Capstone Project-2

Bike Sharing Demand Prediction

(Supervised Machine Learning regression)

BY

Kartike

Anupam Mishra

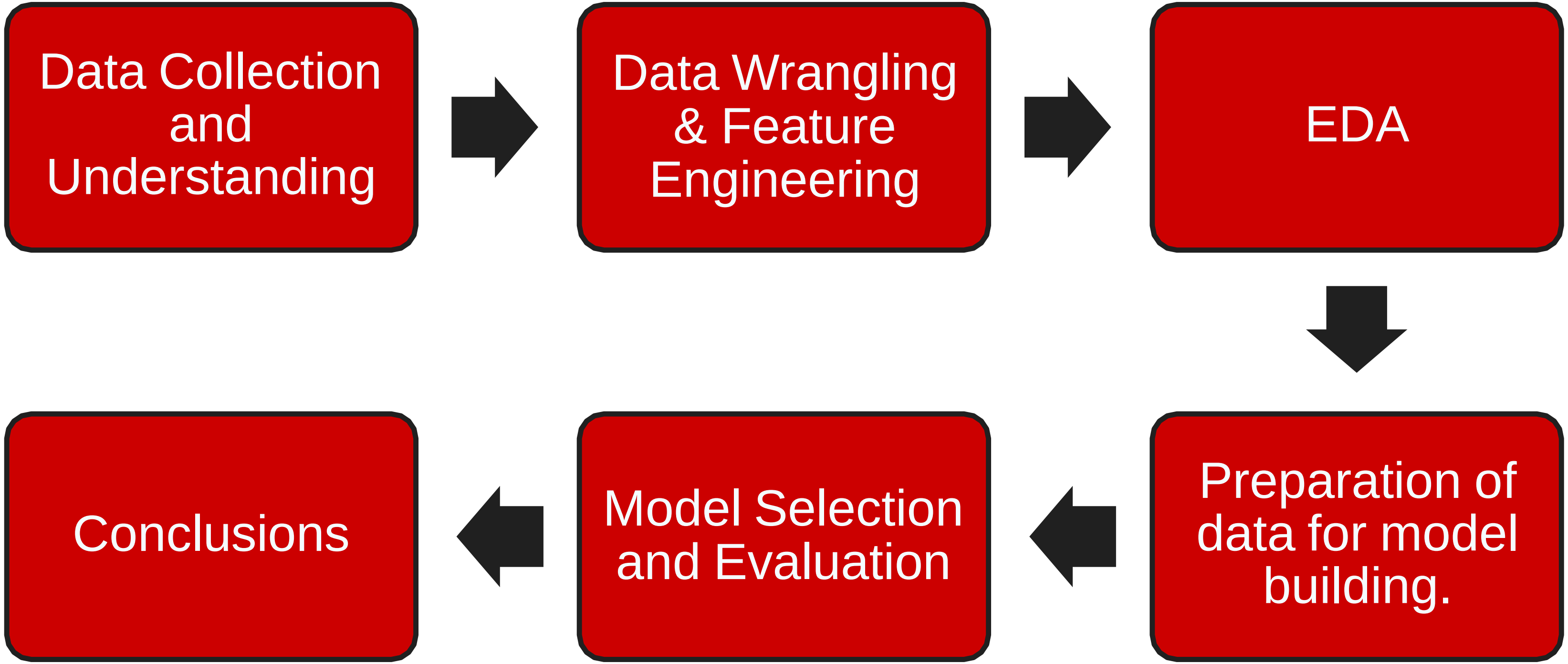
Animesh Chakraborty

Problem Statement:

- Currently Rental bikes are introduced in many urban cities for the enhancement of mobility comfort. The client is Seoul Bike, which participates in a bike share program in Seoul, South Korea. An accurate prediction of bike count is critical to the success of the Seoul bike share program. 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 final aim of this project is the prediction of bike count required at each hour for the stable supply of rental bikes.

Work Flow :

- So we will divide our work flow into following steps.



Data Collection and Understanding:

- We had a Seoul Bike Data for our analysis and model building
- The dataset contains weather information (Temperature, Humidity, Wind speed, Visibility, Dew point, Solar radiation, Snowfall, Rainfall), the number of bikes rented per hour and date information.
- In this we had total 8760 observations and 14 features including target variable.

Data Description:

Date : year-month-day.
Hour - Hour of he day.
Temperature-Temperature in Celsius.
Humidity - %. **Wind speed** - m/s. **Visibility** - m.
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).
Rented Bike count - Count of bikes rented at each hour (Target Variable i.e Y variable).

Data Wrangling and Feature Engineering:

As we know we had 8760 observations and 14 features.

- **Categorical Features:** Seasons, Holiday and Functioning day.
- **Numerical Columns:**

Date, Hour, Temperature, Humidity, Wind speed, Visibility, Dew point temperature, Solar radiation, Rainfall, Snowfall, Rented Bike count .

```
# Checking dummy and labeled columns
df.head()
```

	Rented Bike Count	Hour	Temperature(°C)	Wind speed (m/s)	Solar Radiation (MJ/m2)	Rainfall(mm)	Snowfall (cm)	Holiday	Functioning Day	Month	Weekdays_or_weekend	Spring	Summer	Autumn	Winter
0	254	0	-5.2	2.2	0.0	0.0	0.0	0	1	1	0	0	0	0	1
1	204	1	-5.5	0.8	0.0	0.0	0.0	0	1	1	0	0	0	0	1
2	173	2	-6.0	1.0	0.0	0.0	0.0	0	1	1	0	0	0	0	1
3	107	3	-6.2	0.9	0.0	0.0	0.0	0	1	1	0	0	0	0	1
4	78	4	-6.0	2.3	0.0	0.0	0.0	0	1	1	0	0	0	0	1

❖ Data Wrangling and Feature Engineering:

- We had zero null values in our dataset.
- Zero Duplicate entries found.
- We changed the data type of Date column from ‘object’ to ‘datetime64[ns]’. This was done for feature engineering.
- We Created two new columns with the help of Date column ‘Month’ and ‘Day’. Which were further used for EDA. And later we dropped Date column.

In [ ]:
```
# Change The datatype of Date columns to extract 'Month' ,'Day', "year". so  further we can analyze the Bike rentals with respect to year months and d
dataset['Date'] = dataset['Date'].astype('datetime64[ns]')
```

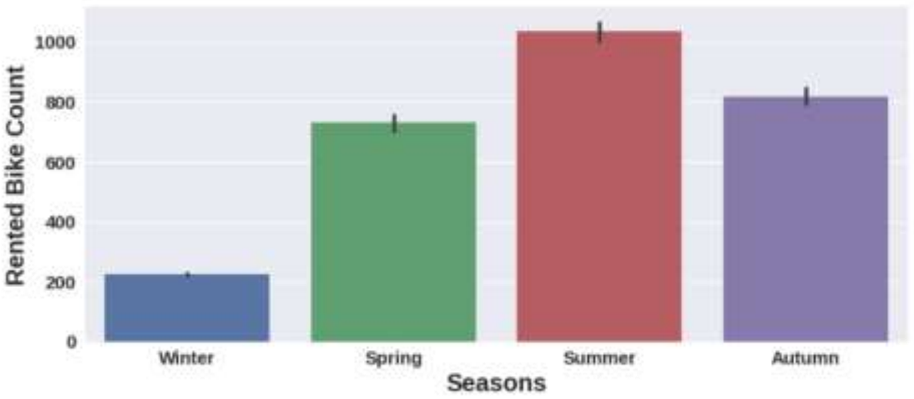
In [ ]:
```
# Creating new columns 'Month', 'Year' ,'Day.
dataset['Month'] = dataset['Date'].dt.month

dataset['Day'] = dataset['Date'].dt.day_name()
```

In [ ]:
```
# checking Duplicate rows in our BikeData.
duplicates = dataset.duplicated().sum()
print(f"We have {duplicates} duplicate rows in our Bike Data.")
```

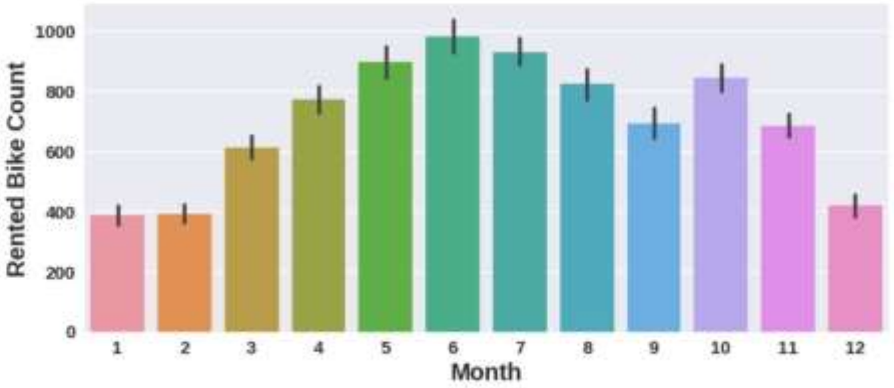
```
We have 0 duplicate rows in our Bike Data.
```

❖ EDA (Exploratory Data Analysis):



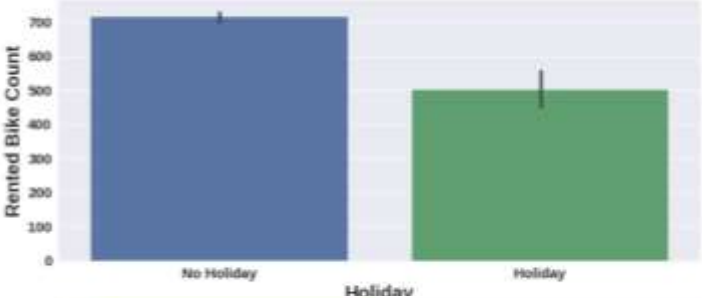
Relation of rented bike count with categorical features:

Summer season had the highest Bike Rent Count. People are more likely to take rented bikes in summer. Bike rentals in winter is very less compared to other seasons.



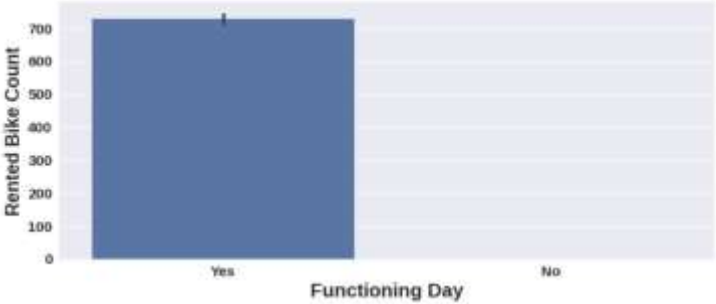
From March Bike Rent Count started increasing and it was highest in June.

❖ EDA (Exploratory Data Analysis):

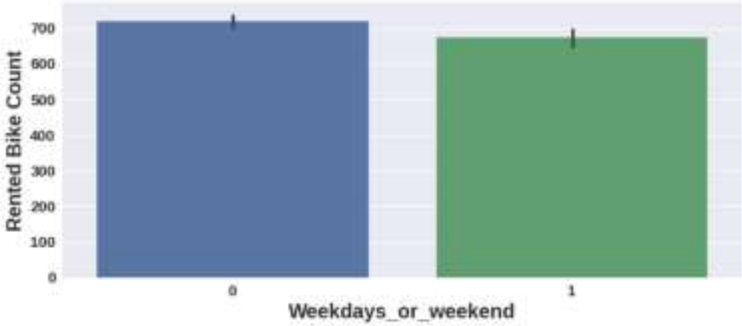


Conclusions:

High number of bikes were rented on No Holidays.
Which is almost 700 bikes.



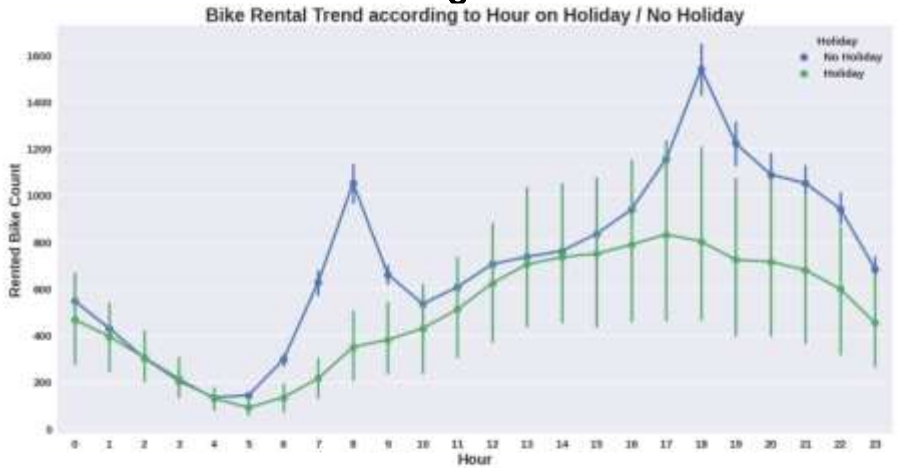
Zero Bikes were rented on no functioning day.
More than 700 bikes rented on functioning day.



More than 700 bikes were rented on weekdays.
On weekdays, almost 650 bikes were rented.

❖ EDA (Exploratory Data Analysis):

Bike Rent Trend according to hour in different scenarios.



Observations:

1)Here we observed that, Bike rental trend according to hours is almost similar in all scenarios.

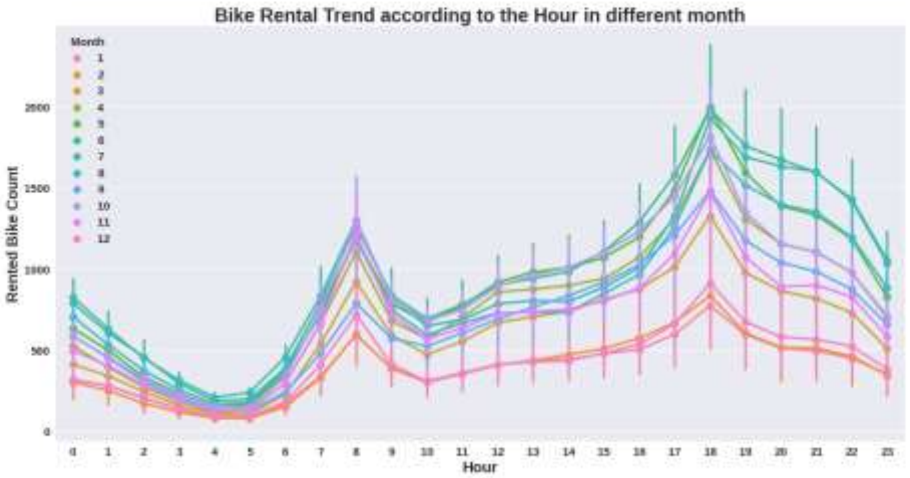
2)There is sudden peak between 6/7AM to 10 AM. Office /College going time could be the reason for this sudden peak on NO Holiday. But on Holiday the case is different,very less bike rentals happened.

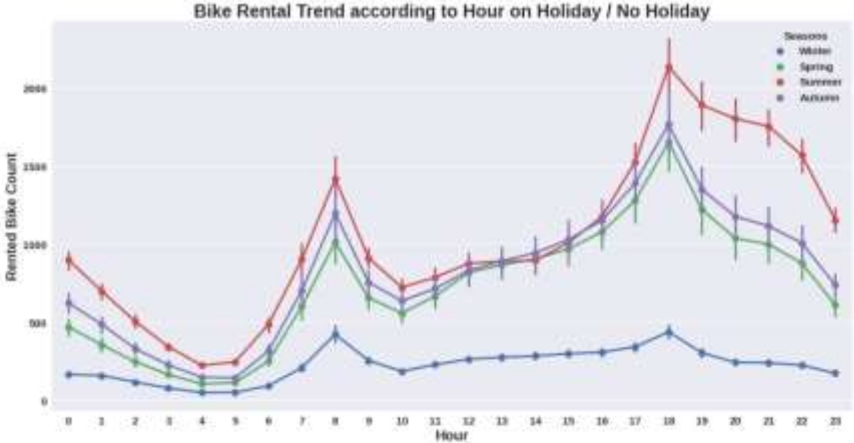
3)Again there is peak between 4PM to 7 PM. may be its office leaving time for the above people.(NO Holiday).

4)Here the trend for functioning day is same as of No holiday. Only the difference is on No functioning day there were zero bike rentals.

❖ EDA (Exploratory Data Analysis):

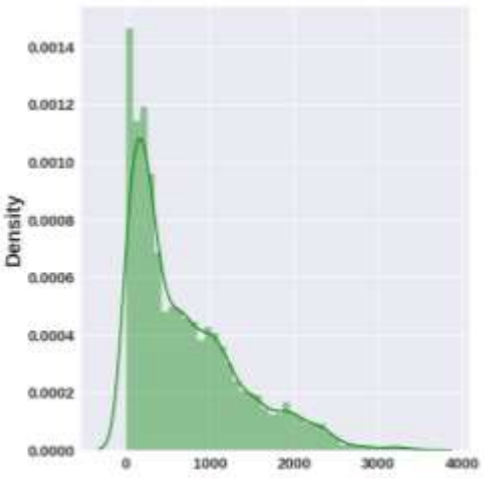
Bike Rent Trend according to hour in different scenarios.

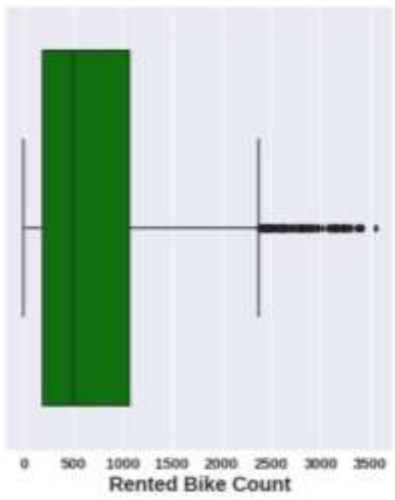


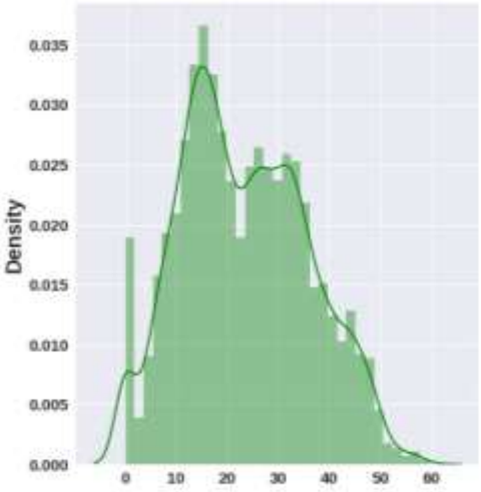


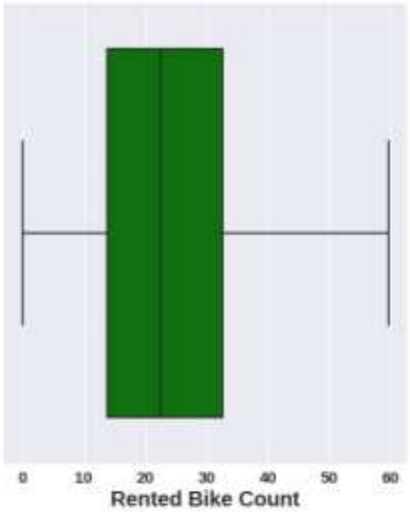
❖ EDA (Exploratory Data Analysis):

Distribution of target variable- Bike Rent Count





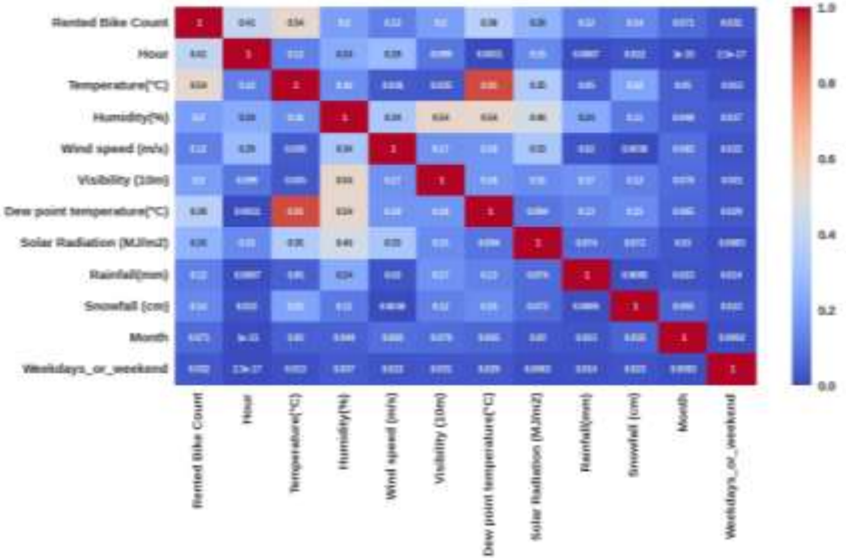




Distribution is rightly skewed and some outliers are observed.

To normalize the distribution we applied square root method. After normalization no outliers were found.

Preparation of data for model building:



- With the heat map we dropped highly correlated variables.
As we can see Temperature and Dew point temperature are 91 % correlated. So we dropped the Dew point temperature because it has very low correlation with our target variable as compared to temperature.

```
In [ ]:
# Create dummy variables for the catgeorical variable Season
df['Spring'] = np.where(df['Seasons'] == 'Spring', 1, 0)
df['Summer'] = np.where(df['Seasons'] == 'Summer', 1, 0)
df['Autumn'] = np.where(df['Seasons'] == 'Autumn', 1, 0)
df['Winter'] = np.where(df['Seasons'] == 'Winter', 1, 0)
```

```
In [ ]:
# Labeling for holiday=1 and no holiday=0
df['Holiday'] = df['Holiday'].map({'No Holiday':0, 'Holiday':1})
```

```
In [ ]:
# # Labeling for Yes=1 and no No=0
df['Functioning Day'] = df['Functioning Day'].map({'Yes':1, 'No':0})
```

- Later by using variation inflation factor we dropped 'Visibility' and 'Humidity' features as they had VIF value more than 5.
- Next we created dummy variables for categorical Seasons column and did mapping with 0 and 1 for holiday and functioning column.

- Thus we prepared our data for model building.

❖ Model Selection and Evaluation:

AI

As this is the regression problem we are trying to predict continuous value. For this we used following regression models.

- ➢Linear Regression
- ➢Lasso regression (regularized regression)
- ➢Ridge Regression(regularized regression)
- ➢Elastic Net Regression (regularized regression)
- ➢Decision Tree regression.
- ➢Random forest regression
- ➢Gradient Boosting regression.

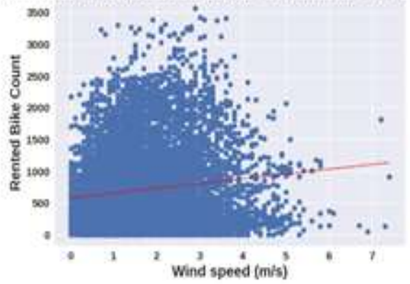
Assumptions of regression line:

1. The relation between the dependent and independent variables should be almostlinear.
2. Mean of residuals should be zero or close to 0 as much as possible. It is done to check whether our line is actually the line of “best fit”.
3. There should be homoscedasticity or equal variance in a regression model. This assumption means that the variance around the regression line is the same for all values of the predictor variable (X).
4. There should not be multicollinearity in regression model. Multicollinearity generally occurs when there are high correlations between two or more independent variables.

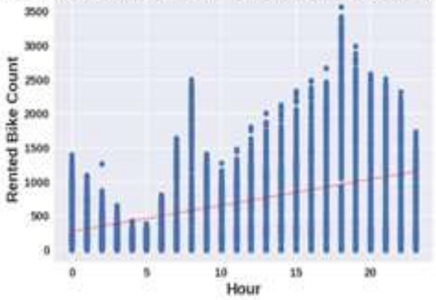
➢Before and after applying these models we checked our regression assumptions by distribution of residuals, scatter plot of actual and predicted values, removing multi-colinearity among independent variables.

❖ Model Selection and Evaluation :

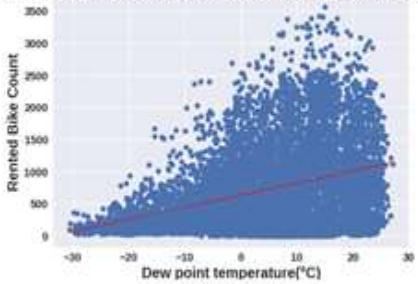
Rented Bike Count vs Wind speed (m/s)- correlation: 0.12110844818838669



Rented Bike Count vs Hour- correlation: 0.4102572913224858

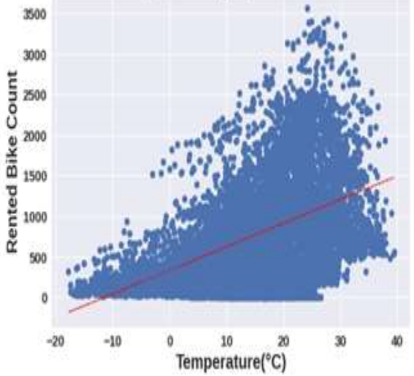


Rented Bike Count vs Dew point temperature(°C)- correlation: 0.37978812124497235

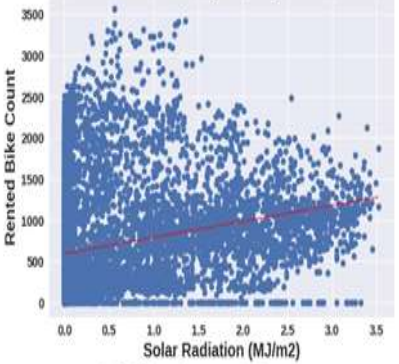


- From the above regression plot of all numerical features we see that the columns 'Temperature', 'Wind_speed', 'Visibility', 'Dew_point_temperature', 'Solar_Radiation' are positively relation to the target variable,which means the rented bike count increases with increase of these features.

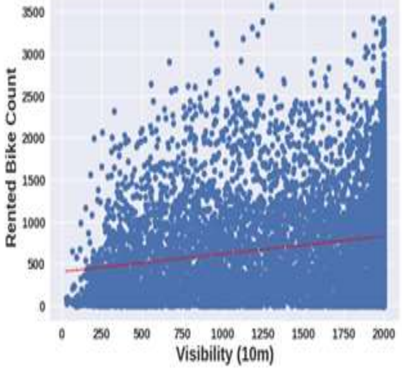
Rented Bike Count vs Temperature(°C)- correlation: 0.5385581530139789



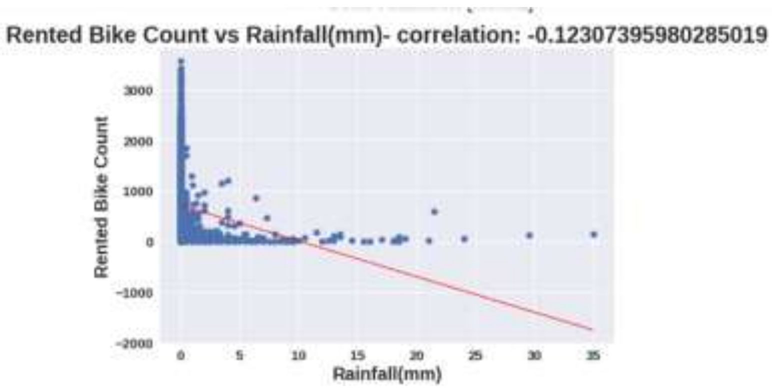
Rented Bike Count vs Solar Radiation (MJ/m2)- correlation: 0.261836985509591

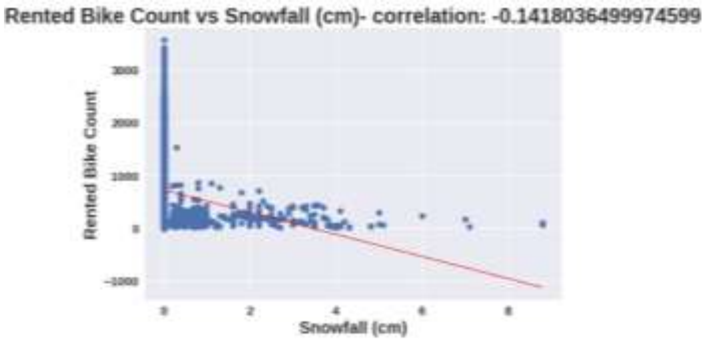


Rented Bike Count vs Visibility (10m)- correlation: 0.19928029673135897

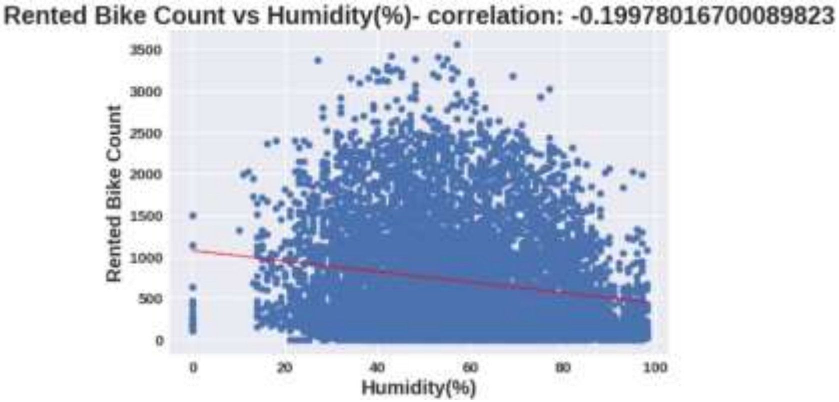


❖ Model Selection and Evaluation :





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



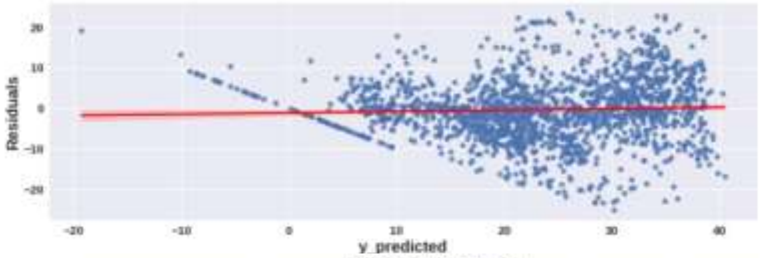
❖ Model Selection and Evaluation :

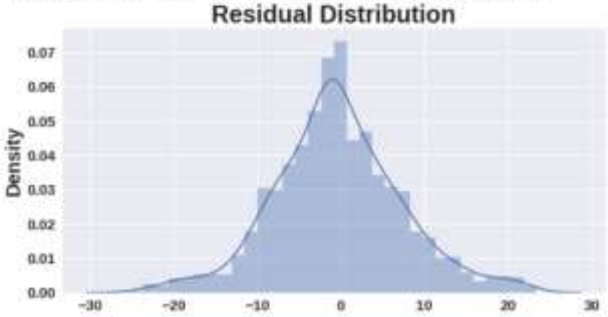
Linear regression, Lasso and Ridge Regression:

- **Linear Regression**

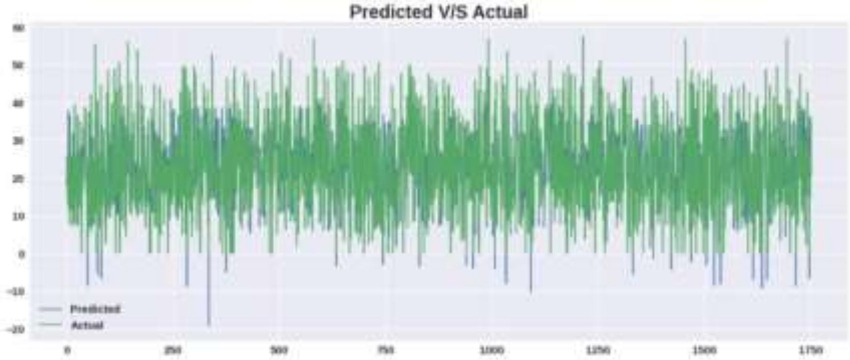
Scores on Test set

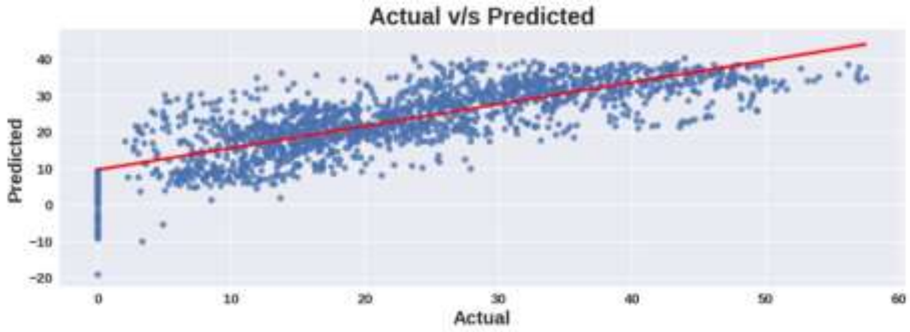
```
The Mean Absolute Error (MAE) is 5.906871386707488.
The Mean Squred Error(MSE) is 60.03961038562118.
The Root Mean Squared Error(RMSE) is 7.748523109962387.
The R2 Score is 0.6187628593015847.
Adjusted R2 is 0.6156901362332037.
```





Mean of residuals should be zero or close to 0 as much as possible. It is done to check whether our line is actually
the line of “best fit”



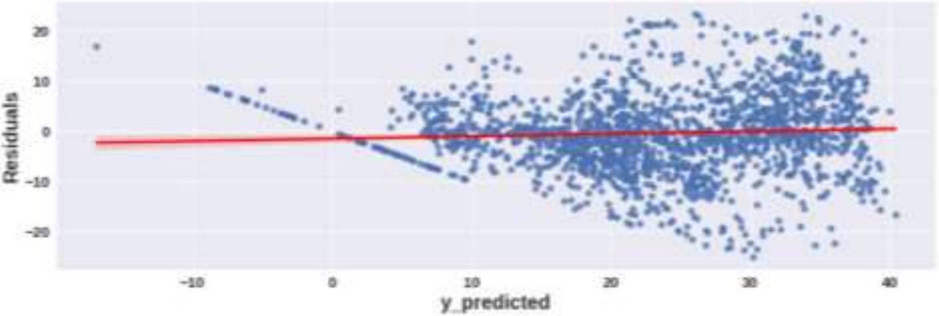


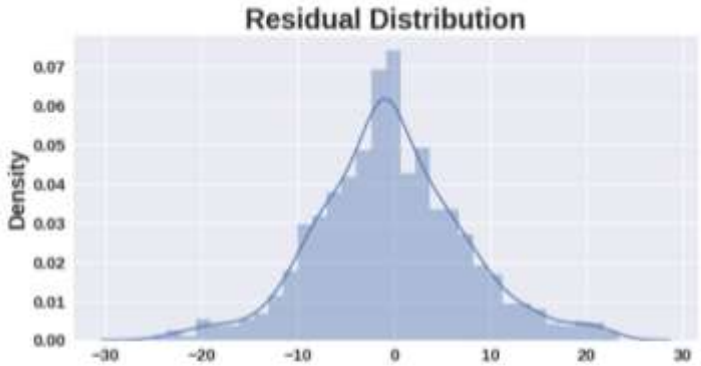
❖ Model Selection and Evaluation :

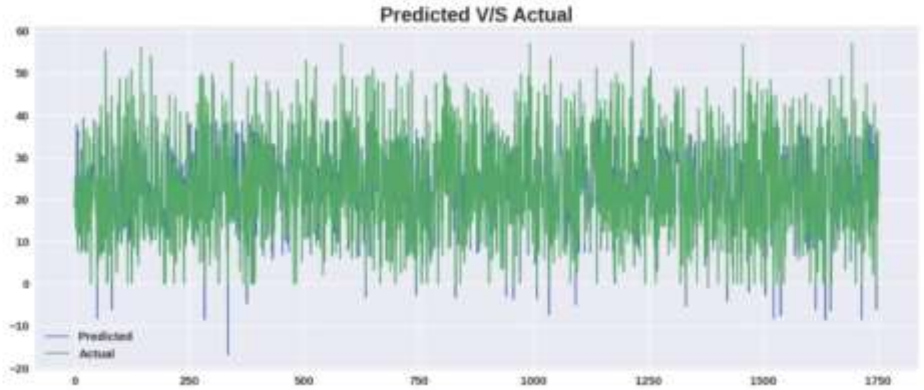
➢ **Lasso (Hyper-parameter tuned- alpha=0.01)**

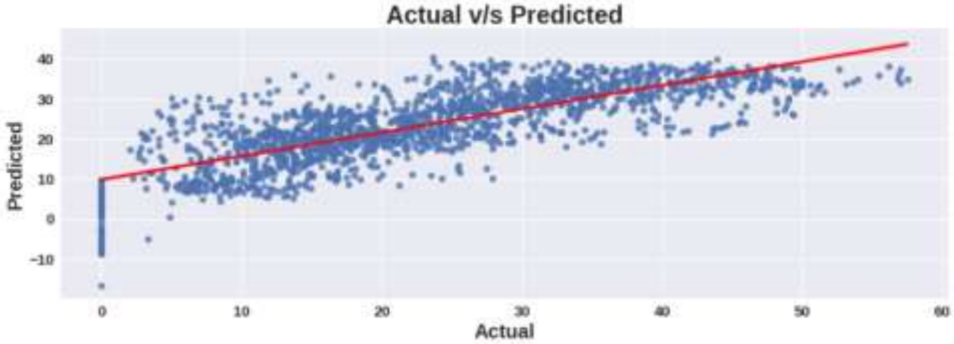
Scores on Test set

```
The Mean Absolute Error (MAE) is 5.92425618846209.
The Mean Squred Error(MSE) is 60.37268999136832.
The Root Mean Squared Error(RMSE) is 7.769986485919285.
The R2 Score is 0.6166478836096304.
Adjusted R2 is 0.6135581141050448.
```







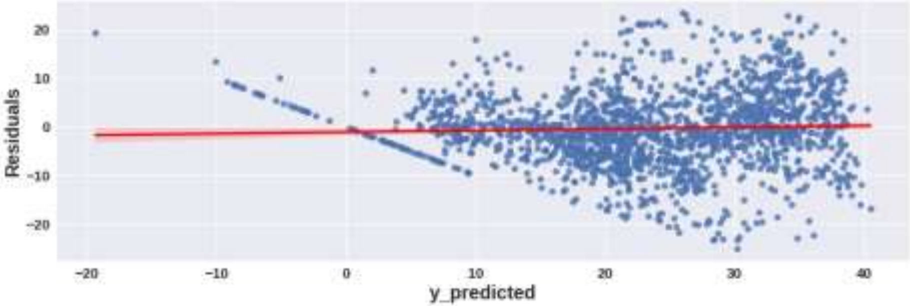


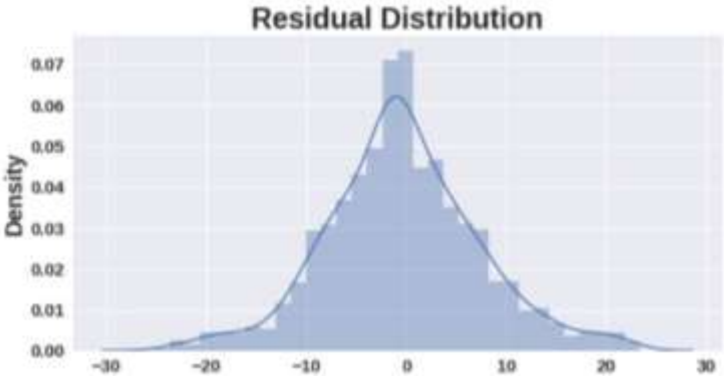
❖ Model Selection and Evaluation :

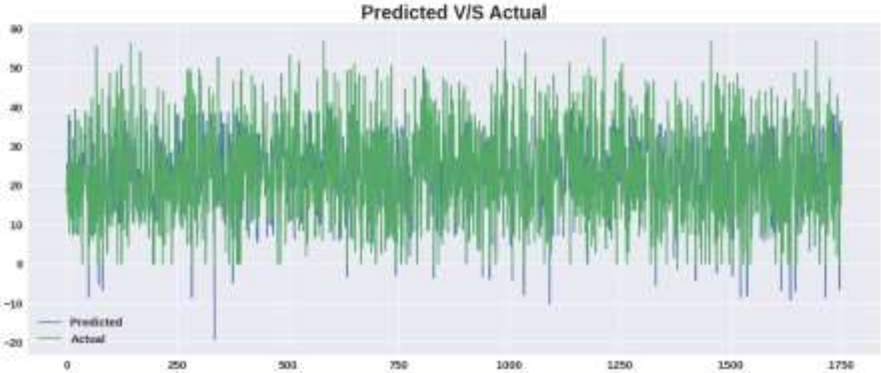
➢ **Ridge (Hyper-parameter tuned- alpha=0.01)**

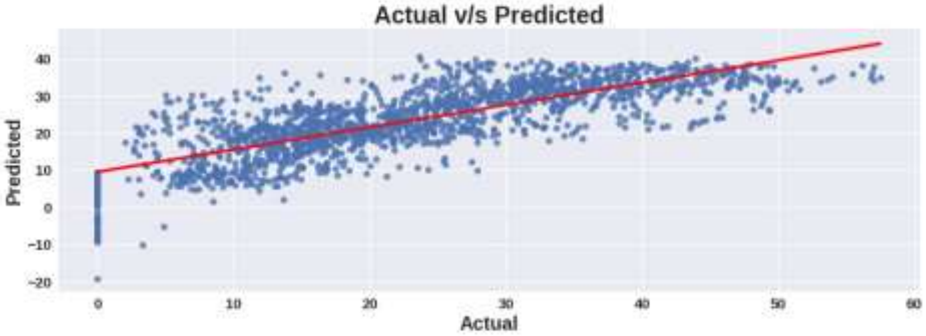
Scores on Test set

```
The Mean Absolute Error (MAE) is 5.908166202641087.
The Mean Squred Error(MSE) is 60.040522295485154.
The Root Mean Squared Error(RMSE) is 7.748581953847114.
The R2 Score is 0.6187570688924384.
Adjusted R2 is 0.6156842991540932.
```







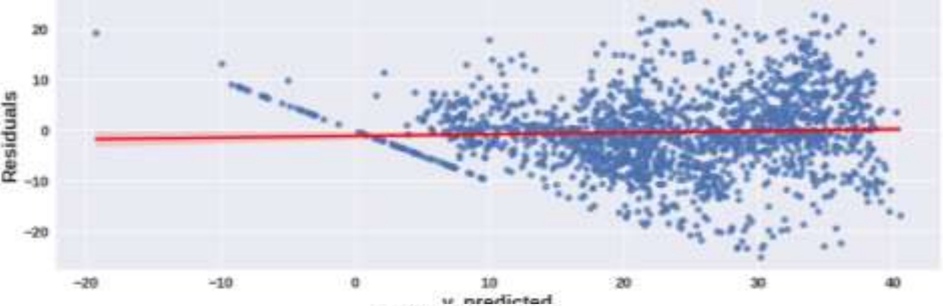


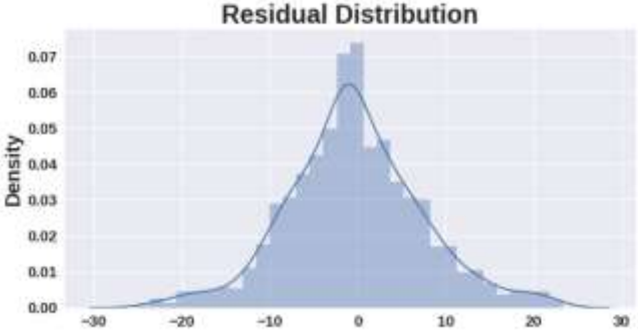
❖ Model Selection and Evaluation :

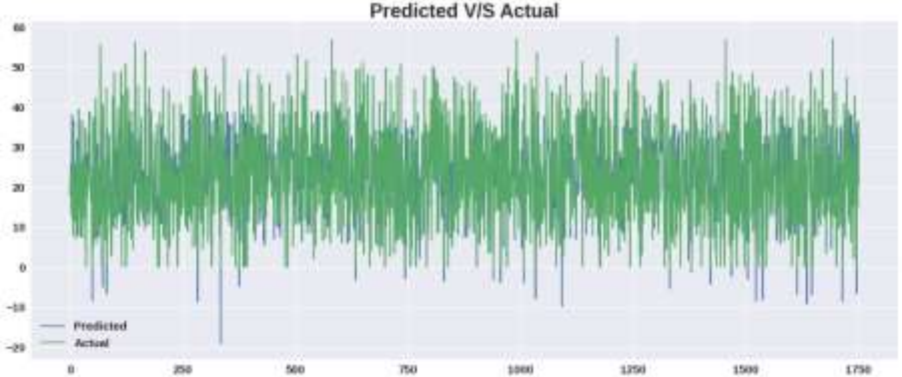
➢ **Elastic Net (Hyper-parameter tuned- alpha=1e-05,l1_ratio=0.3)**

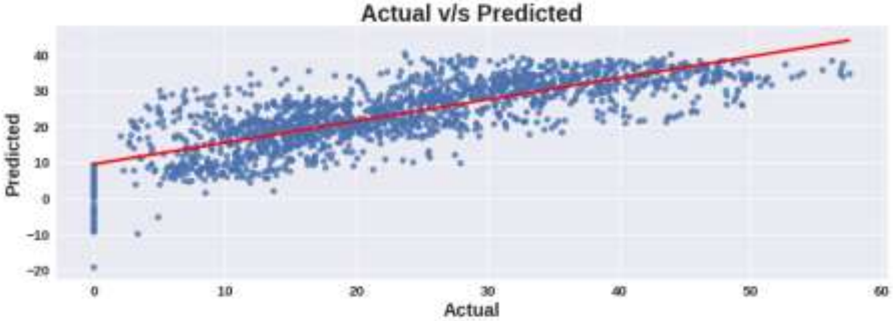
Scores on Test set

```
The Mean Absolute Error (MAE) is 5.9087303285240935.
The Mean Squred Error(MSE) is 60.04994705278263.
The Root Mean Squared Error(RMSE) is 7.749190090117975.
The R2 Score is 0.6186972239417335.
Adjusted R2 is 0.6156239718606652.
```









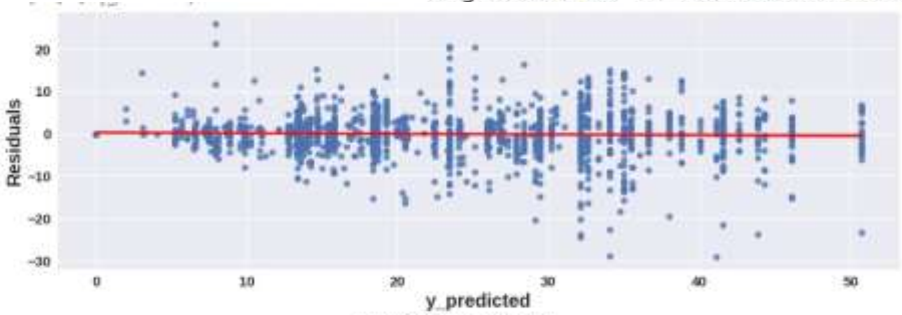
❖ Model Selection and Evaluation :

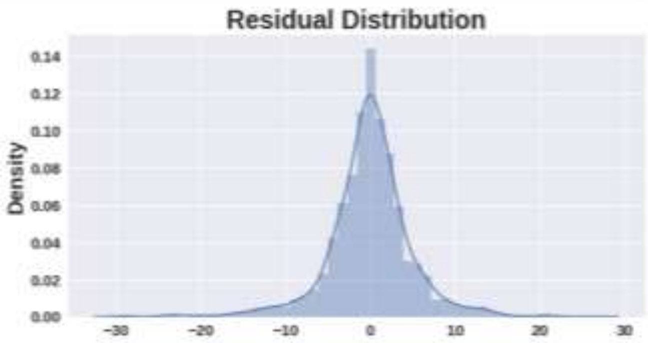
➢ **Decision Tree regression(Hyper-parameter tuned- max_depth=9,max_features='auto')**

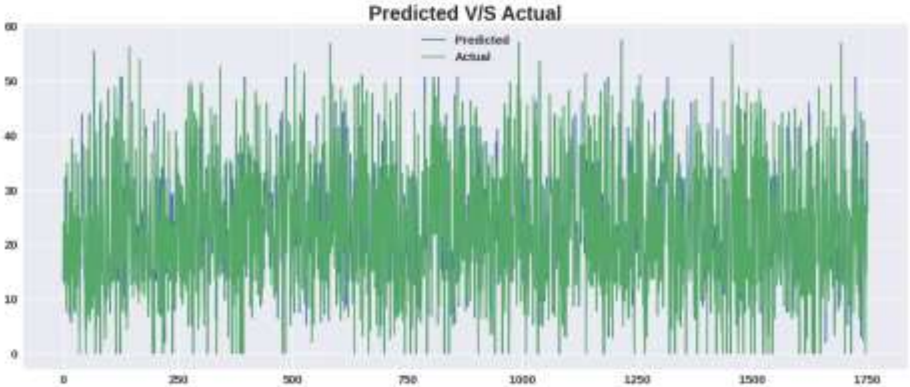
Scores on Test set

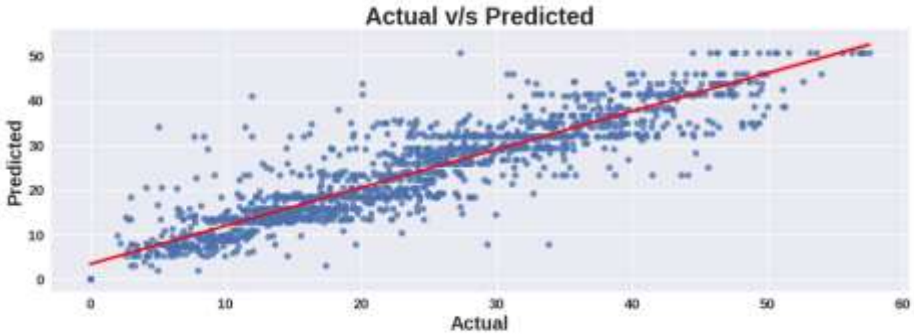
```
The Mean Absolute Error (MAE) is 3.355251849734944.
The Mean Squred Error(MSE) is 24.88273322917681.
The Root Mean Squared Error(RMSE) is 4.9882595390754085.
The R2 Score is 0.8420006057979907.
Adjusted R2 is 0.84072715069216.
```

The number of features to consider when looking for the best split







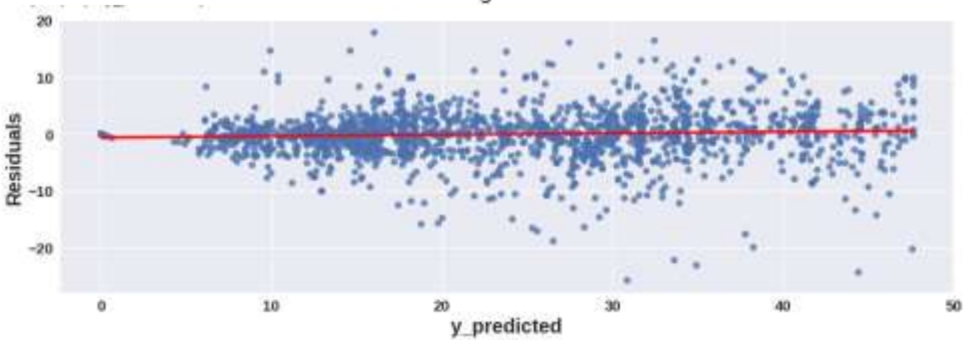


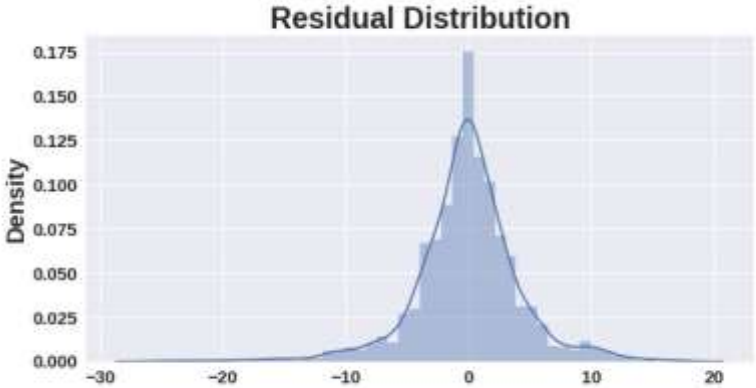
❖ Model Selection and Evaluation :

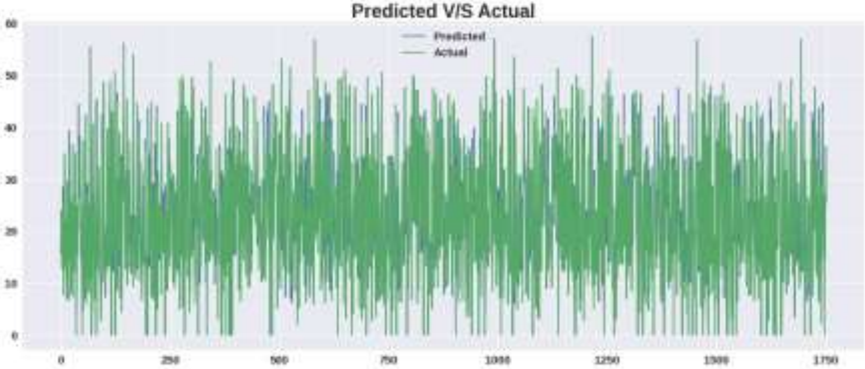
➢ **Random forest regression(Hyper-parameter tuned- 'max_depth': 9, 'n_estimators': 80')**

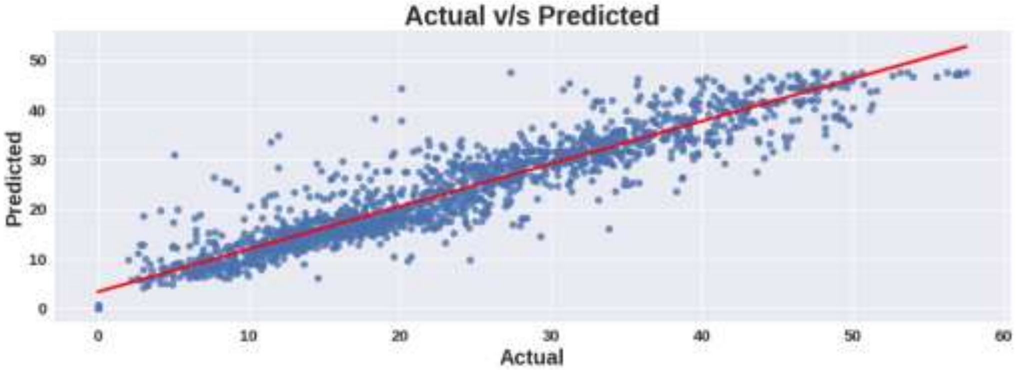
Scores on Test set

```
The Mean Absolute Error (MAE) is 2.433545438669562.
The Mean Squred Error(MSE) is 14.092426666121442.
The Root Mean Squared Error(RMSE) is 3.753988101489061.
The R2 Score is 0.9105164671591388.
Adjusted R2 is 0.9097952412179919.
```









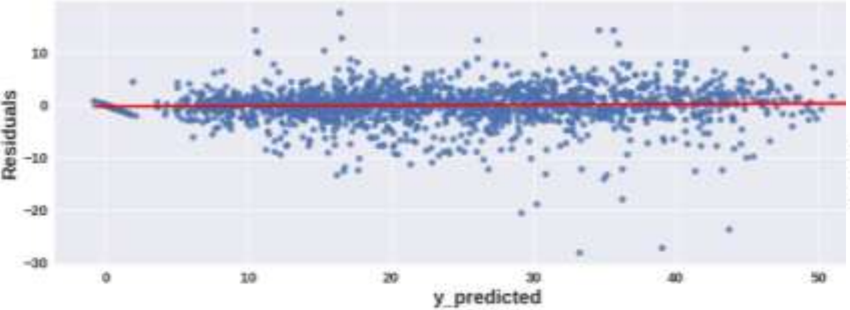
❖ Model Selection and Evaluation :

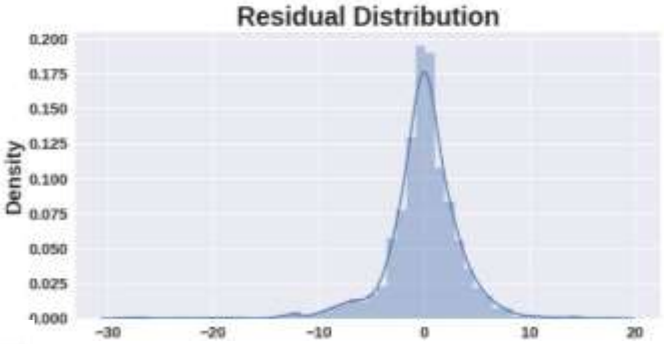
AI

➢Gradient boosting regression(Hyper-parameter tuned- 'learning_rate': 0.04, 'max_depth': 10, 'n_estimators': 150, 'subsample': 0.5)

Scores on Test set

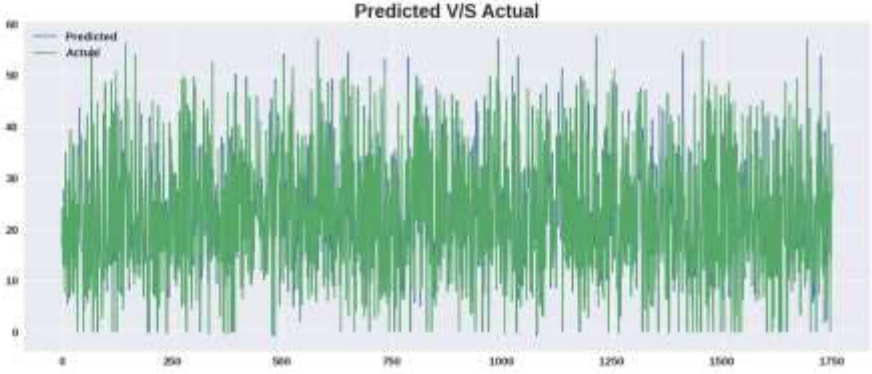
```
The Mean Absolute Error (MAE) is 2.323065845346332.
The Mean Squred Error(MSE) is 12.694823744207726.
The Root Mean Squared Error(RMSE) is 3.562979616024729.
The R2 Score is 0.9193909108532262.
Adjusted R2 is 0.9187412118042597.
```

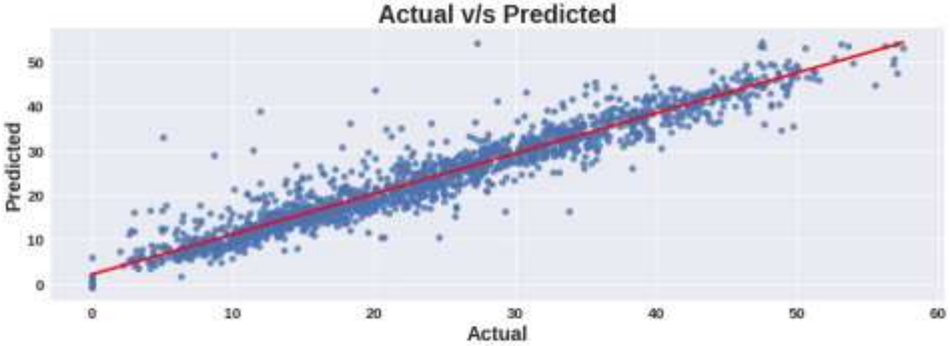




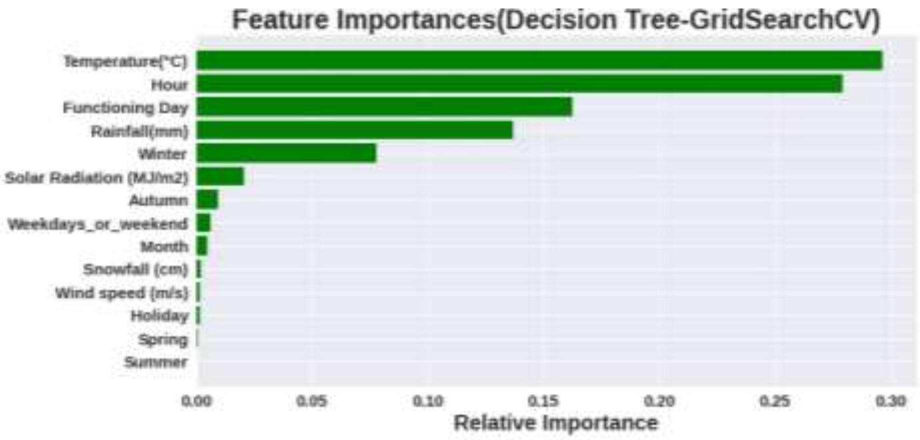
➢Learning rate shrinks the contribution of each tree by learning_rate. There is a trade-off between learning_rate and n_estimators.

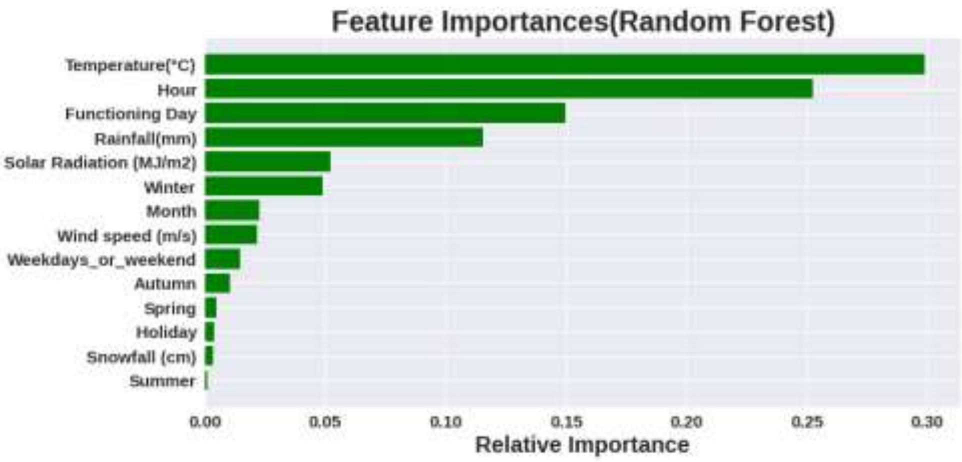
➢Choosing subsample < 1.0 leads to a reduction of variance and an increase in

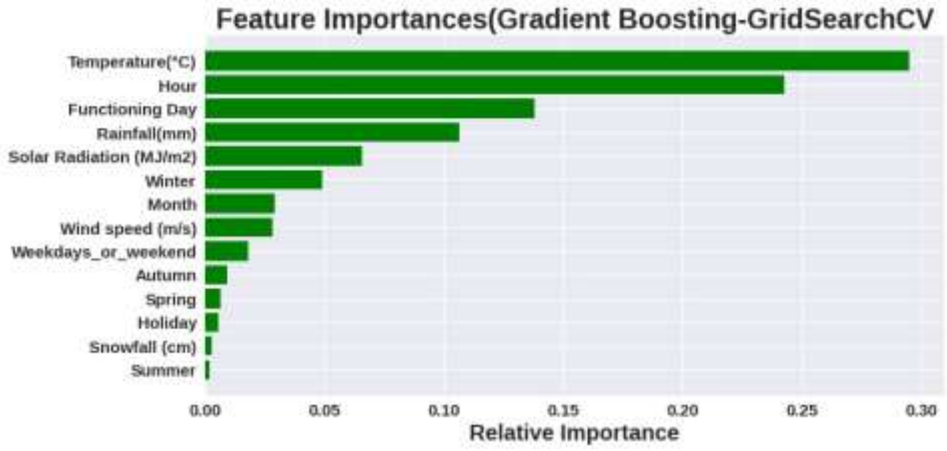




Feature importance's :







From all 3 models we can say that temperature, hour, functioning day are the top three important features.

❖ Conclusion:

	Model	MAE	MSE	RMSE	R2_score	Adjusted_R2
0	Linear Regression	5.9069	60.0396	7.7485	0.6188	0.6157
1	Lasso	5.9243	60.3727	7.7700	0.6166	0.6136
2	Ridge(GridsearchCv Tunned)	5.9082	60.0405	7.7486	0.6188	0.6157
3	ElasticNet(GridSearchCV-Tunned)	5.9087	60.0499	7.7492	0.6187	0.6156
4	Decision Tree Regressor(GridsearchCV)	3.3553	24.8827	4.9883	0.8420	0.8407
5	Random Forest Regressor	2.4335	14.0924	3.7540	0.9105	0.9098
6	Random Forest Regressor(GridsearchCV)	2.9444	18.7717	4.3326	0.8808	0.8798
7	Gardient boosting Regression	3.2419	21.5640	4.6437	0.8631	0.8620
8	Gradient Boosting Regression(GridSearchCV)	2.3231	12.6948	3.5630	0.9194	0.9187

As we have calculated MAE, MSE, RMS, R2_Score and Adjusted_R2 score for each model. Based on Adjusted_R2 score will decide our model performance

Linear, Lasso, Ridge and Elastic Net:

From the above data frame, we can see that linear, Lasso, Ridge and Elastic regression models have almost similar Adjusted_R2 scores 61% (approx.) on test data.(Even after using GridserachCV we have got similar results as of base models).

Decision Tree Regression:

After hyperparameter tuning we got Adjusted_R2 score as 84% (approx.) on test data which is quite good for us.

Random Forest:

On Random Forest regressor model, without hyperparameter tuning we got Adjusted_R2 as 90% (approx.) on test data. Thus our model memorized the data. So it was a overfitted model, as per our assumption.

After hyperparameter tuning we got Adjusted_R2 score as 87% (approx.) on test data which is very good for us.

Gradient Boosting Regression(Gradient Boosting Machine):

On Random Forest regressor model, without hyperparameter tuning we got Adjusted_R2 as 86% (approx.) on test data. Our model performed well without hyperparameter tuning.

After hyperparameter tuning we got Adjusted_R2 score as 91% (approx.) on test data, thus we improved the model performance by hyperparameter tuning.

Conclusion:

Thus Gradient Boosting Regression(GridSearchCV) and Random forest(GridSearchCv) gives good adjusted r2 scores. We can deploy any one of these models according to our requirement.

Challenges:

- A huge amount of data needed to be deal while doing the project which is quite an important task and also even small inferences need to be kept in mind.

- As dataset was quite big enough which led more computation time.

- Handling the numerical and categorical data to build high accuracy model.

Signing off…

THANK YOU