Capstone Project

Seoul Bike Sharing Demand Prediction

Prajwal D U

Bike Rental Industry

Bike Rental Market

- By Service Type (Pay as you go and Subscription Based),
- By Propulsion (Petrol and Electric),
- By Duration (Short Term and Long Term), and
- By Application (Touring and Commuting)

Introduction

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.

POINTS FOR DISCUSSION

- Data Summary
- Feature Types
- Visualizing Distribution
- Checking outliers
- Manipulating The Data
- Checking linearity in Data
- Correlation matrix
- Model building
    - Linear Regression
    - Decision Tree
    - Random Forest Regressor
    - XGBoost Regressor
    - Gradient Boosting Regressor
- Conclusion

Data Summary

```
] # Check first 5 rows of dataset
data.head()
```

	Date	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)	Seasons	Holiday	Functioning Day
0	2017-01-12	254	0	-5.2	37	2.2	2000	-17.6	0.0	0.0	0.0	Winter	No Holiday	Yes
1	2017-01-12	204	1	-5.5	38	0.8	2000	-17.6	0.0	0.0	0.0	Winter	No Holiday	Yes
2	2017-01-12	173	2	-6.0	39	1.0	2000	-17.7	0.0	0.0	0.0	Winter	No Holiday	Yes
3	2017-01-12	107	3	-6.2	40	0.9	2000	-17.6	0.0	0.0	0.0	Winter	No Holiday	Yes
4	2017-01-12	78	4	-6.0	36	2.3	2000	-18.6	0.0	0.0	0.0	Winter	No Holiday	Yes

- This dataset contains 8760 rows and 14 columns.
- Contains three categorical features ‘Seasons’, ‘Holiday’, and ‘Functioning day’,
- Has one date column(contains date,year,month).
- Remaining 10 columns are numeric types.
- Rented bike count is our target variable.

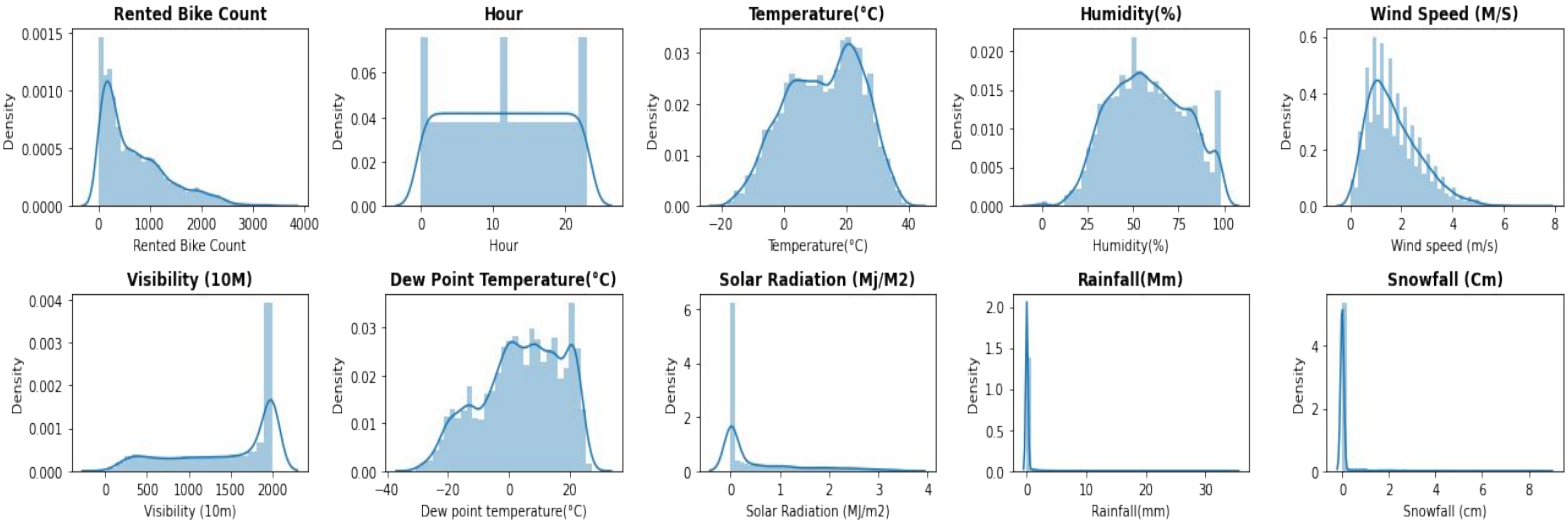
Data Summary

- There is no missing values present.
- There is no duplicate values present.
- There is no null values
- The dataset shows hourly rental data for one year (1 December 2017 to 31
- November 2018) (365 days).

Feature Summary

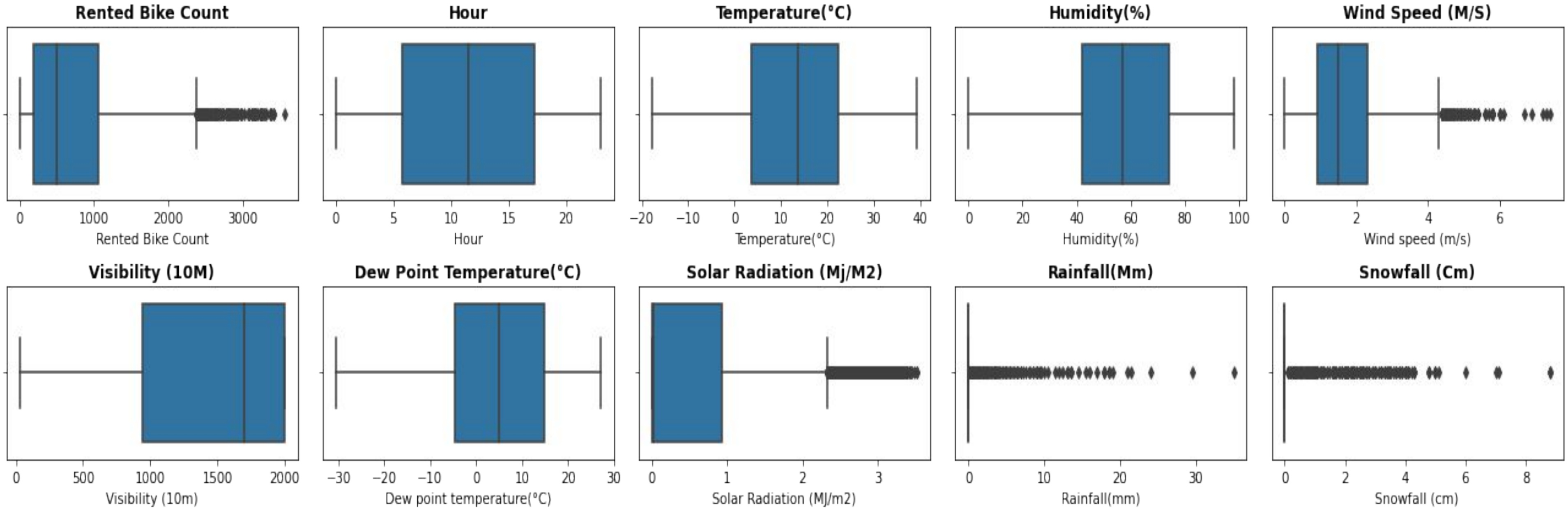
- Date : year-month-day
- Rented Bike count : Count of bikes rented at each hour
- Hour : Hour of the day
- Temperature : Temperature in Celsius
- Humidity : %
- Windspeed : m/s
- Visibility : 10m
- Dew point temperature : Celsius
- Solar radiation : MJ/m2
- Rainfall : mm
- Snowfall : cm
- Seasons :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

Visualizing Distribution



As we see distribution plot all features are not distributed linearly

Checking outliers



- As wee seen above plot wind speed, solar radiation, rainfall and snowfall contains are contains outliers
- But know we don’t treat it as outliers because seasons are different in different countries, so think that as anomalies(rare event).
- We treated outliers in the target variables by capping with IQR method

Manipulating The Dataset

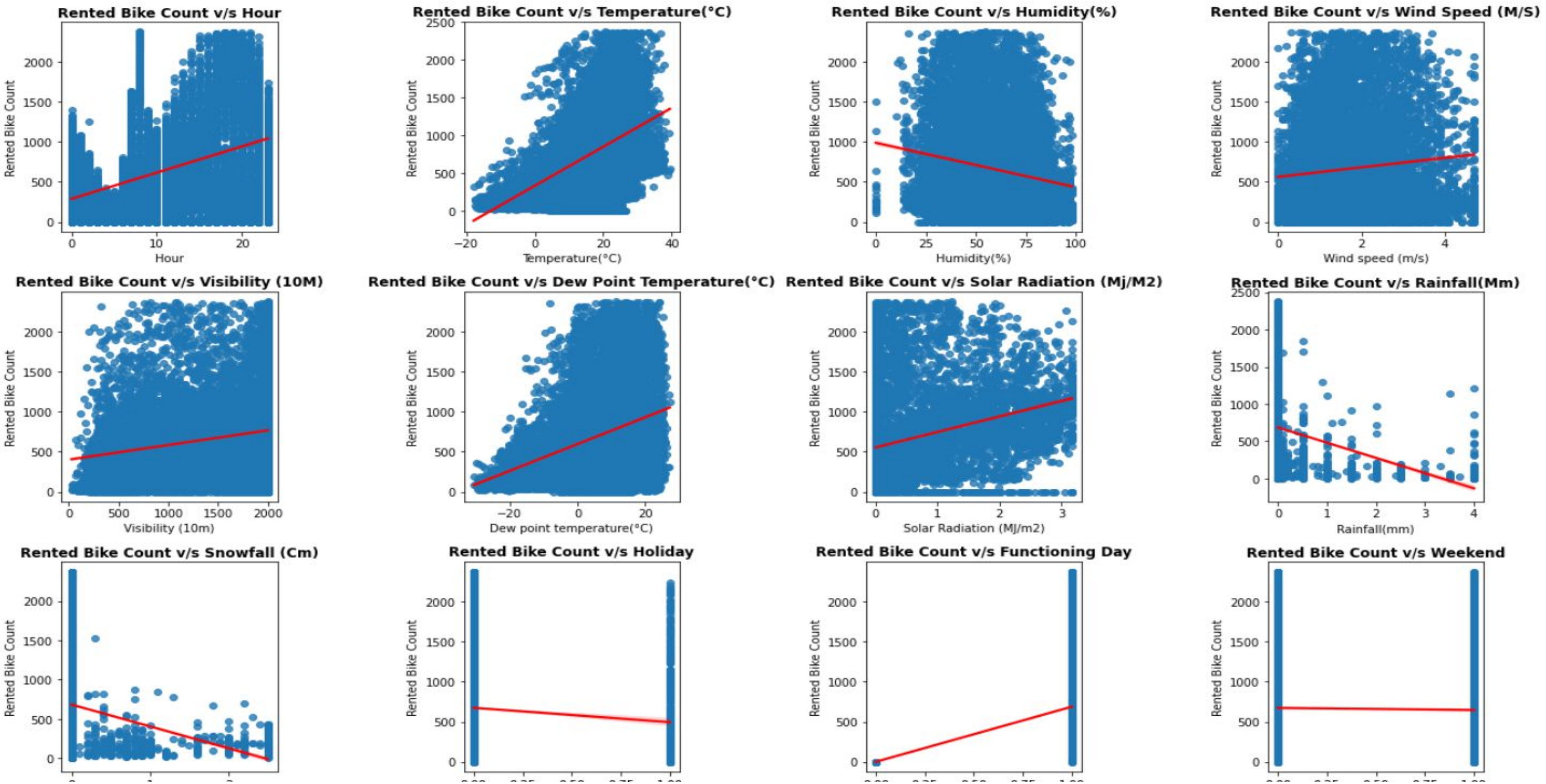
Added new features using date column

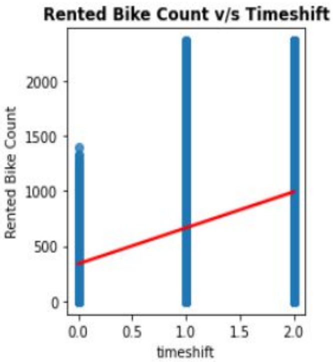
- Created dummy features using season columns **Spring**, **summer**, **autumn**, **winter** with one hot encoding
- **Weekend** that shows whether it is weekend are not(saturday and sunday are 1 else 0).
- **Functioning day** giving operating day are 1 else 0.
- **timeshift** based on time interval, here we divided into 3 values day(1),night(0) and evening(2).
- And after taking all data we drop **date** column.

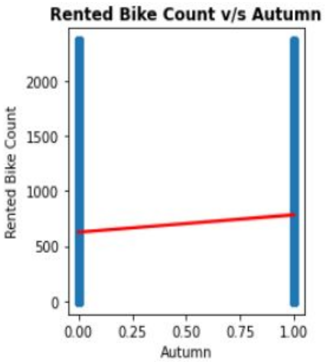
```python
for col in categorical_features:
  print(data[col].value_counts(), '\n')
```

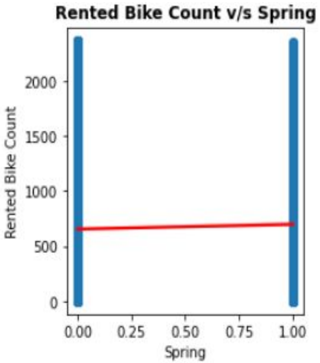
```
Spring     2208
Summer     2208
Autumn     2184
Winter     2160
Name: Seasons, dtype: int64

No Holiday    8328
Holiday        432
Name: Holiday, dtype: int64

Yes    8465
No      295
Name: Functioning Day, dtype: int64

day        3650
night      2555
evening    2555
Name: timeshift, dtype: int64
```

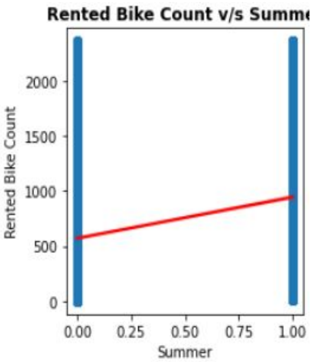
Checking linearity in Data

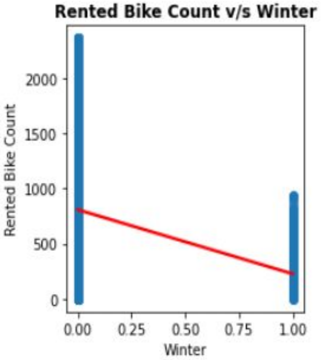






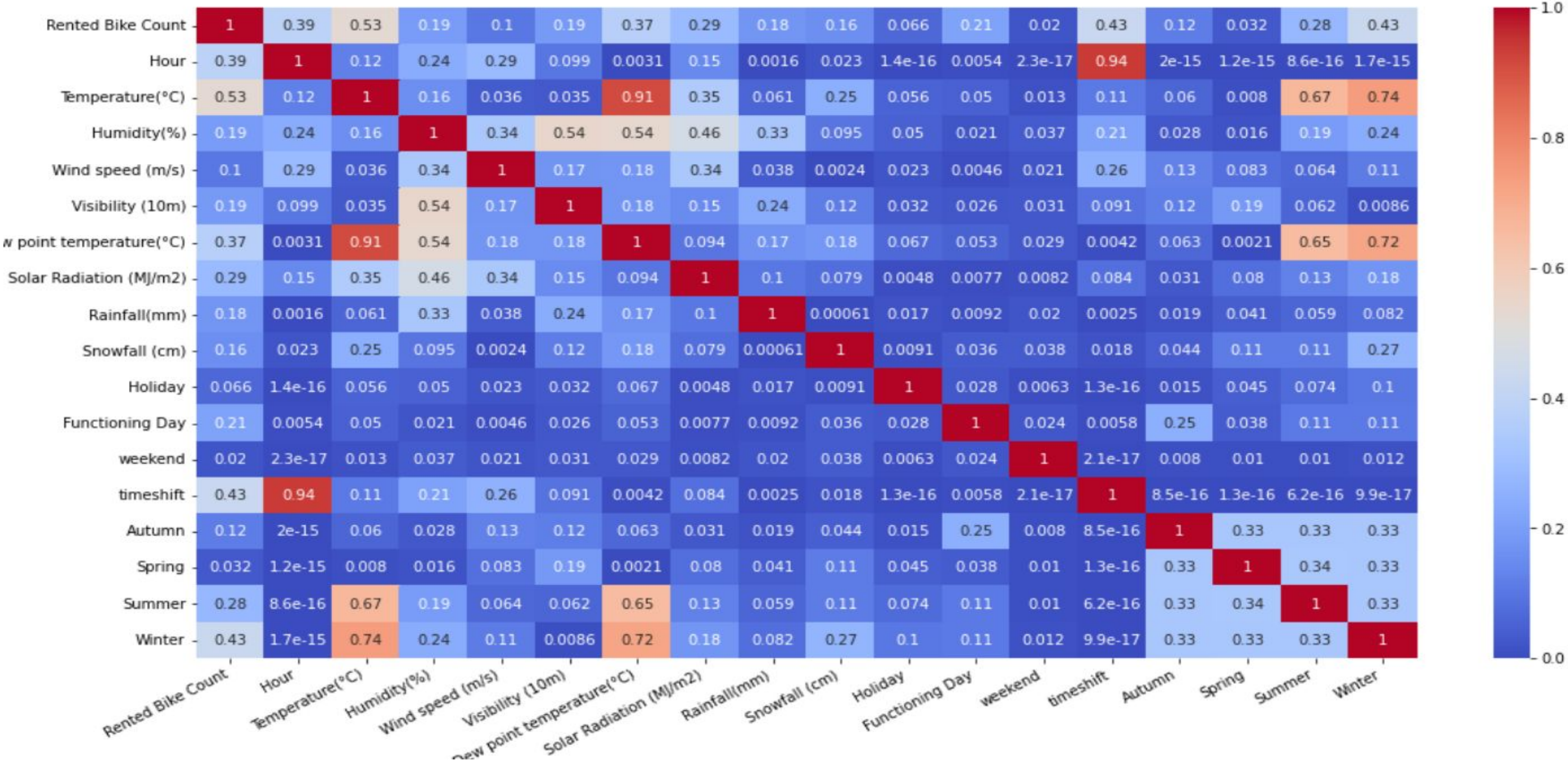






- From this visualization we see Hour, Temperature, Solar Radiation, Dew point Temperature are positively correlated with Rented Bike Count.
- Humidity, Rainfall, Snowfall, and Winter are negatively correlated with Rental Bike Count.
- Some features are also showing close to zero correlation with the target variable as the regression line is not inclined.

Correlation matrix



Handling Multicollinearity

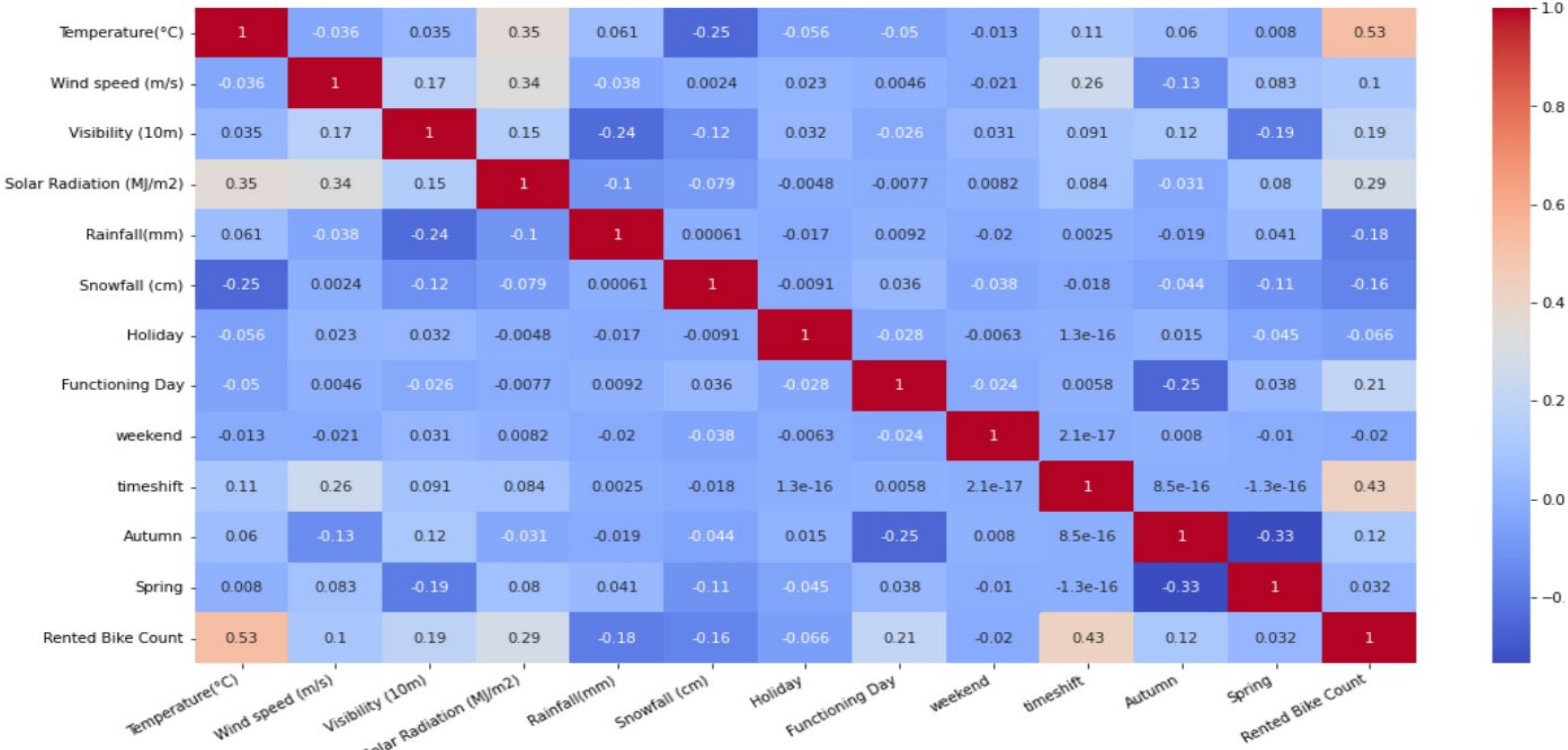
	variables	VIF
0	Dew point temperature(°C)	119.298136
1	Summer	116.141121
2	Spring	112.673201
3	Autumn	110.725563
4	Winter	107.844468
5	Temperature(°C)	90.833188
6	Humidity(%)	21.238433
7	Hour	8.781649
8	timeshift	8.555039
9	Solar Radiation (MJ/m2)	2.078721
10	Visibility (10m)	1.691780
11	Wind speed (m/s)	1.313277
12	Rainfall(mm)	1.179250
13	Snowfall (cm)	1.147787
14	Functioning Day	1.081776
15	Holiday	1.023520
16	weekend	1.007038

- Multicollinearity allow us to look at correlations(that is how one variable changes with respect to other).
- **Correlation** tells us how strongly, pairs of variables are related to one another.
- Dew point temperature and summer are highly correlated followed by Hour and timeshift.
- We can see some highly correlated, let's treat them by excluding them from dataset and checking variance inflation factors.
- VIF determines the strength of the correlation between independent variables, It is predicted by taking a variable and regressing it against every other variable, VIF score of an independent variable represents how well the variable is explained by other independent variables.

	variables	VIF
0	Functioning Day	8.973136
1	Visibility (10m)	6.903425
2	Wind speed (m/s)	4.784533
3	timeshift	2.956516
4	Temperature(°C)	2.685255
5	Solar Radiation (MJ/m2)	1.944365
6	Spring	1.528702
7	Autumn	1.468795
8	weekend	1.396051
9	Snowfall (cm)	1.131983
10	Rainfall(mm)	1.110783
11	Holiday	1.056152

- Since Summer and Winter can also be classified on the basis of temperature and we already have that feature present. Even if we drop these features the useful information will not be lost. So we dropped them.
- We continued to exclude the features with VIF > 10 and finally we obtained the following results.

Updated Heatmap



Model Building Prerequisites

Before fitting the model we has to normalize our dataset.

Feature scaling or standardization : It is a step of data preprocessing which is applied to independent variables, It basically helps to normalize the data within a particular range and sometimes also helps in speeding up the calculations in an algorithm.
MinMaxScaler : It's range between (0 to 1), **Normalization** scales our features to predefined range, independently of the statistical distribution they follow, it does this using the minimum and maximum values of each features in dataset.

$$Xnormalised = \frac{X - Xmin}{Xmax - Xmin}$$

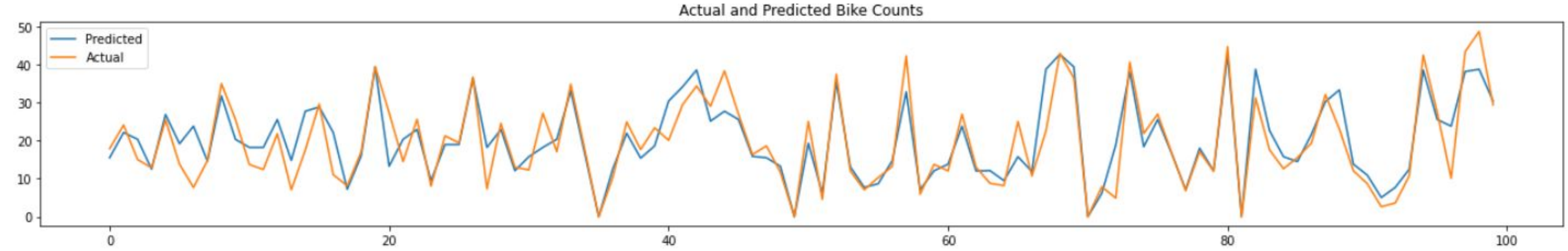
Model Building Prerequisites

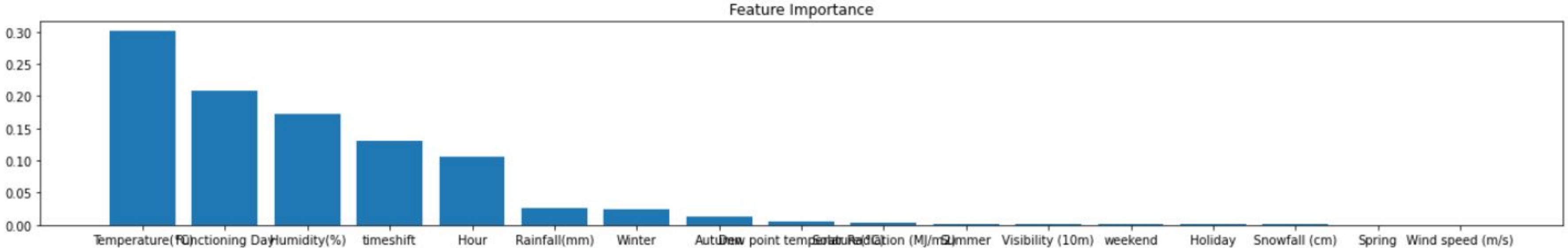
For fitting a model we defining a function called **analyse_model** to save time in model building, in **analyse_model** we wrote **model, X_train, X_test, y_train, y_test,** and evaluation matrix like **MSE, RMSE, MAE, TRAIN R2, TEST R2, ADJUSTED R2** and also plotted the feature importance based on the algorithm used.

We also defined some range of values for Hyperparameter :

- Number of trees: n_estimators =[50,100,150]
- Maximum depth of trees: [6,8,10]
- Minimum number of samples required to split a node: [50,100,150]
- Minimum number of samples required at each leaf node: [40,50]
- learning rate : Eta=[0.05, 0.08, 0.1]

Linear Regression

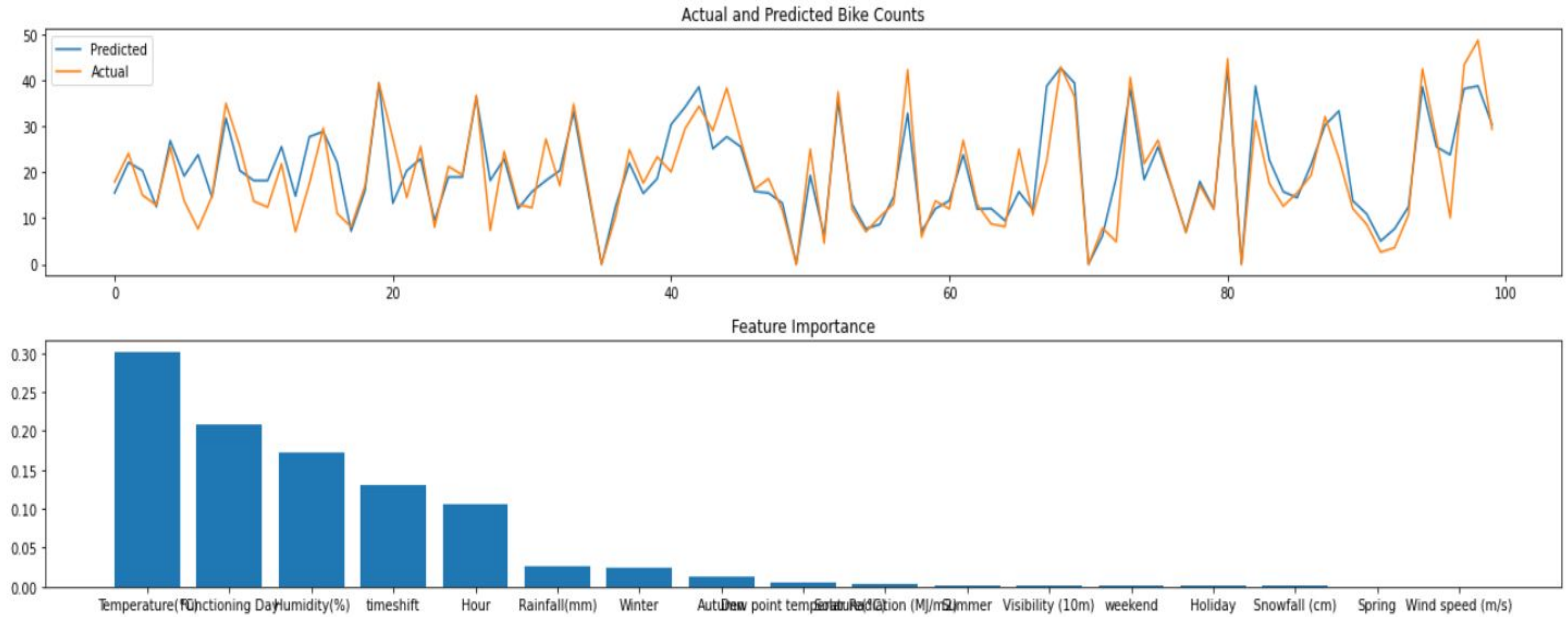





```
MSE : 137241.3084686744
RMSE : 370.46094054390454
MAE : 254.74045552944642
Train R2 : 0.5837621350247335
Test R2 : 0.5924062591863408
Adjusted R2 :  0.5895936514291448
```

- We plotted the actual and predicted using line chart and also plotted feature importance using bar chart.
- Since the performance of simple linear model is not so good, so we experiment with some other complex model.

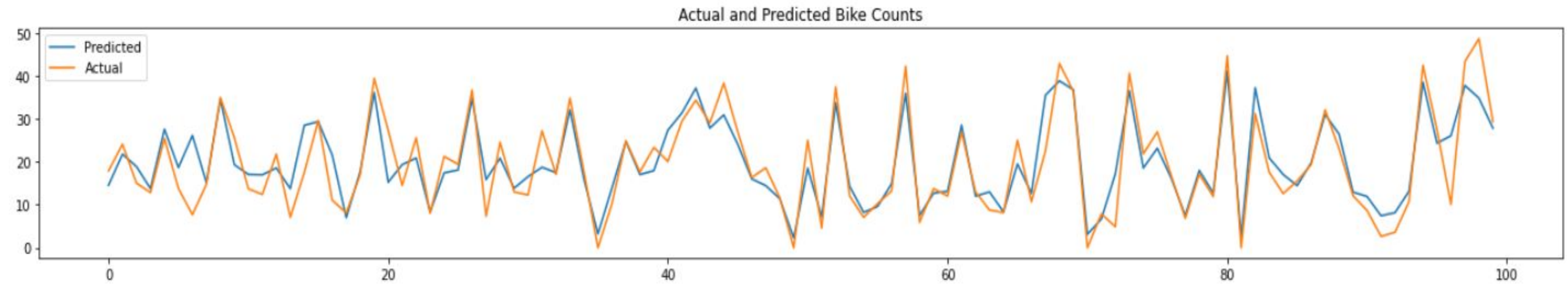
Decision Tree

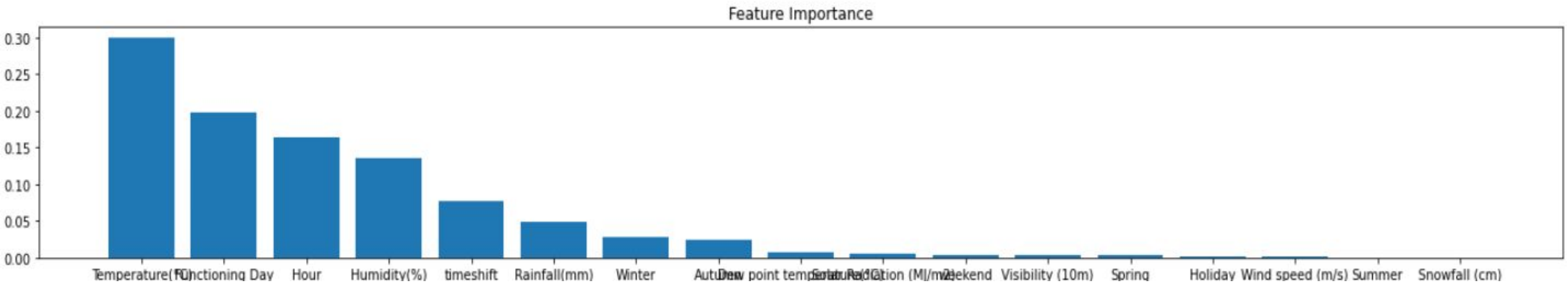



```
MSE : 91524.53332018365
RMSE : 302.53021885455286
MAE : 188.5071046099557
Train R2 : 0.7598960015979025
Test R2 : 0.7281807691252598
Adjusted R2 :  0.7255158747049192
```

- DecisionTreeRegressor(max_depth=10,min_samples_leaf=40,min_samples_split=50,random_state=1)
- Decision tree performs well better than the linear reg with a test r2 score more than 70%.

Random Forest Regressor

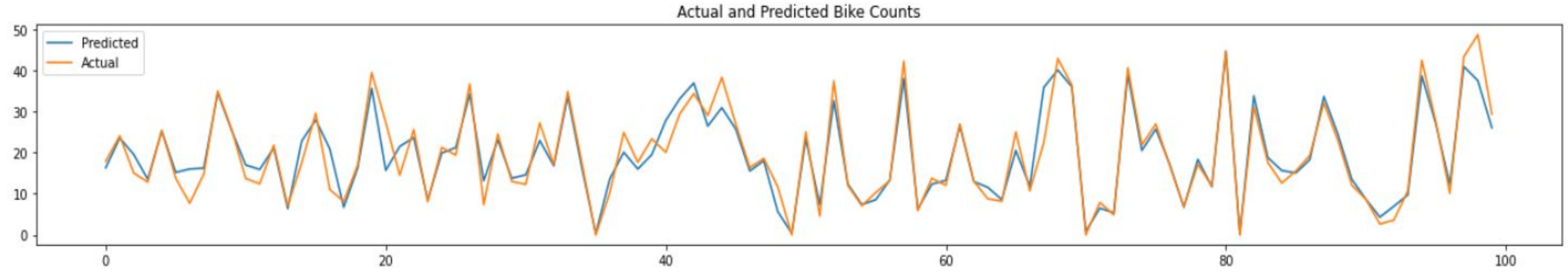


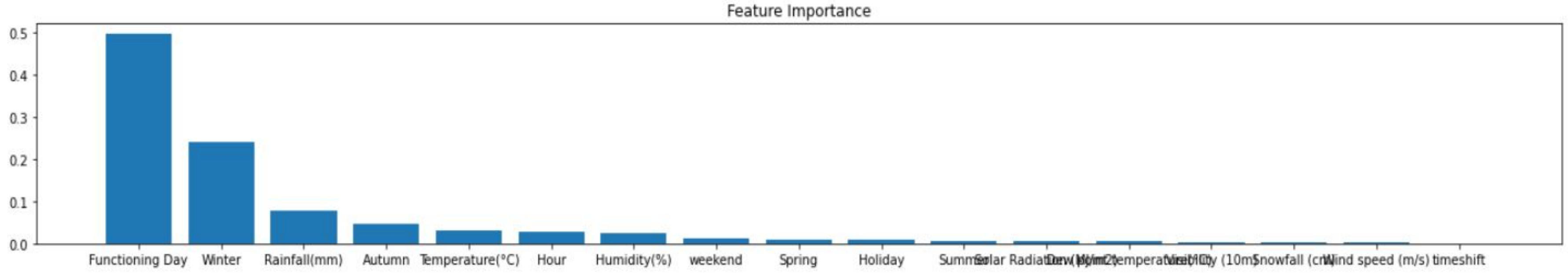



```
MSE : 84724.33236299588
RMSE : 291.0744447095895
MAE : 179.09620419309925
Train R2 : 0.773412260770052
Test R2 : 0.7483767245365814
Adjusted R2 :  0.7459098296790969
```

- RandomForestRegressor(max_depth =10, min_samples_leaf =40, min_samples_split =50, random_state =2).
- Random forest also performs well in both test and train data with a r2 score 77% on train data and around 75% on the test data.

XGBoost Regressor

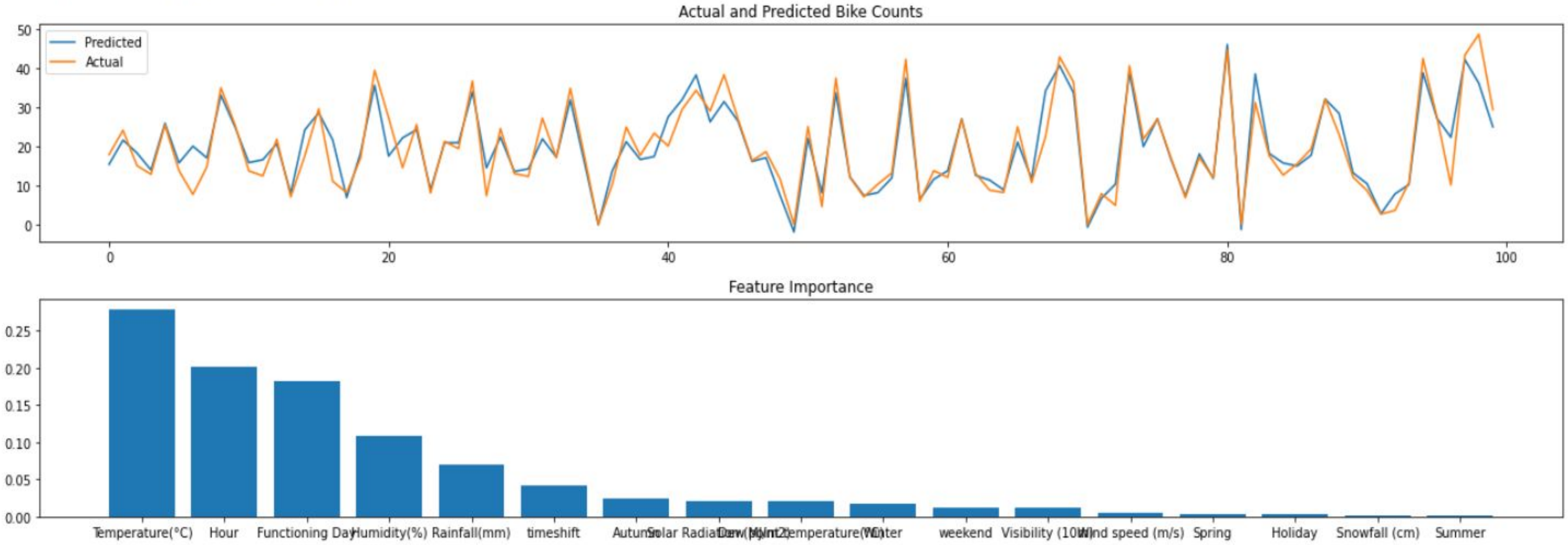





```
MSE : 58913.99867290589
RMSE : 242.72206054025227
MAE : 134.1361227702089
Train R2 : 0.961794302333021
Test R2 : 0.825030981026666
Adjusted R2 :  0.8233155984877119
```

- XGBRegressor (eta=0.05, max_depth=8, min_samples_leaf =40, min_samples_split =50, n_estimators =150, random_state =3, silent=True)
- XGBoost regressor emerges as the best model according to the evaluation matrix score both in the train and test.

Gradient Boosting Regressor




```
MSE : 61590.84383506893
RMSE : 248.17502661442174
MAE : 141.1977249022155
Train R2 : 0.9078337007467008
Test R2 : 0.8170810033894738
Adjusted R2 :  0.8152876798932921
```

GradientBoostingRegressor(max_depth =10, min_samples_leaf =50, min_samples_split =50, n_estimators =150, random_state =4)

We experimented this boosting algorithm in order to enhance the performance but we found out that its performance is closely equal to the XGBoost model.

Conclusion

- The independent variables in data given does not have a good linear relation with the target variable so the simple linear model was not performing good on this data. Tree based Algorithms seem to perform well in this case.
- Functioning day is the most influencing feature and temperature is at the second place for Linear regression.
- Temperature is the most important feature for Decision Tree , Random Forest and Gradient Boosting Regressor.
- Functioning day is the most important feature and Winter is the second most for XGBoost Regressor.
- The feature temperature is on the top list for all the regressors except XGBoost.
- XGBoost is acting different from all the regressors as it is considering whether it is winter or not. And is it a working day or not. Though winter is also a function of temperature only but it seems this trick of XGBoost is giving better results.
- XGBoost Regressor has the Least Root Mean Squared Error (242.72). So It can be considered as the best model for given problem.

Thank You

PRAJWAL D U