Capstone Project-2

Bike Sharing Demand Prediction

CONTENT

1. Problem Statement
2. Introduction
3. Steps Involved
4. Algorihtms
5. Model Performance
6. Conclusion

Problem Statement:

- To create a prediction model that can be used to anticipate the number of bike rentals throughout the year based on weather conditions.
- As a result, it will be easier to predict fast and accurately.

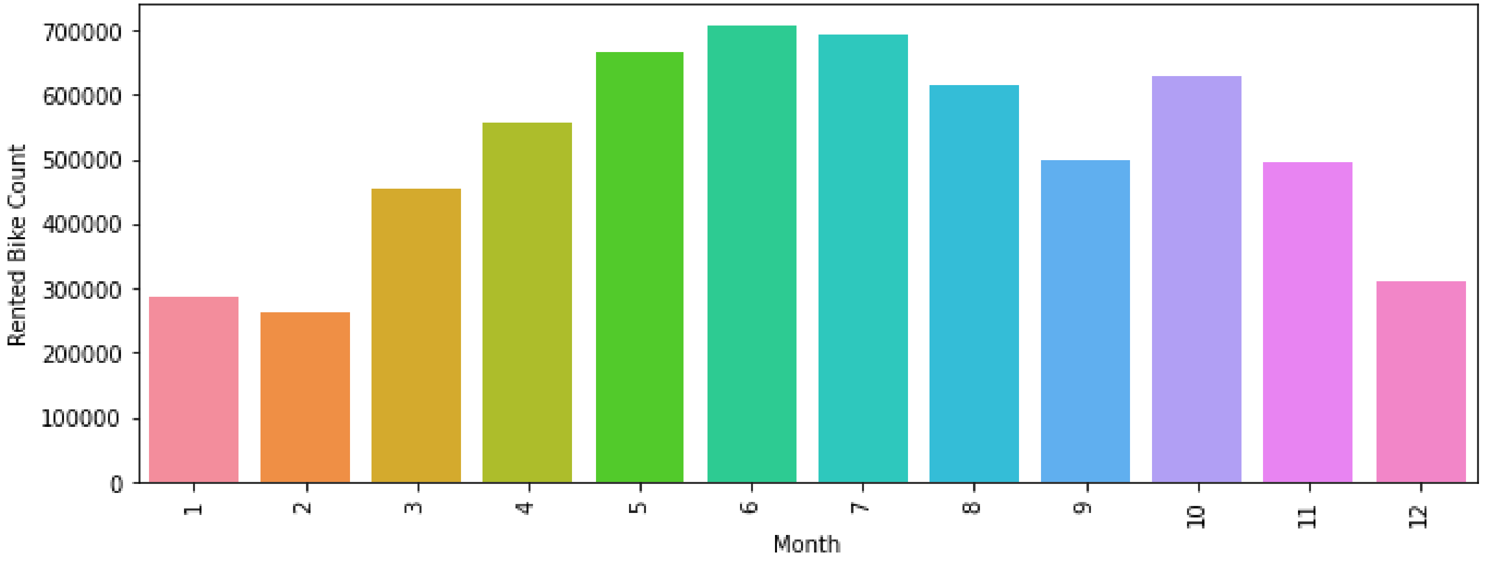
Introduction

- The main goal is to create a prediction model that can be used to anticipate the number of bike rentals each hour based on weather conditions. As a result, it would be easier to anticipate fast and accurately.
- Exploratory Data Analysis is performed on a dataset to determine the graph distribution by comparing the target variable to the other variables.
- We search for outliers and null values that were not discovered. We also use correlation analysis to pick the most critical and relevant features from the dataset, and then train the model using train test split.
- The goal of this project is to combine the historical bike usage patterns with the weather data to forecast bike rental demand. The data set consists of hourly rental data spanning two years.

Steps Involved:

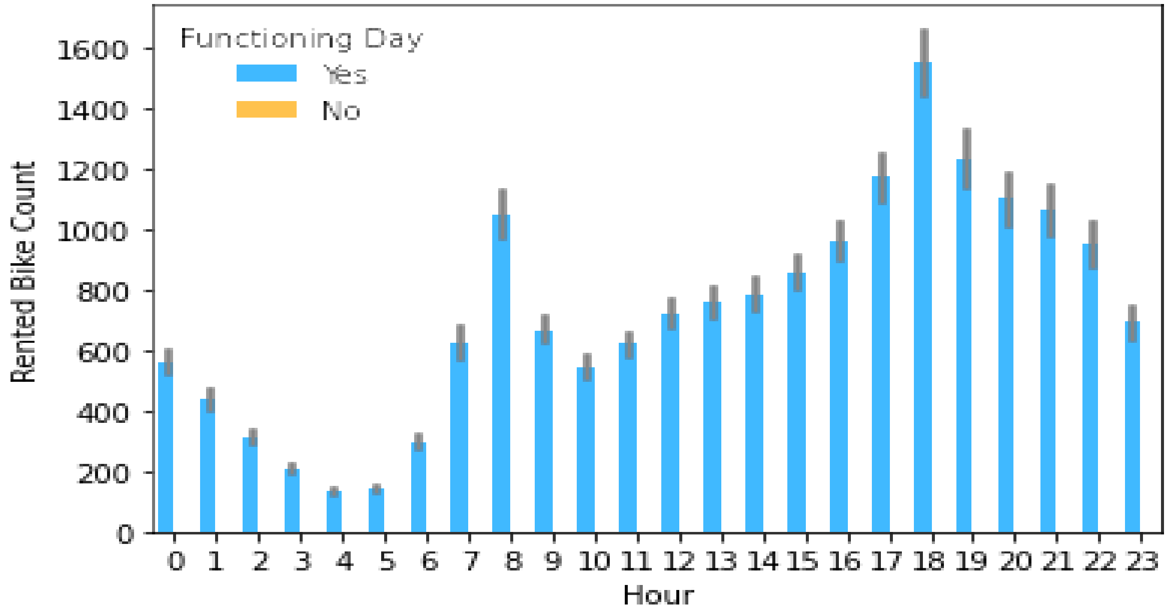
- Exploratory Data Analysis
- Null values Treatment and Outliers
- Numerical and categorical Features
- Label encoding
- Correlation Analysis
- Train test Split
- Fitting different models
- a) Linear Regression
- b)Polynomial Regression
- c)Random Forest Regressor
- Tuning the hyperparameters for better accuracy

Analysing in which Month the Rented Bike Count was maximum:



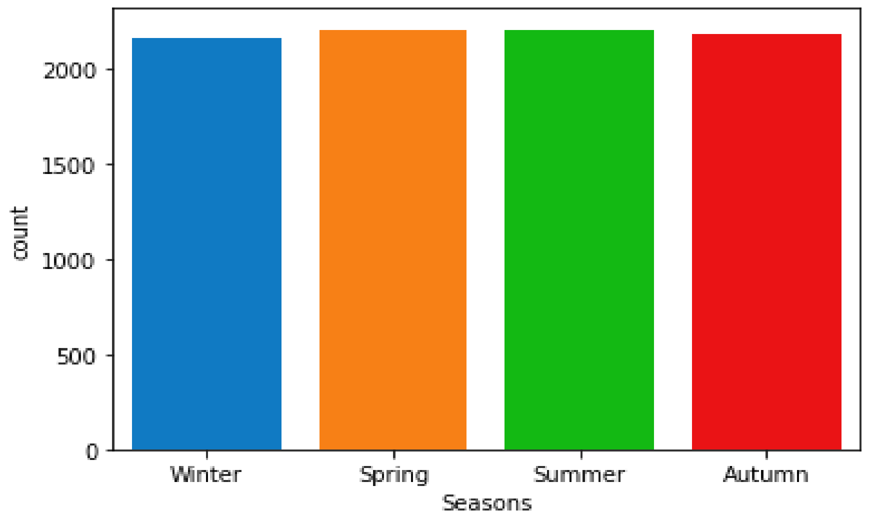
Month 6 i.e June the Rented Bike Count was maximum.

Analysing at which hour the Rented Bike countis maximum w.r.t. Functional day



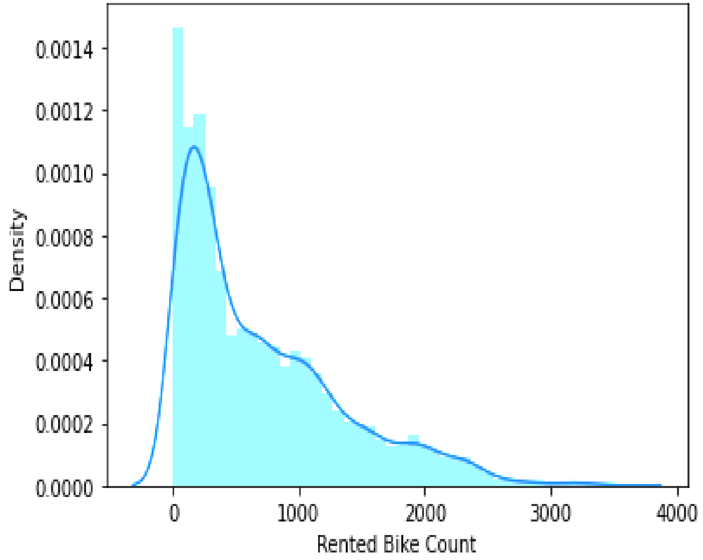
At 18th Hour i.e. 6 P.M. the Rented bike count is maximum.

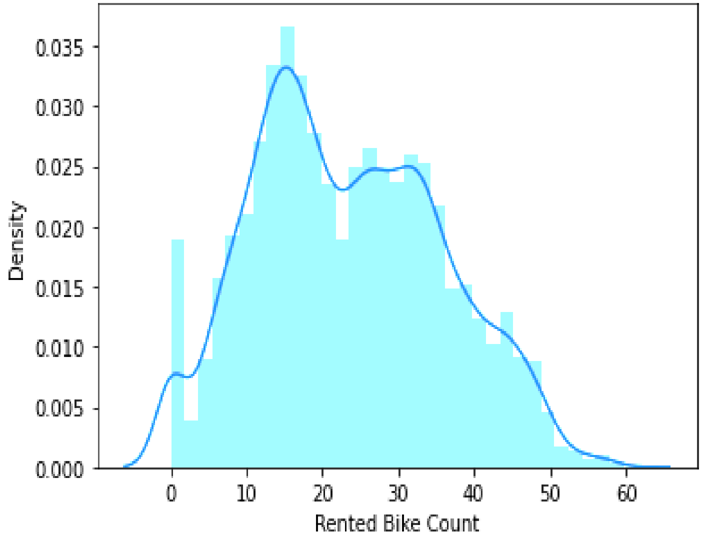
Analysing Count of Rented Bikes for different seasons.



Rented bikes count is max. in spring and in summer season.
i.e.Spring season= 2208
Summer season=2208
Autumn season= 2184
Winter season= 2160

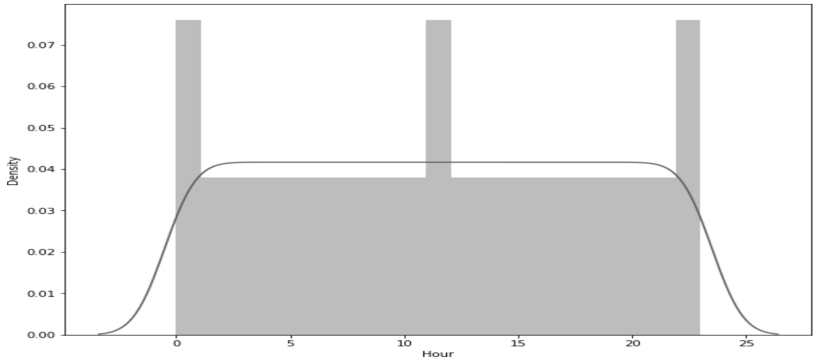
Distribution of dependent variables

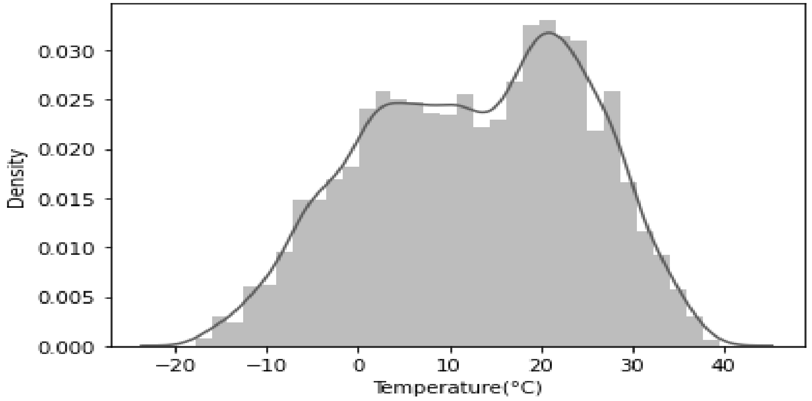


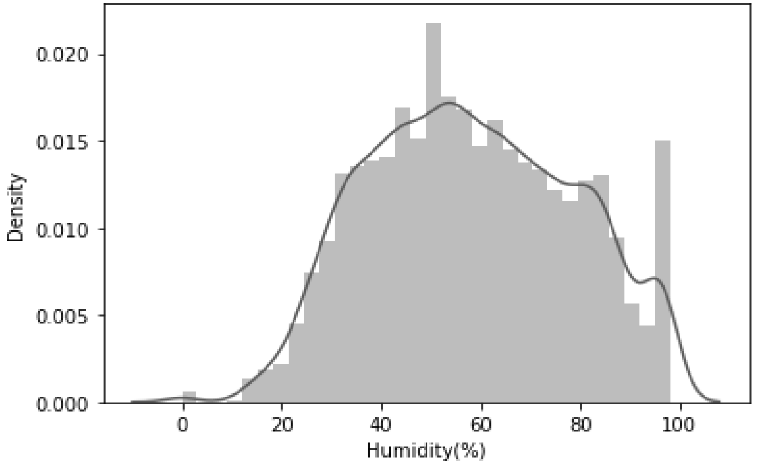


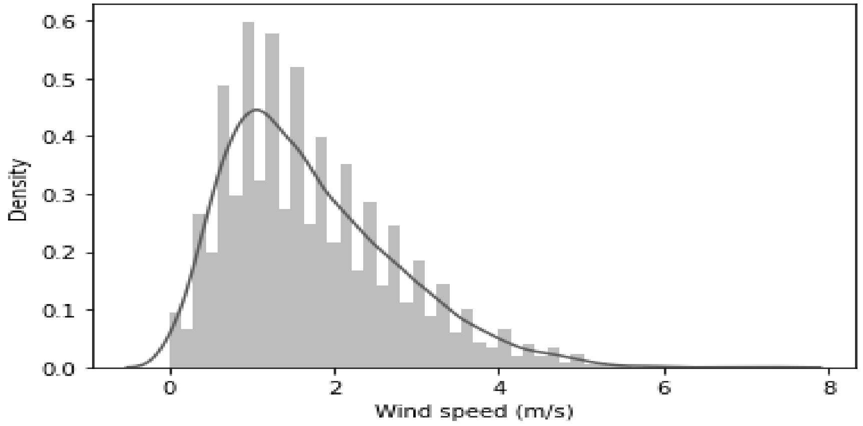
As it was right skewed, so wehave taken square root of dependent variable to visualize it in a better way...

Distribution of independent variables

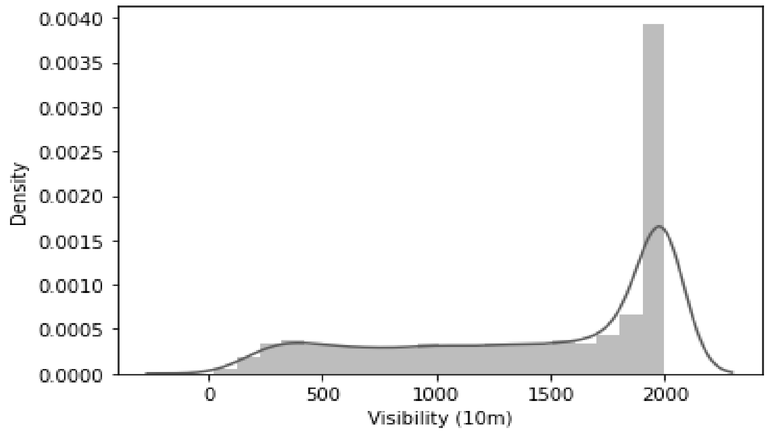


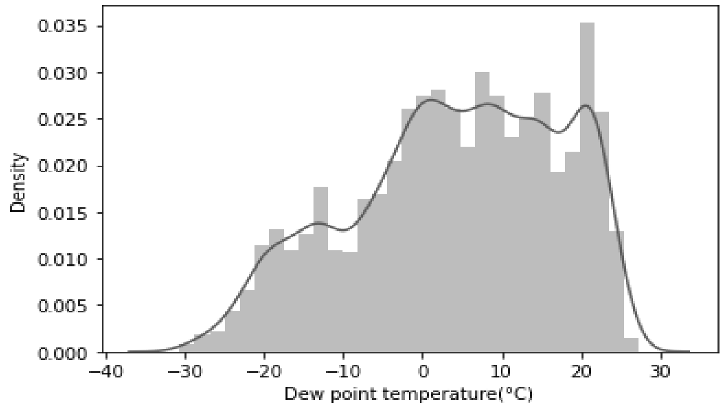


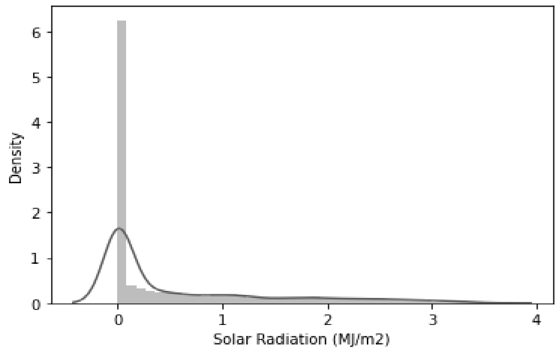


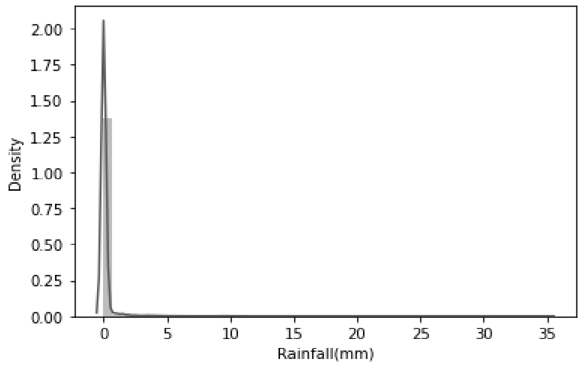


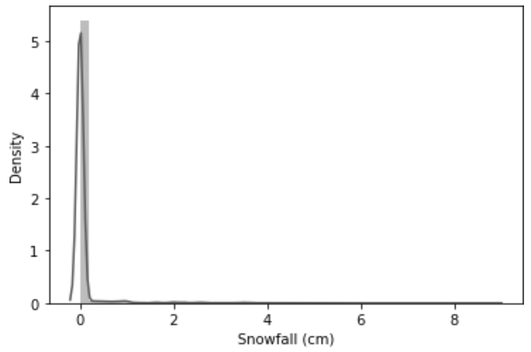
Distribution of independent variables











Skewed Data

Right Skewed Data:

- Wind speed (m/s)
- Solar Radiation (MJ/m2)
- Rainfall(mm)
- Snowfall (cm)

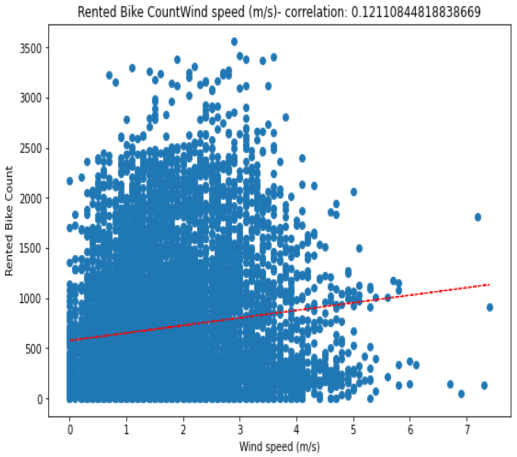
Left Skewed Data:

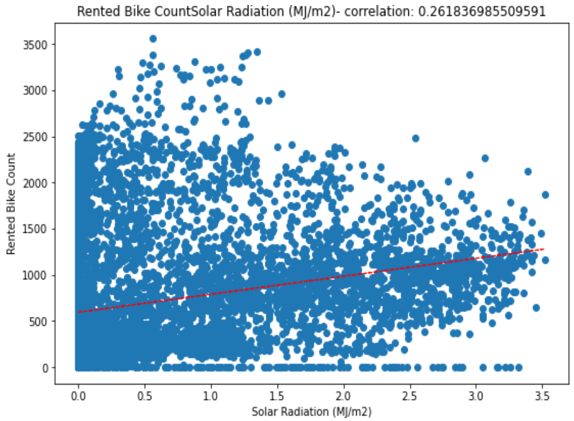
- Visibility (10m)
- Dew point temperature(°C)

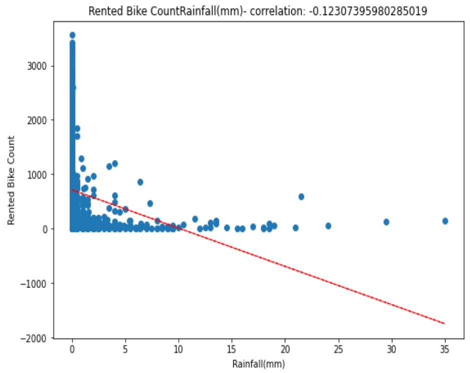
Temperature, Hour and Humidity they are already or almost in Normal Form, So we are not considering them as skewed.

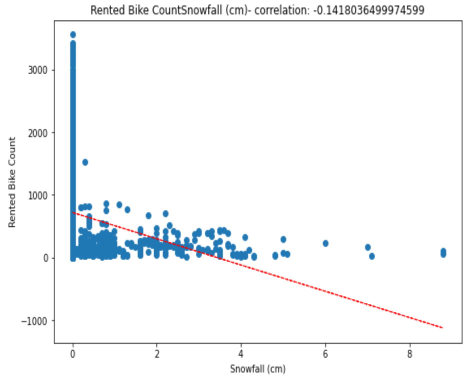
Independent Variables	Skewness
Visibility (10m)	-0.701786
Dew point temperature(°C)	-0.367298
Temperature(°C)	-0.198326
Month	-0.010458
Hour	0.000000
Humidity(%)	0.059579
Wind speed (m/s)	0.890955
Solar Radiation (MJ/m2)	1.504040
Snowfall (cm)	8.440801
Rainfall(mm)	14.533232

Visualizing the relationship b/w dependent & independent variable after transformation :





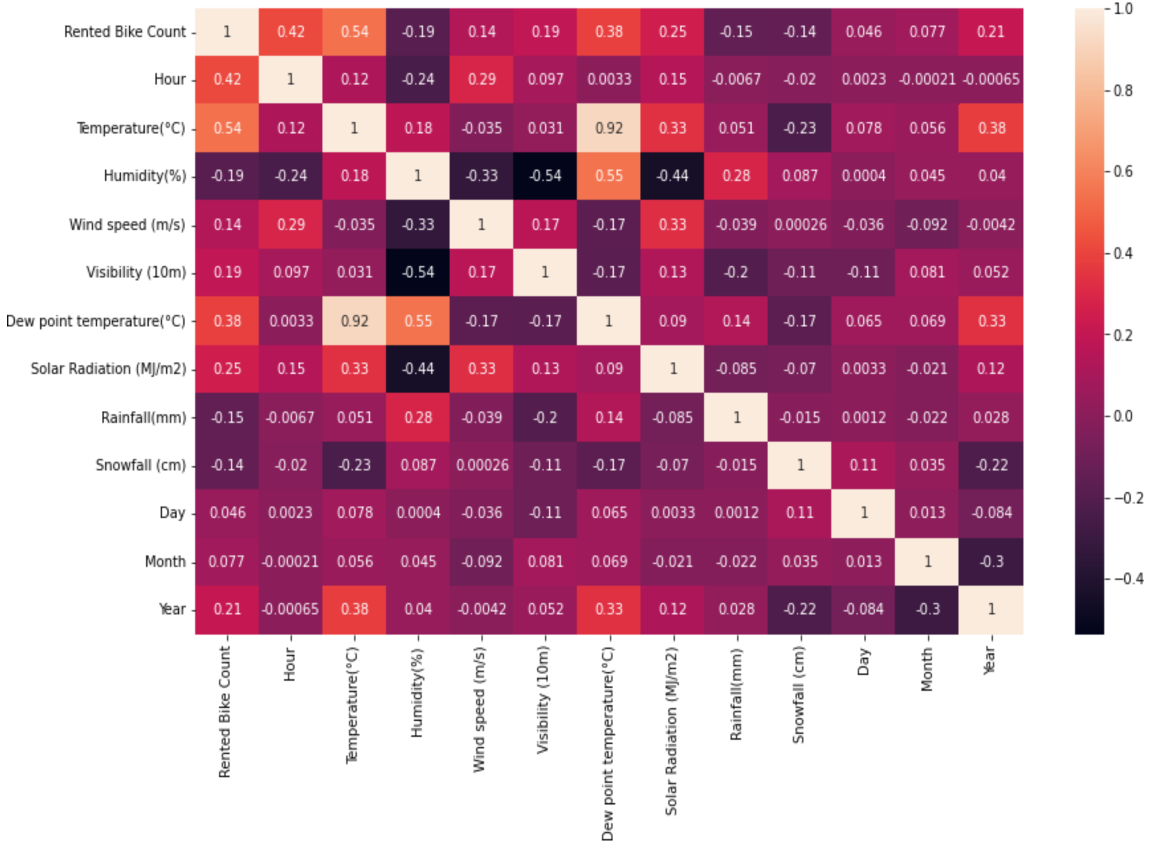




So after visualizing these scatter plots we removed the unwanted or extra data which were making our dataset quite unwell.

So, for windspeed – value higher than 4.5m/s, Solar Radiation(MJ/m2) value higher than 3MJ/m2, Rainfall value higher than 10mm & snowfall value higher than 4cm were not taken.

Multicollinearity :



	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)	Day	Month	Year
Rented Bike Count	1	0.42	0.54	-0.19	0.14	0.19	0.38	0.25	-0.15	-0.14	0.046	0.077	0.21
Hour	0.42	1	0.12	-0.24	0.29	0.097	0.0033	0.15	-0.0067	-0.02	0.0023	-0.00021	-0.00065
Temperature(°C)	0.54	0.12	1	0.18	-0.035	0.031	0.92	0.33	0.051	-0.23	0.078	0.056	0.38
Humidity(%)	-0.19	-0.24	0.18	1	-0.33	-0.54	0.55	-0.44	0.28	0.087	0.0004	0.045	0.04
Wind speed (m/s)	0.14	0.29	-0.035	-0.33	1	0.17	-0.17	0.33	-0.039	0.00026	-0.036	-0.092	-0.0042
Visibility (10m)	0.19	0.097	0.031	-0.54	0.17	1	-0.17	0.13	-0.2	-0.11	-0.11	0.081	0.052
Dew point temperature(°C)	0.38	0.0033	0.92	0.55	-0.17	-0.17	1	0.09	0.14	-0.17	0.065	0.069	0.33
Solar Radiation (MJ/m2)	0.25	0.15	0.33	-0.44	0.33	0.13	0.09	1	-0.085	-0.07	0.0033	-0.021	0.12
Rainfall(mm)	-0.15	-0.0067	0.051	0.28	-0.039	-0.2	0.14	-0.085	1	-0.015	0.0012	-0.022	0.028
Snowfall (cm)	-0.14	-0.02	-0.23	0.087	0.00026	-0.11	-0.17	-0.07	-0.015	1	0.11	0.035	-0.22
Day	0.046	0.0023	0.078	0.0004	-0.036	-0.11	0.065	0.0033	0.0012	0.11	1	0.013	-0.084
Month	0.077	-0.00021	0.056	0.045	-0.092	0.081	0.069	-0.021	-0.022	0.035	0.013	1	-0.3
Year	0.21	-0.00065	0.38	0.04	-0.0042	0.052	0.33	0.12	0.028	-0.22	-0.084	-0.3	1

Here Temperature's and Dew Point Temperature's VIF are highly correlated so we will focus on these two to remove collinearity.
So we will be focusing on Dew Point Temperature.

Label Encoding:

Label Encoding is a popular encoding technique for handling categorical variables. In this technique, each label is assigned a unique integer based on alphabetical ordering.

Here we map the variables like Functioning Day and Holiday in the form of 0 and 1.
also convert the seasons column into dummy variables like Spring, Summer and Winter.

```
#Mapping the Variables
df['Functioning Day']=df['Functioning Day'].map({'Yes':1,'No':0})
df['Holiday']=df['Holiday'].map({'No Holiday':0,'Holiday':1})
```

Feature Engineering:

It is the process of designing artificial features into an algorithm. These artificial features are then used by that algorithm in order to improve its performance, or in other words reap better results.

In Feature Engineering, we apply lambda function to convert respective columns in the form of 0 and 1.
Ex. We convert Visibility column in the form of 1 when it is greater than 2000, also for rainfall if the value is greater than 0.148 then it is converted into 1 otherwise0.
Same procedure follows for snowfall and solar radiation

Feature Selection:

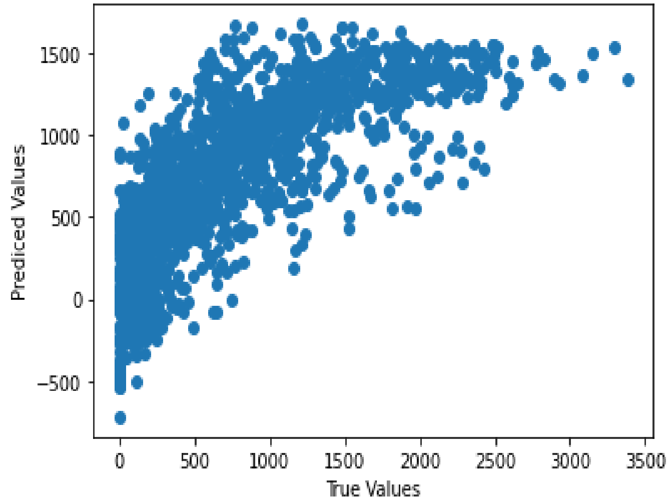
In Feature selection we remove non-informative or redundant predictors from themodel.

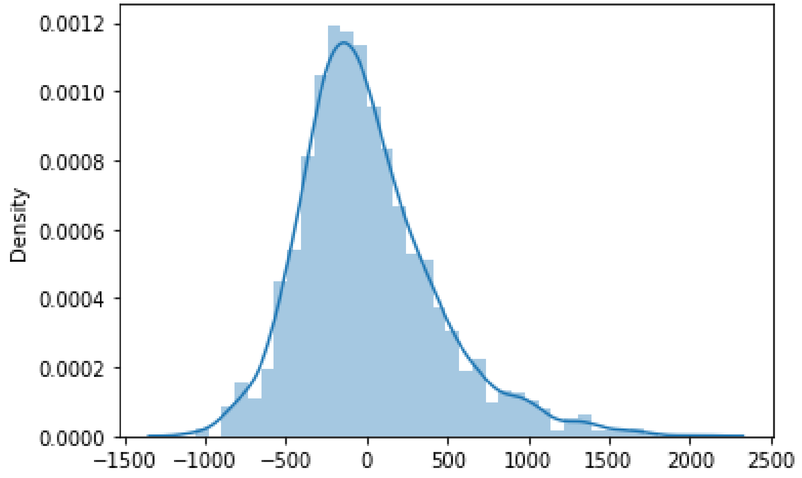
At beginning we have 8760 rows and 14 columns.
After label encoding and feature enginnering we get 8459 rows and 15columns

Algorithms:

1] Linear Regression:

- We have use linear regression model over our data and then we got our prediction model accuracy as in train it was 56.32% and in test it was 57.35%.
- From this we can clearly say that the model is underfitted.
- As we can see from following scatter plot our model is not giving us satisfied result:





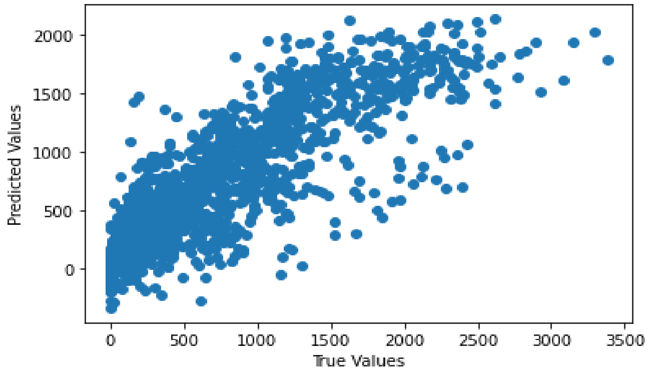
Linear Regression cont.

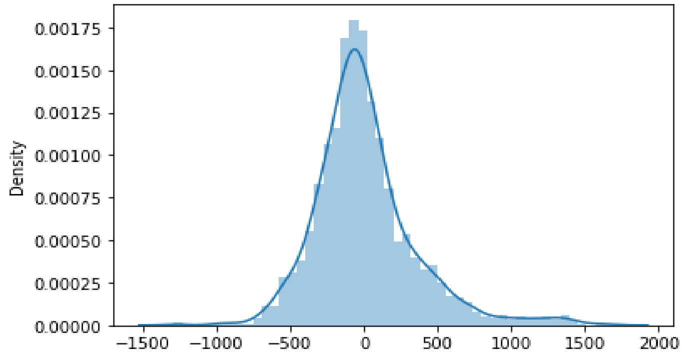
- ❑ According to Linear Regression Assumption.The following cases are not satisfied:

I. There was no Linearity between Predictors vs Target
II. The Most of the features were not normally distributed.
III. There was Some Noisy data in features.
IV. Mean of residuals were not close to zero.

2] **Polynomial Linear Regression:**

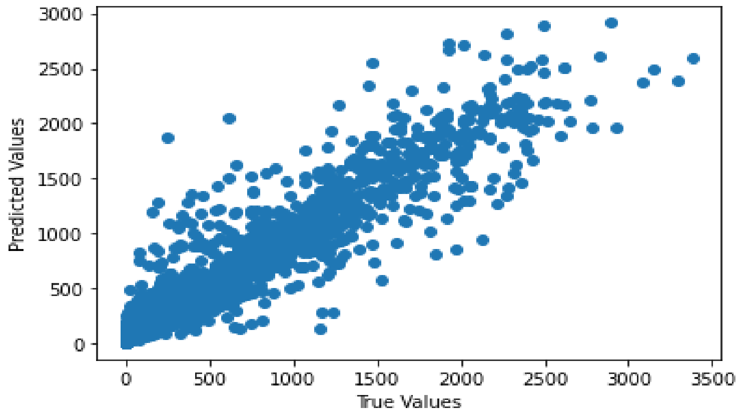
- We received an underfitted model after implementing the linear regression algorithm therefore we're now using the Polynomial Regression algorithm to get an optimal model and satisfied prediction.
- After using the Polynomial Regression algorithm on our data, we were able to achieve better results than the previous algorithm, with 70.89 % on our train data and 71.38 % accuracy on our test data.
- This can be seen in the scatter plot below, our model produces better results as compared to the linear regression algorithm:

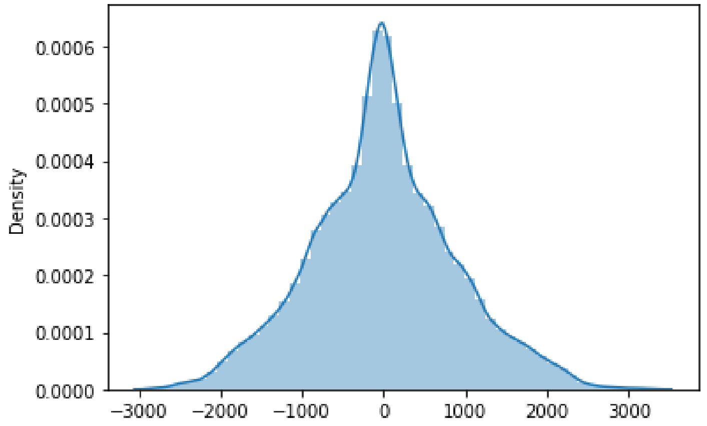




3] Random Forest Regressor:

- We obtained better results after implementing the Polynomial Regression algorithm than we did with previous algorithm, but we still weren't able to get a satisfactory result over our training and testing data, so we used the random forest algorithm to improve our performance.
- So, after using the random forest algorithm, we were able to improve our model prediction accuracy to 98.16 % on training data and 86.08 % on testing data.
- As it can be observed in the scatter plot below, implementing the random forest algorithm, our model delivers better results as compared to linear regression algorithm and Polynomial Regression algorithm:





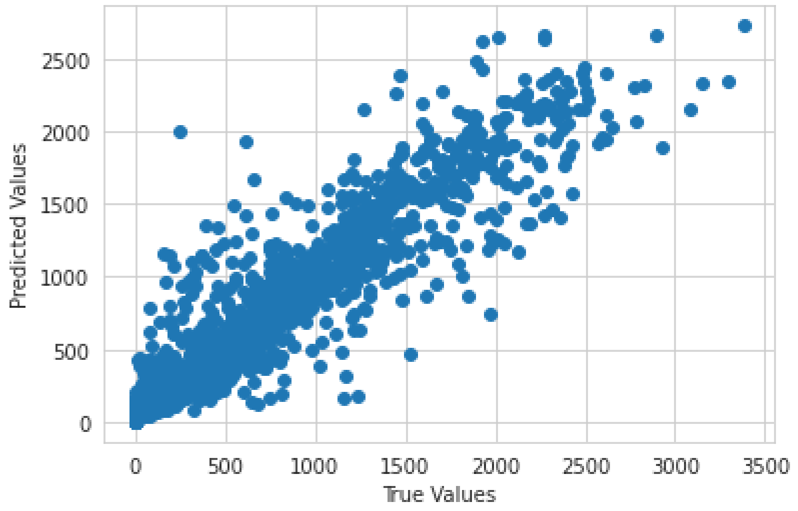
Random Forest Regressor cont.

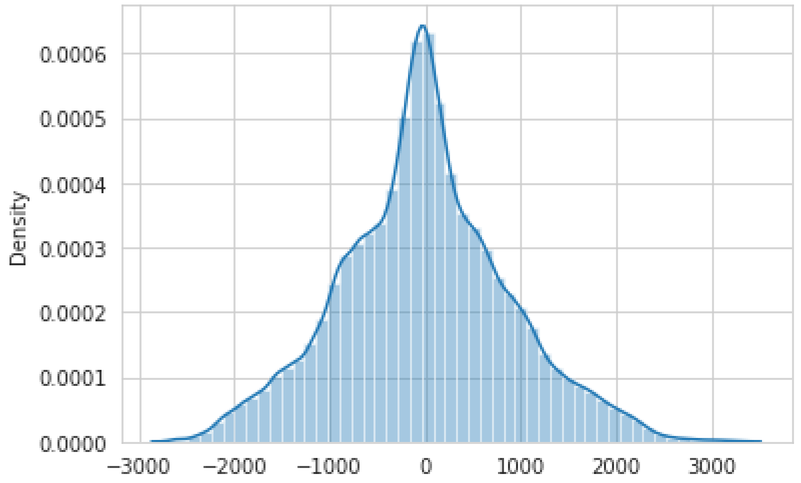
❑ **Hyperparameter tuning using Grid Search CV on Random Forest Regressor:**

- Using random forest, we were able to achieve 98.16 % accuracy on our training data and 86.08 % accuracy on our testing data, resulting in a gap of 12.08 % between our training and testing accuracy.
- As a result of these numbers, we can say that our model is overfitting.
- The algorithm may overfit if we use this model on unknown data to predict the number of rental bikes required at each hour to maintain a stable supply.
- As a result, we must overcome this challenge, and we are doing so by implementing Hyperparameter tuning using Grid Search CV.
- We were able to get 94.95% accuracy on our training data and 86.25% accuracy on our testing data after implementing Hyperparameter tuning using Grid Search CV on Random Forest Regressor.
- By analyzing this outcome, we can conclude that we now have an optimal model with satisfying results.

Random Forest Regressor cont.

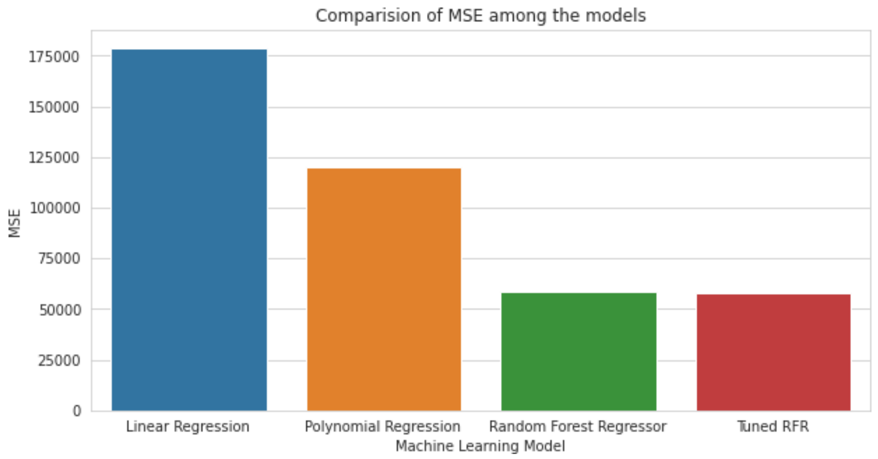
- We can now use this model to predict the number of bikes rented each hour.
- Implementing Hyperparameter tuning on a Random Forest Regressor using Grid Search CV gives satisfactory results, as shown in the scatter plot below:

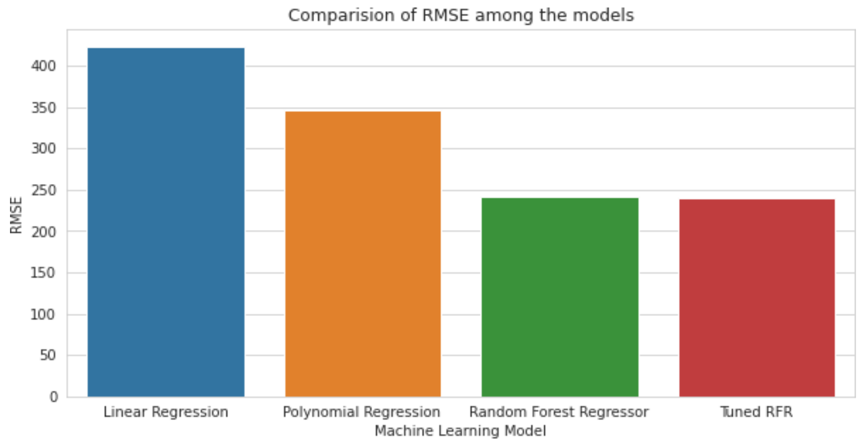




Comparing Evaluation Metrics among all the models being used

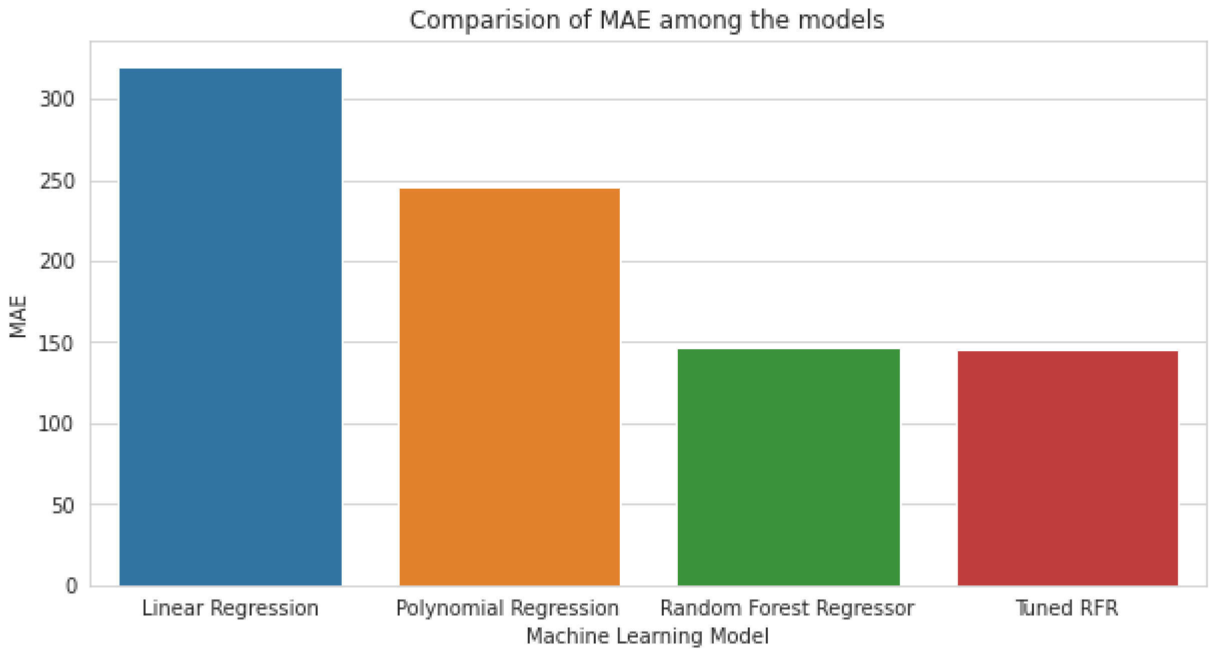
Comparing Mean Squared Error and Root Mean Squared Error among all models being used





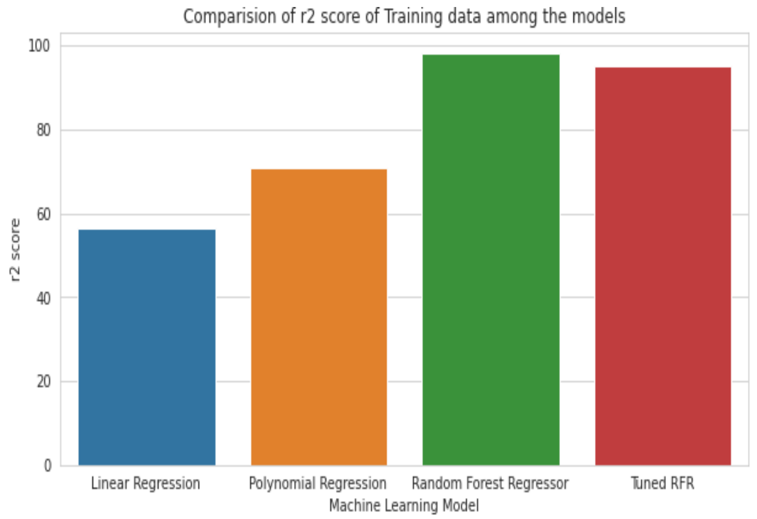
Comparing Evaluation Metrics among all the models being used

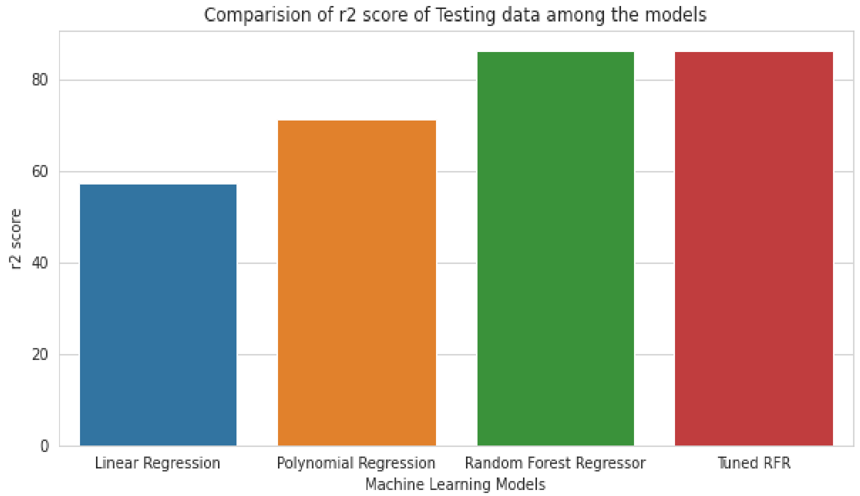
Comparing Mean Absolute Error among all models being used



Comparing Evaluation Metrics among all the models being used

Comparing r2 score on training & testing data among all models being used





Conclusion

- We have used three different models. Linear Regression, Polynomial Regression, Random Forest Regressor, as Linear Regression model is underfit. After then, polynomial regression produces a model that is slightly over fit. As a result, we tested Bagging Models like the Random Forest Regressor, and our r2 score on both the training and testing datasets increased. But RFR was overfitted so we used hyper parameter tuning and we improved the model as a result, Random Forest Regressor has finalized as our final model.

- **Rented Bike Count is very much dependent :-**

1. On what Hour the Bike is rented.
2. At what Temperature the Bike is rented.
3. How much Humidity present in the atmosphere.
4. Is it a functioning day or not.
5. Is it raining outside or not.

- We can say our model is optimized and can be used for predicting bike count on a new data.

Thank you