rented bike count.

 From the above regression plot of all numerical features we see that the columns 'Temperature', 'Wind\_speed', 'Visibility', 'Dew\_point\_temperature', 'Solar\_Radiation' are positively relation to the target variable which means the rented bike count increases with increase of these features.

Rainfall' ,'Snowfall' ,'Humidity' these features are negatively related with the target variable which means the rented bike count decreases when intensity of these features increase.

Firstly we want to convert data columns to data format conversion

# Date columns to Date format conversion

df2['Date'] = df2['Date'].apply(lambda x:dt.datetime.strptime(x, "%d/%m/%Y"))

# extracting day,month, day of week and weekdays/weekend from date column

df2['day'] = df2['Date'].dt.day\_name()

df2['month'] = df2['Date'].dt.month

df2['year'] = df2['Date'].dt.year

df2.drop(columns = 'Date', axis=1, inplace=True)

df2['weekdays\_weekend'] = df2['day'].apply(lambda x: 1 if x=='Saturday' or x=='Sunday' else 0)

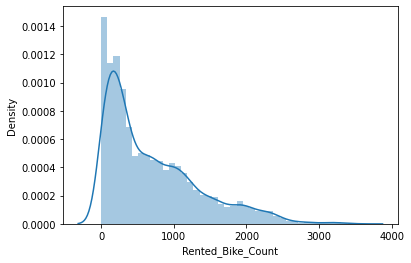
df2.drop(columns={'day','year'},axis=1,inplace=True)

#Visualising distribution

The first visualization I usually make for distributions is a histogram. You can see here that this is a terrible and uninformative way to look at the data. The differences in the sample sizes between the different groups makes them incomparable using this method.

#visualising distribution

sns.distplot(df2['Rented\_Bike\_Count'])



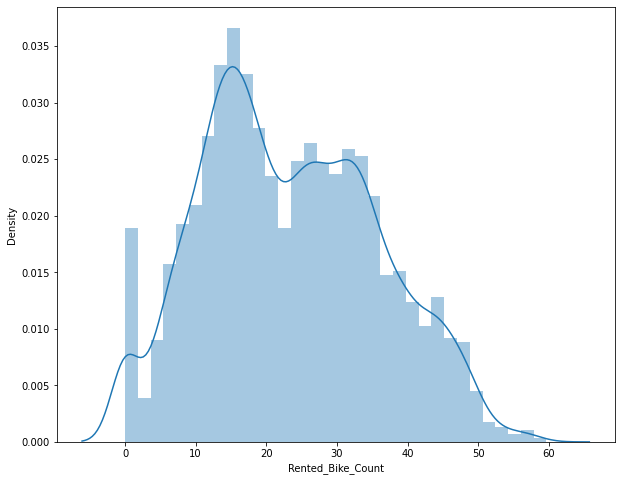
# Square Root Transformation

When you apply a square root transformation to a variable, high values get compressed and low values become more spread out. Log transformation does the same thing but more aggressively.

# square\_root transformation

plt.figure(figsize=(10,8))

sns.distplot(np.sqrt(df2['Rented\_Bike\_Count']))



#plotting histogram

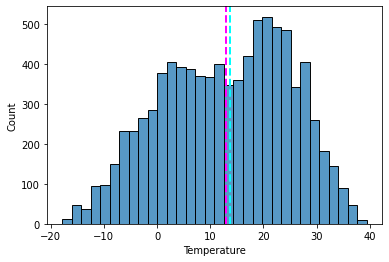
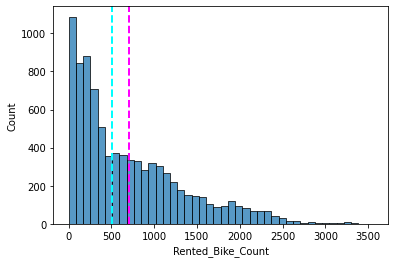
for col in numeric\_features[:]:

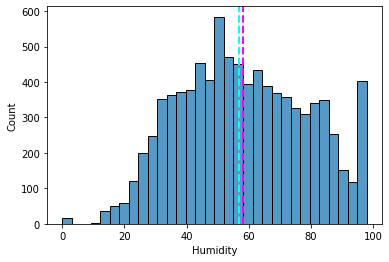
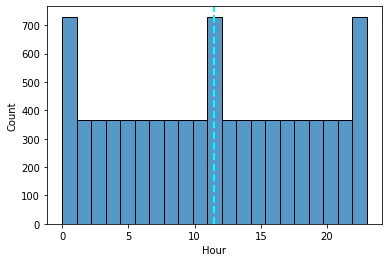
  sns.histplot(df2[col])

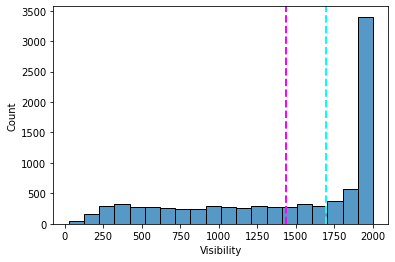
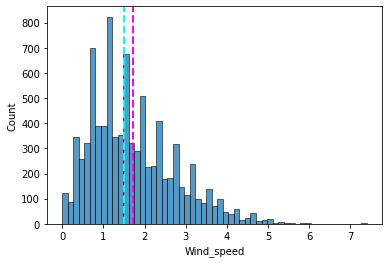
  plt.axvline(df2[col].mean(), color='magenta', linestyle='dashed', linewidth=2)

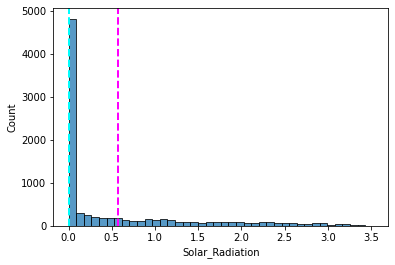
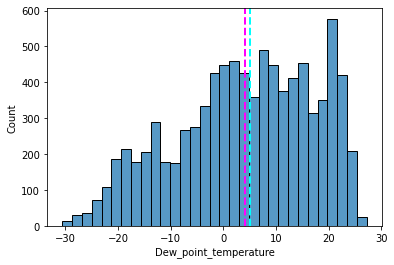
  plt.axvline(df2[col].median(), color='cyan', linestyle='dashed', linewidth=2)

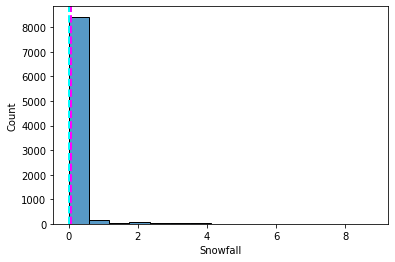
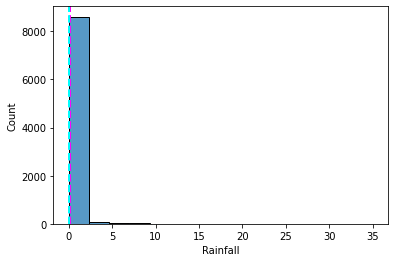
  plt.show()

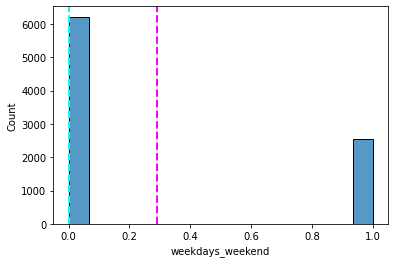
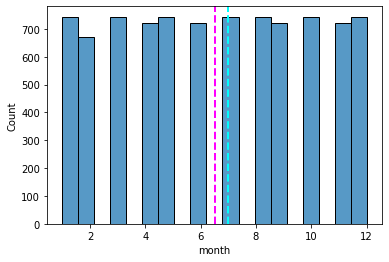












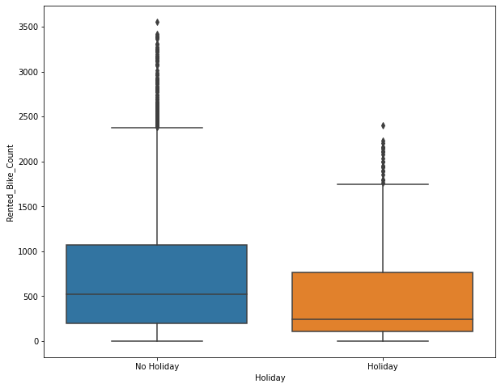
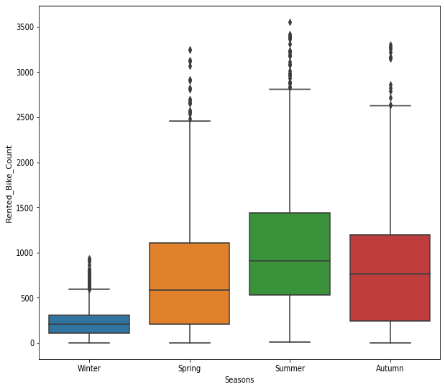
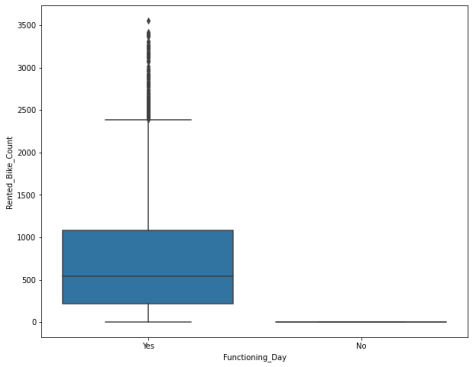
#ploting Box plot to visualize and trying to get information from plot

for col in categorical\_features:

  plt.figure(figsize=(10,8))

  sns.boxplot(x=df2[col],y=df2["Rented\_Bike\_Count"])

  plt.show()

Conclusion

Less demand on winter seasons

Sligthly Higher demand during Non holidays

Almost no demnad on non functioning day

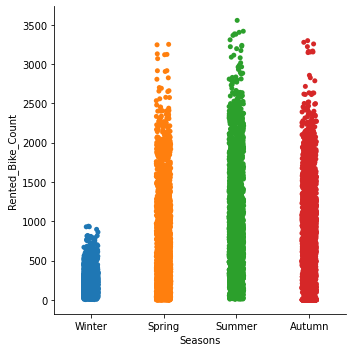
#checking counts of functioning day

df2['Functioning\_Day'].value\_counts()

#ploting cat plot for more info

here we describe the seasons on x-axis and rented bike count on y-axis

sns.catplot(x='Seasons',y='Rented\_Bike\_Count',data=df2)



Conclusion

we can clearly see that there  less demand of rented bike during winter season.

#ploting line graph

# group by Hrs and get average Bikes rented, and precent change

avg\_rent\_hrs = df2.groupby('Hour')['Rented\_Bike\_Count'].mean()

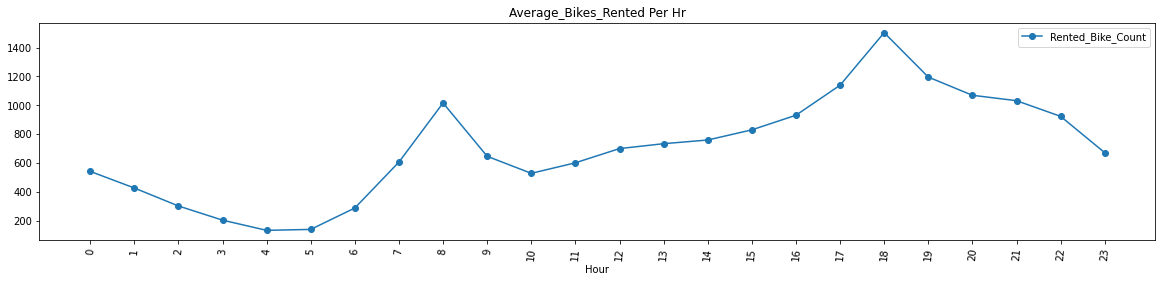
# plot average rent over time(hrs)

plt.figure(figsize=(20,4))

a=avg\_rent\_hrs.plot(legend=True,marker='o',title="Average\_Bikes\_Rented Per Hr")

a.set\_xticks(range(len(avg\_rent\_hrs)));

a.set\_xticklabels(avg\_rent\_hrs.index.tolist(), rotation=85);



Conclusion

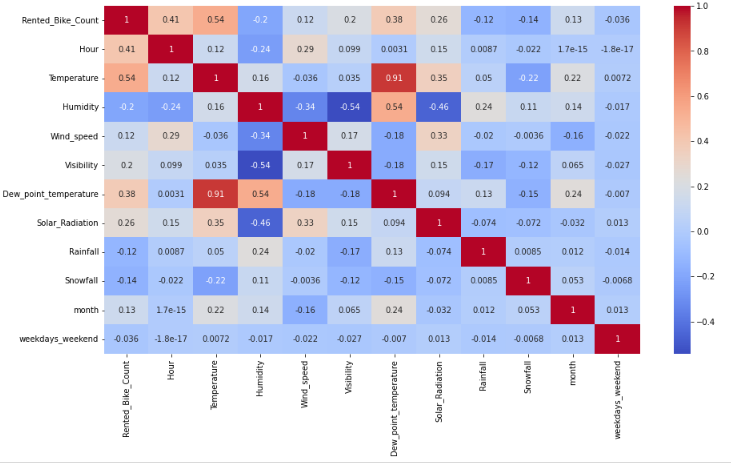
High rise of Rented Bikes from 8:00 a.m to 9:00 p.m means people prefer rented bike during rush hour. we can clearly see that demand rises most at 8 a.m and 6:00 p.m so we can say that that during office opening and closing time there is much high demand

**Heatmap**

Heatmap is a two dimensional representation of information with the help of colors. Heatmap can help the user to visualize simple or complex information. Heat maps are used in many area such as defence, marketing and understanding consumer behavior. Basically Heatmap is used to check the correlation between the dependent variables and independent variables.

From the Heatmap we can say that the relation between temperature and dew\_point\_temperature is very high i.e, 0.91.

So we can remove the dew\_point\_temperature columns from the dataset and it don't affect the outcome of our analysis.



**Categorical Encoding:**

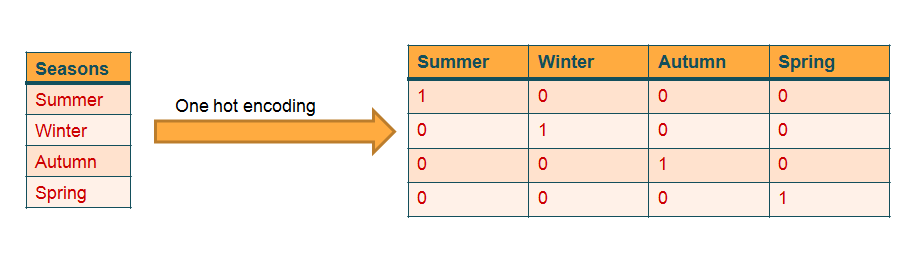
Encoding categorical data is a process of converting categorical data into integer format so that the data with converted categorical values can be provided to the models to give and improve the predictions.

There are two types of Categorical Encoding

1. One-Hot Encoding
2. Label Encoding
3. One-Hot Encoding:

One hot encoding is the most widespread approach, and it works very well unless your categorical variable takes on a large number of values (i.e. you generally won't it for variables taking more than 15 different values. It'd be a poor choice in some cases with fewer values, though that varies.)

One hot encoding creates new (binary) columns, indicating the presence of each possible value from the original data. Let's work through an example.



In our dataset, we had converted datatypes of categorical variables like seasons, Hours, etc to int datatypes. So now we can push that variables to our machine learning model and get our analysis.

## **Model Training:**

In model training our first step is to do a train test split of our data.

The train-test split procedure is used to estimate the performance of machine learning algorithms when they are used to make predictions on data not used to train the model.

It is a fast and easy procedure to perform, the results of which allow you to compare the performance of machine learning algorithms for your predictive modeling problem. Although simple to use and interpret, there are times when the procedure should not be used, such as when you have a small dataset and situations where additional configuration is required, such as when it is used for classification and the dataset is not balanced.

The Machine Learning algorithm we used in this project are as follows:

1. Linear Regression
2. Decision Tree
3. Random Forest
4. Elastic Net Regression

1.**Linear Regression:**

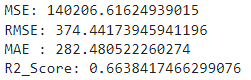
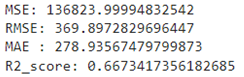
Linear Regression is a machine learning algorithm based on supervised learning. It performs a regression task. Regression models a target prediction value based on independent variables. It is mostly used for finding out the relationship between variables and forecasting.

We use Linear Regression model and evaluate our machine score which are as follows:

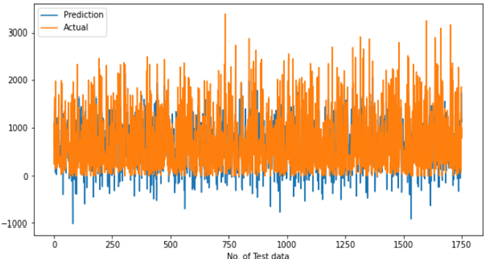
We calculate Mean Squared Error(MSE), Root Mean Squared Error(RMSE),

Mean Absolute Error(MAE) and R2\_Score of the Linear Regression.

Train Data Test Data

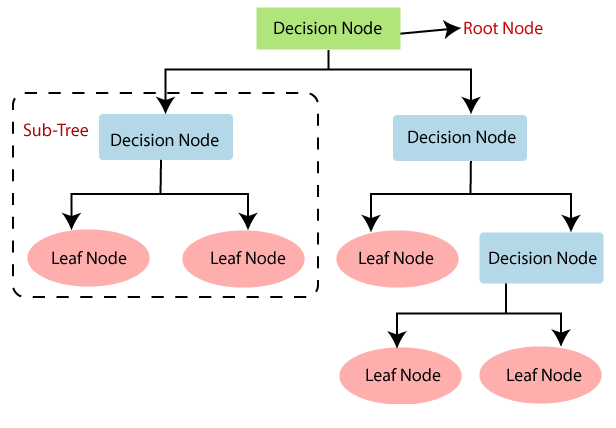
Analysis based on predicted value and actual value



1. **Decision Tree:**

Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.

In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.

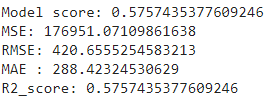
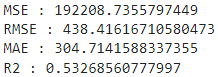


We use Elastic Net Regression model and evaluate our machine score which are as follows:

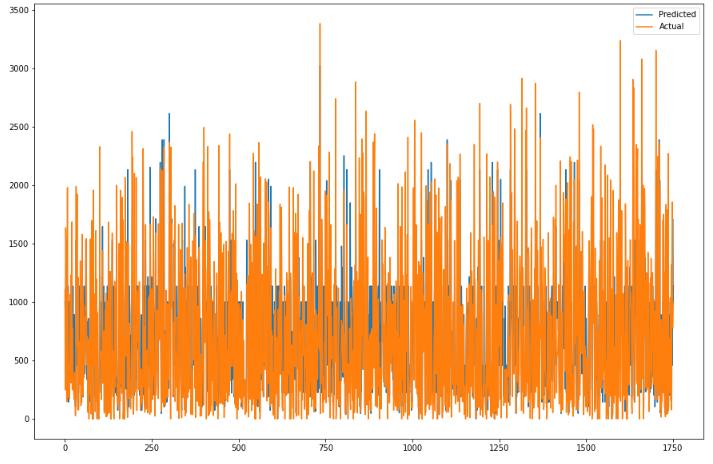
We calculate Mean Squared Error(MSE), Root Mean Squared Error(RMSE),

Mean Absolute Error(MAE) and R2\_Score of the Elastic Net Regression.

Train Data Test Data

Analysis based on predicted value and actual value

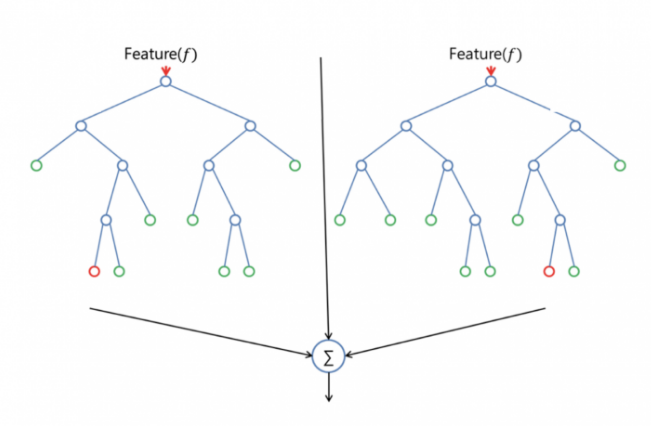
****

1. **Random Forest:**

Random forest is a Supervised Machine Learning Algorithm that is used widely in Classification and Regression problems. It builds decision trees on different samples and takes their majority vote for classification and average in case of regression

**Random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction.**

Random forest has nearly the same hyperparameters as a decision tree or a bagging classifier. Fortunately, there’s no need to combine a decision tree with a bagging classifier because you can easily use the classifier-class of random forest. With random forest, you can also deal with regression tasks by using the algorithm’s regressor.

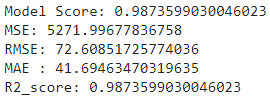
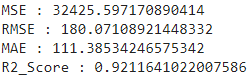


We use Random Forest model and evaluate our machine score which are as follows:

We calculate Mean Squared Error(MSE), Root Mean Squared Error(RMSE),

Mean Absolute Error(MAE) and R2\_Score of the Random Forest.

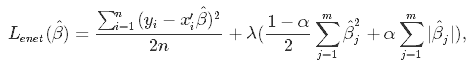
Train Data Test Data

1. **Elastic Net Regression:**

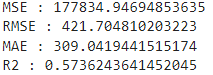
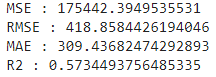
In statitics and, in particular, in the fitting of linear or logistics regression models, the elastic net is a regularized regression method that linearly combines the L1 and L2 penalties of the lasso and ridge methods.

Elastic Net first emerged as a result of critique on lasso, whose variable selection can be too dependent on data and thus unstable. The solution is to combine the penalties of ridge regression and lasso to get the best of both worlds. Elastic Net aims at minimizing the following loss function:



where α is the mixing parameter between ridge (α = 0) and lasso (α = 1).

Train Data Test Data

## **Conclusion:**

During the time of our analysis, we initially did EDA on all the features of our datset. We first analysed our dependent variable, 'Rented Bike Count' and also transformed it.Next we analysed categorical variable and dropped the variable who had majority of one class, we also analysed numerical variable, found out the correlation, distribution and their relationship with the dependent variable. We also removed some numerical features who had mostly 0 values and hot encoded the categorical variables.

We implemented 4 Machine Learning algorithms Linear Regression, Decision Tree, Random Forest and Elastic Net Regression.

Random Forest Regressor gives highest R2 Score of 98% for training set and 92% for testing set  Decision Tree gives the lowest R2 Score of 57% for training set and 53% for testing set