

Capstone Project - 2

Bike Sharing Demand Prediction

Introduction



Bike sharing system is a shared transport service in which bicycles are made available for shared use to individuals on a short-term basis for a price.

This system is designed to resolve issues of traffic congestion, noise, and air pollution.

I have used Seoul's bike sharing system dataset which contains weather data and the number of bikes rented per hour.

Problem Statement



It is important to make the rental bike available and accessible to the public at the right time as it lessens the waiting time.

This project tackles the problem of predicting the number of bikes required at each hour for the stable supply of rental bikes.

The main objective is to build various regression models and analyze their performance with respect to each other so as to get the best performing model.

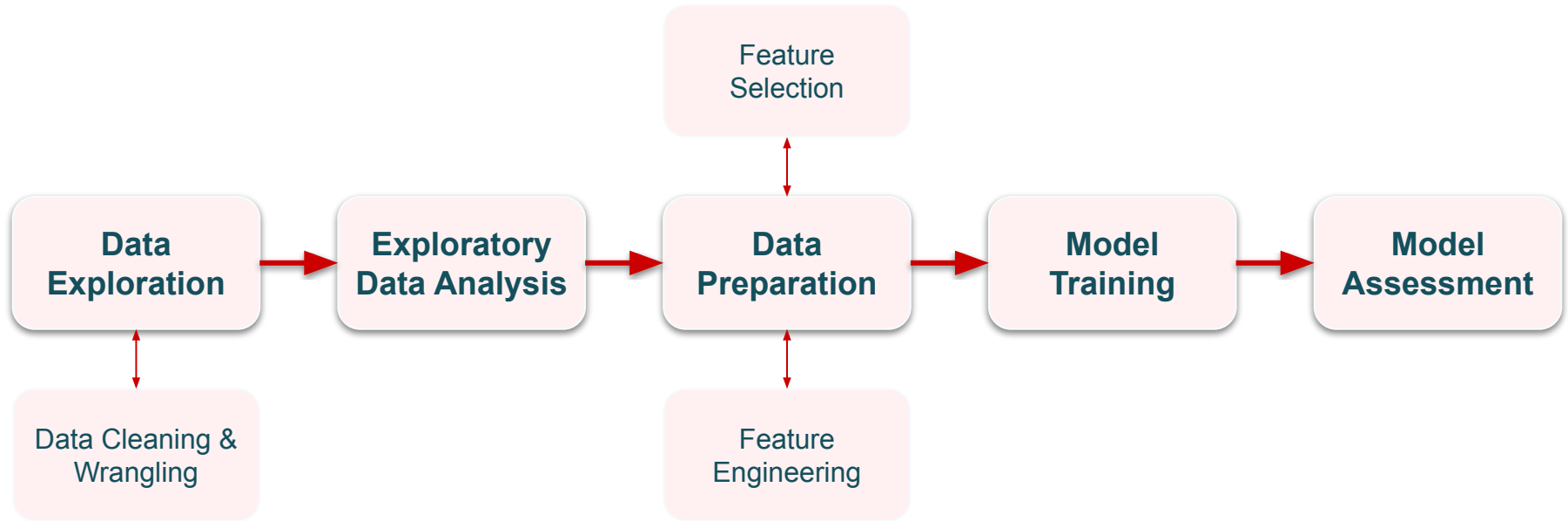
Data Overview

- We are provided with the Seoul Bike sharing demand dataset.
- The dataset contains weather information, the number of bikes rented per hour and date information.
- The time span of the dataset is 365 days from December 2017 to November 2018.

Attributes Information

- 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 - No/Yes

Steps Involved



Data Exploration

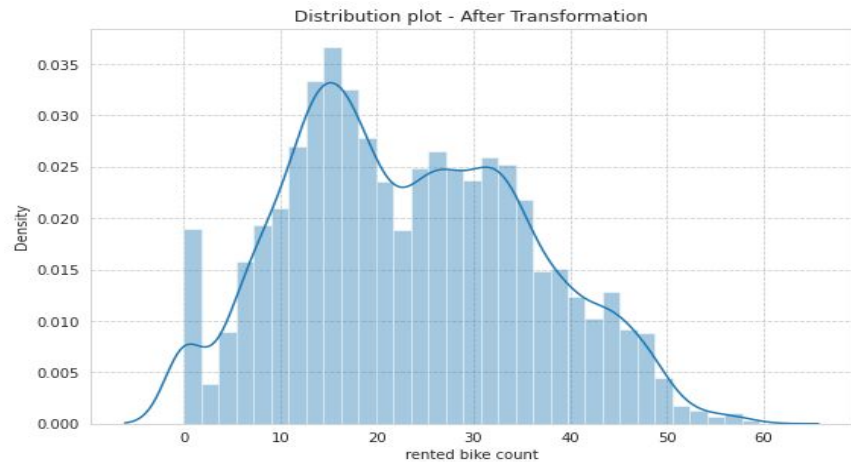
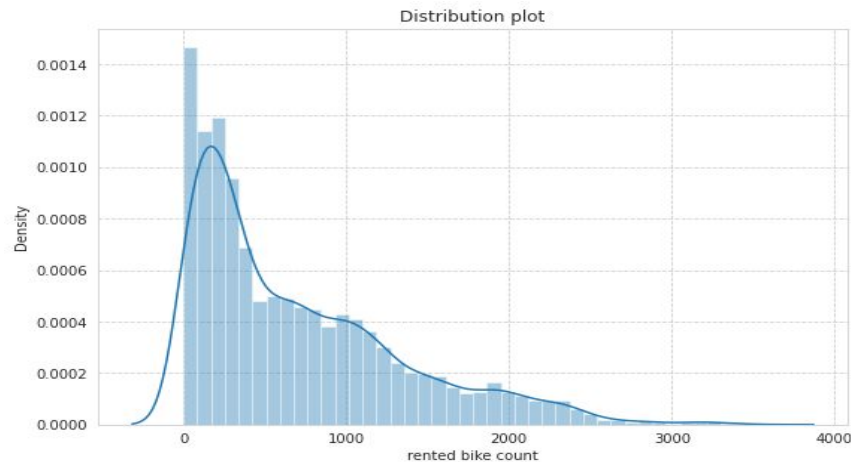
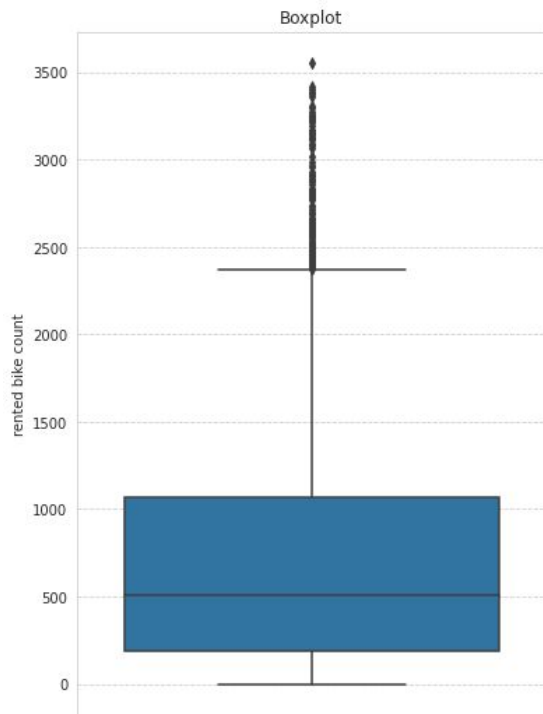
- The dataset consists of 8760 rows and 14 features(columns).
- Dependent Variable- Rented Bike Count
- Categorical variables - Seasons, Holiday and Functioning day
- Date variable was stored as string so converted it to Datetime format

Data Cleaning & Wrangling

- Created some additional features from date.
- No duplicate values
- No missing values

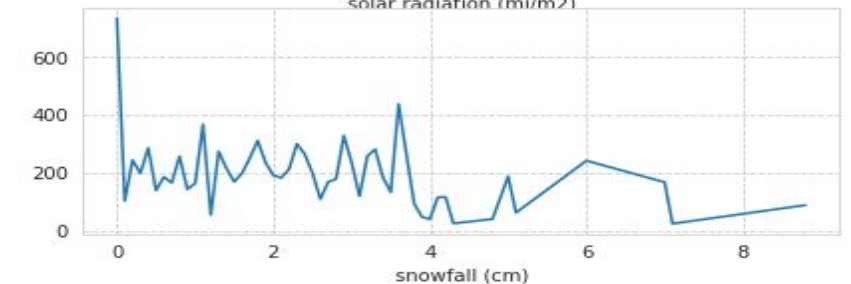
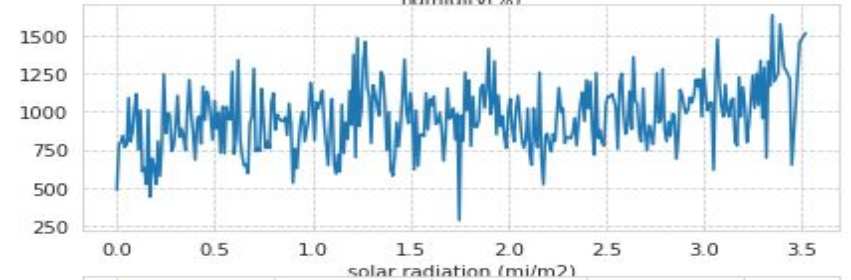
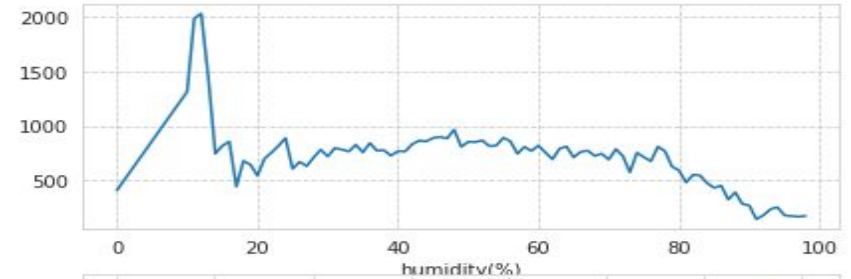
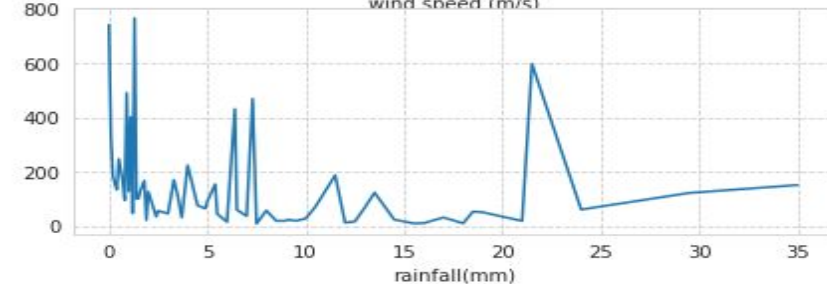
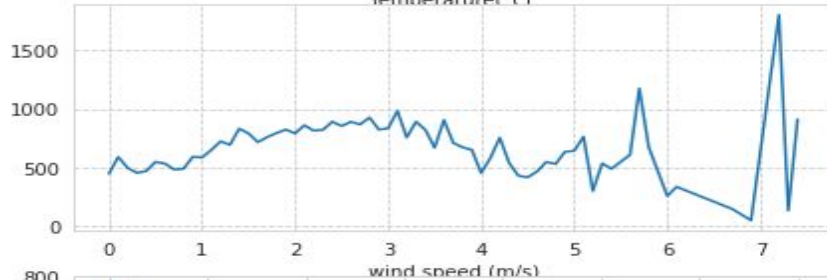
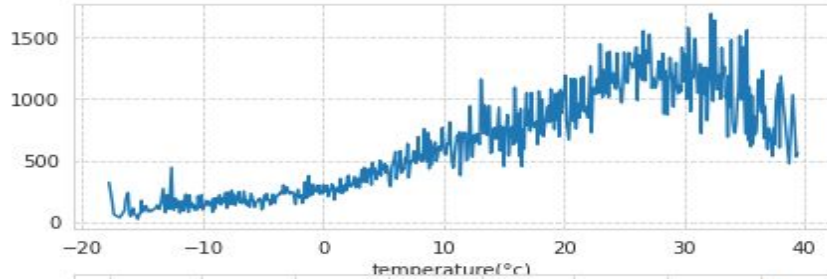
Exploratory Data Analysis

Dependent Variable - Rented Bike Count

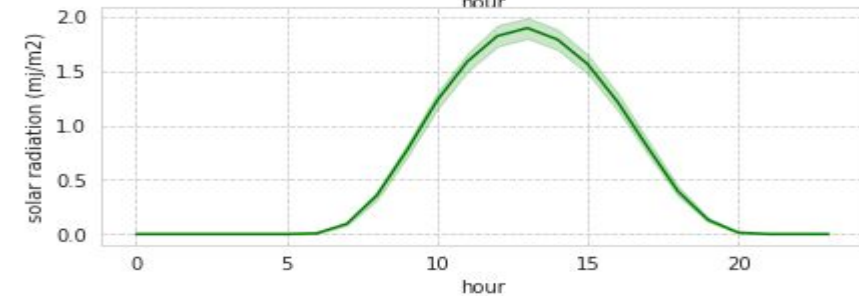
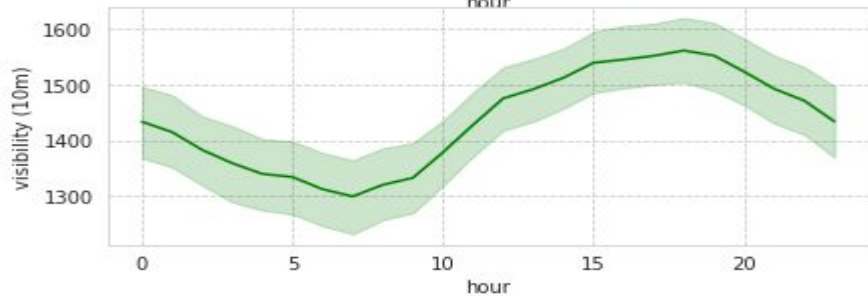
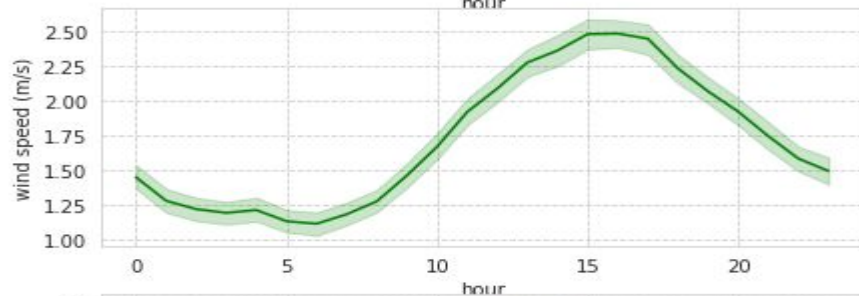
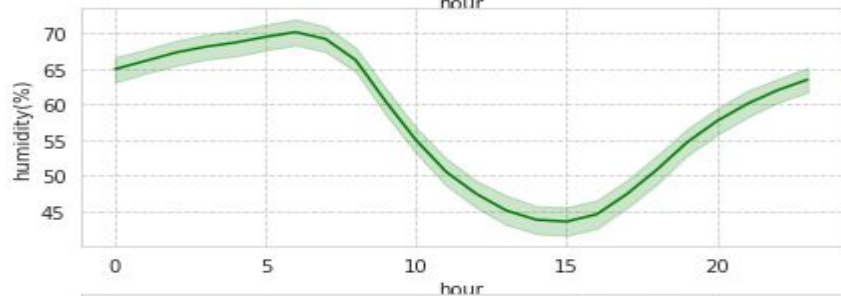
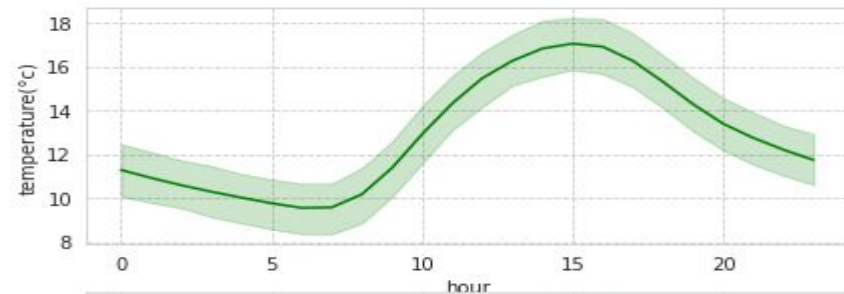
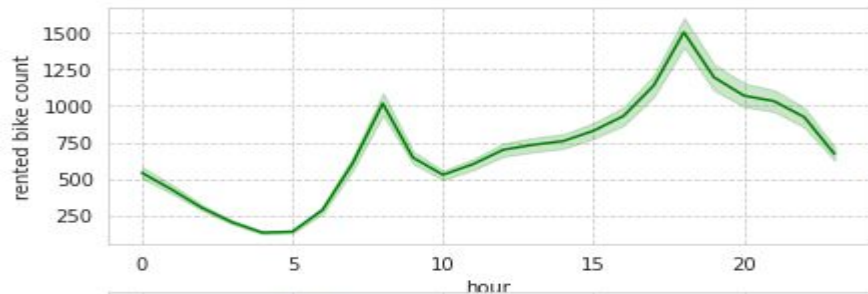


Bivariate Analysis - Numerical Variables

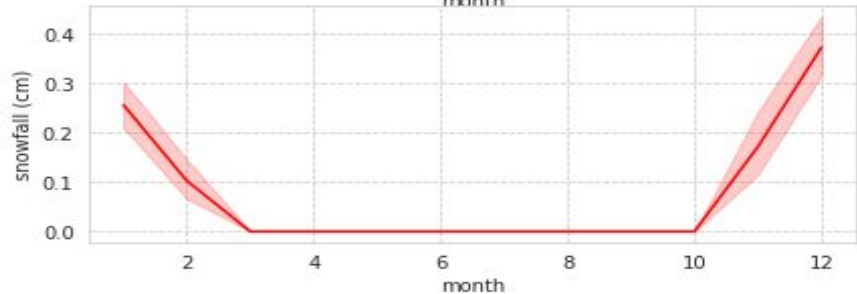
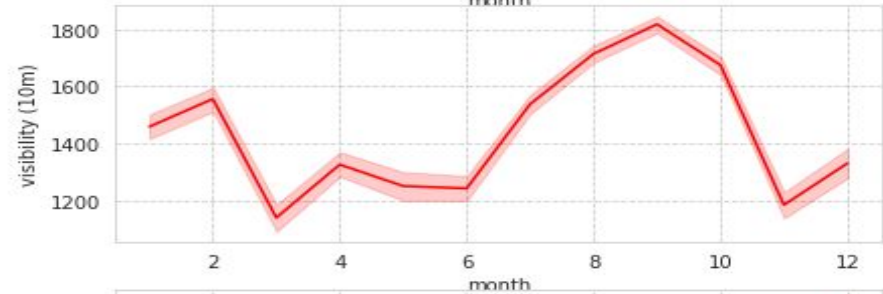
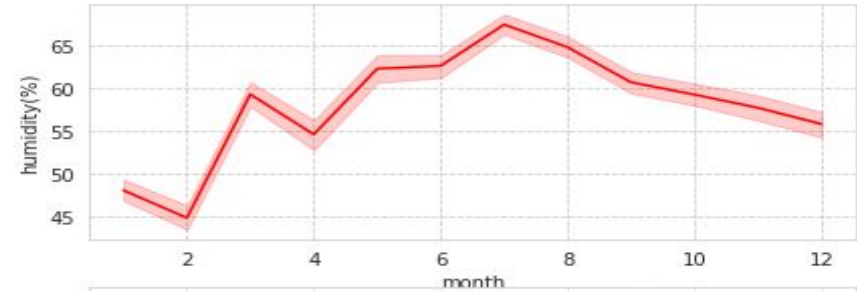
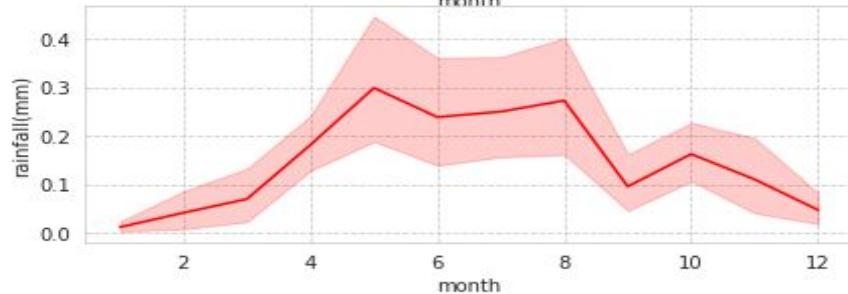
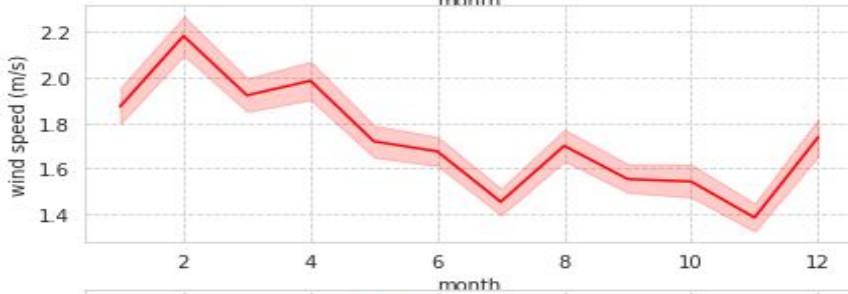
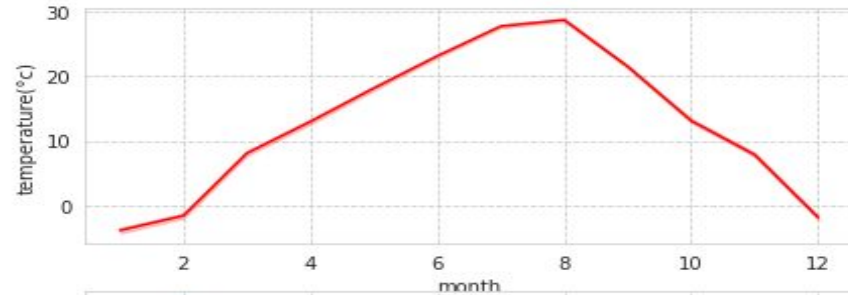
Line Plots - Numerical variables v/s Rented Bike Count



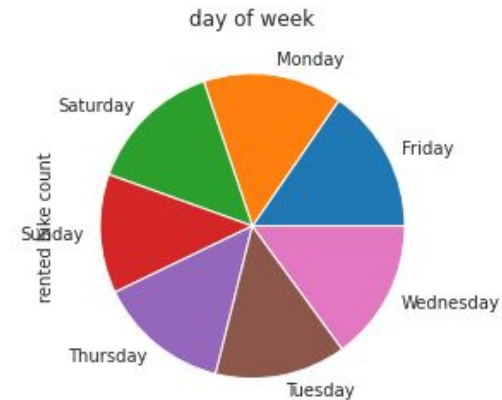
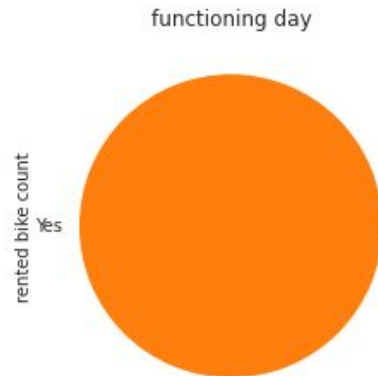
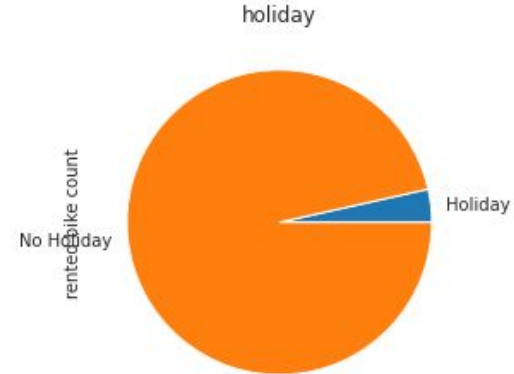
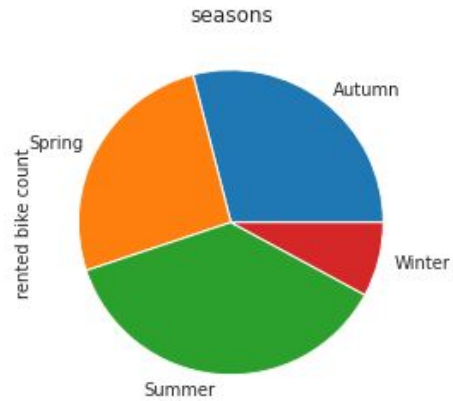
Spread of Numerical variables across hours



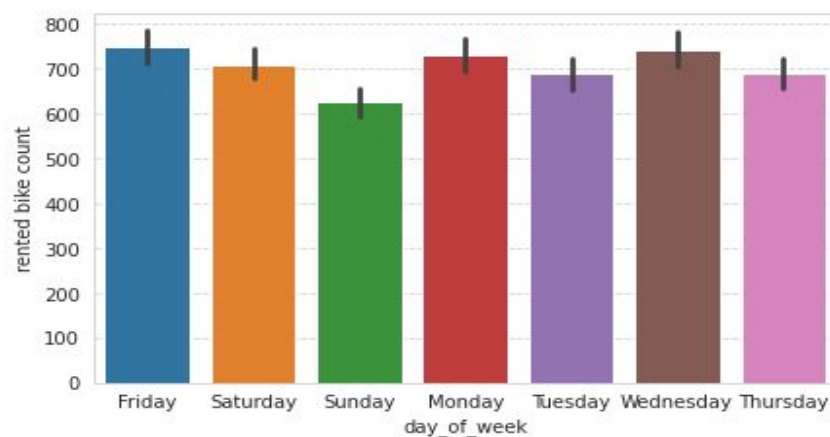
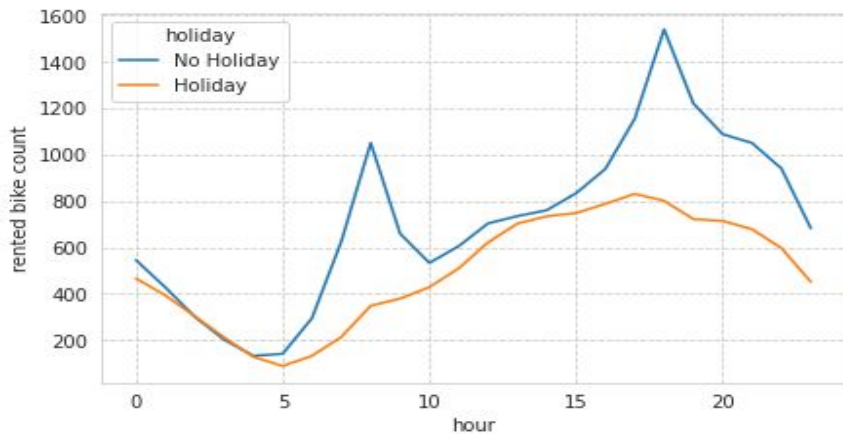
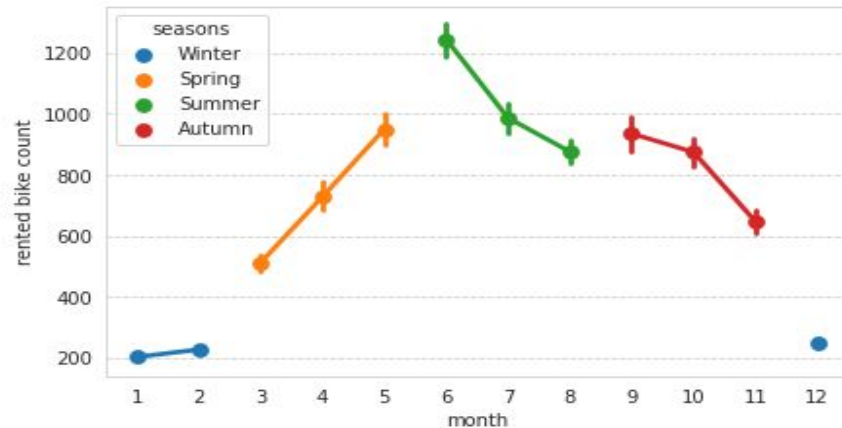
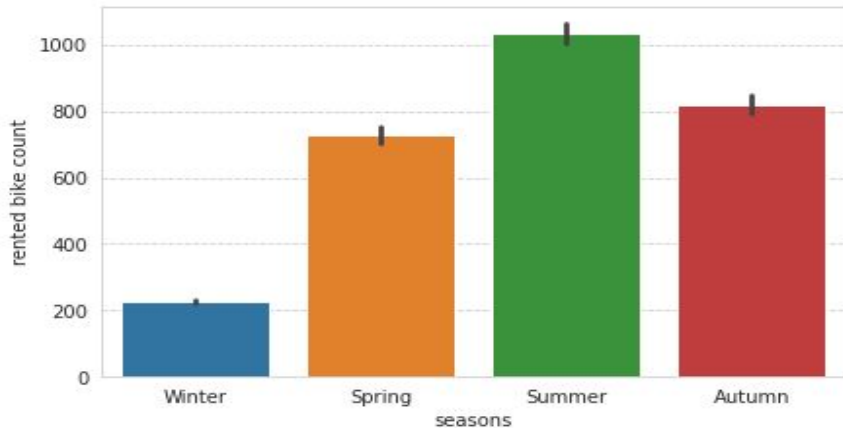
Spread of Numerical variables across months



Bivariate Analysis - Categorical Variables



Spread of Rented Bike Count across categorical variables



Data Preparation



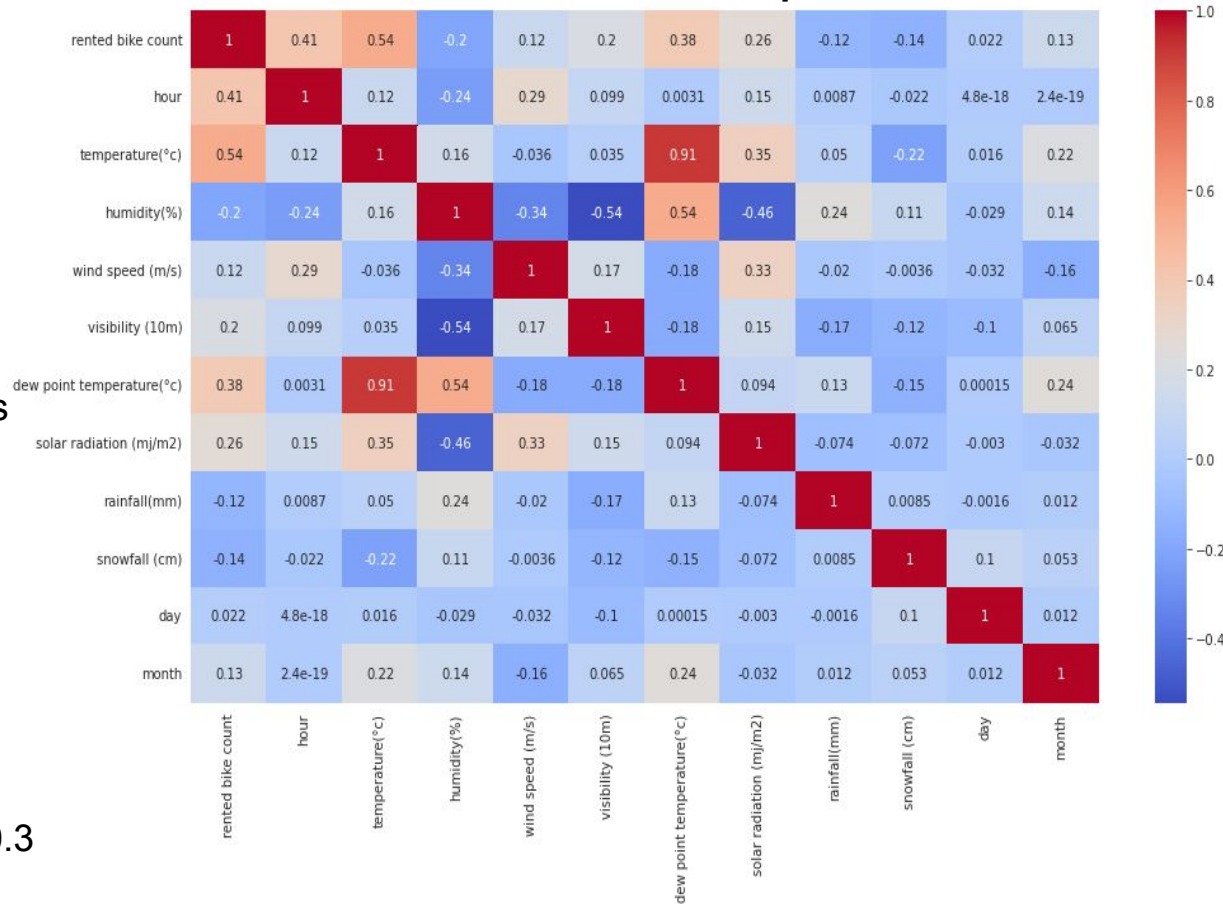
Feature Selection

- Feature Selection with Pearson Correlation Heatmap
- Detecting multicollinearity using Variance Inflation Factor(VIF)
- Dropping highly correlated features

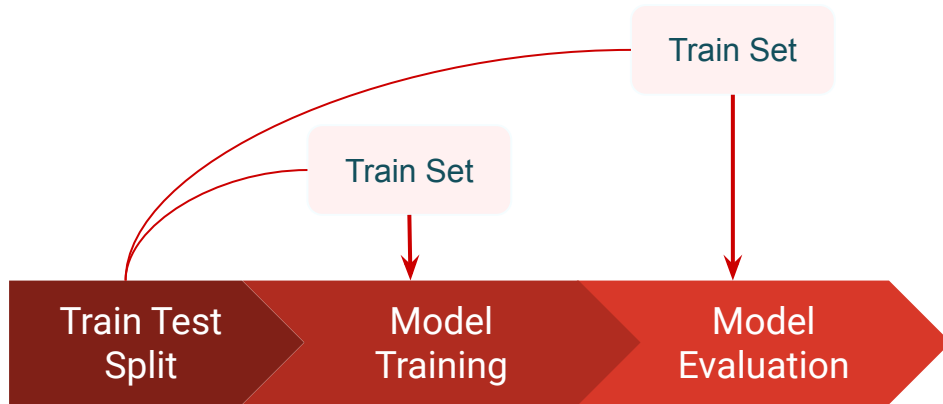
Feature Engineering

- Created additional column of weekend
- Label Encoding of categorical variables
- Train Test Split having test size = 0.3

Correlation Heatmap



Predictive Modeling



Regression Models used -

- | | |
|----------------------|-----------------------|
| 1) Linear Regression | 2) Lasso Regression |
| 3) Ridge Regression | 4) Elastic Net |
| 5) Decision Tree | 6) Random Forest |
| 7) Gradient Boosting | 8) XGBoost Regression |

Predictive modeling includes -

- Building and training the models
- Tuning the hyperparameters to get better performance
- Model Evaluation and Selection

Linear Regression

Train set Metrics

MSE : 53.80611459663623

RMSE : 7.335265134719823

MAE : 5.63805172995284

R2 : 0.6540967727241054

Adjusted R2 : 0.65250945389913

Test set Metrics

MSE : 53.740491144516426

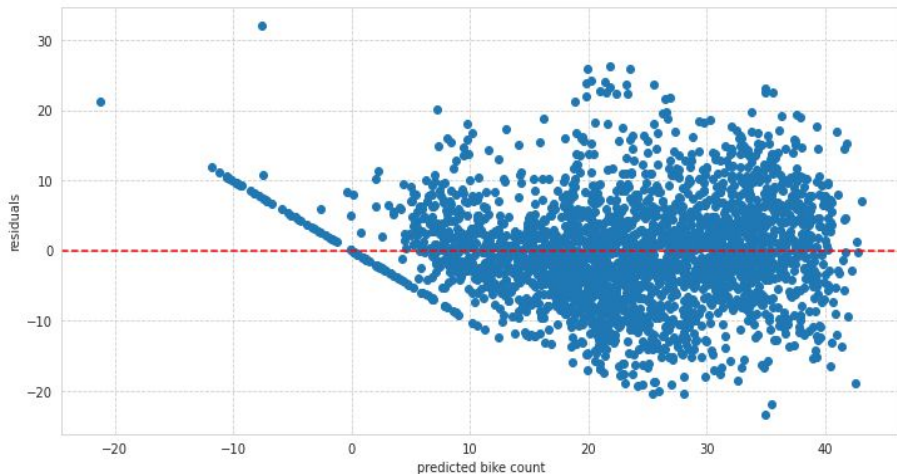
RMSE : 7.330790622062291

MAE : 5.661287586240897

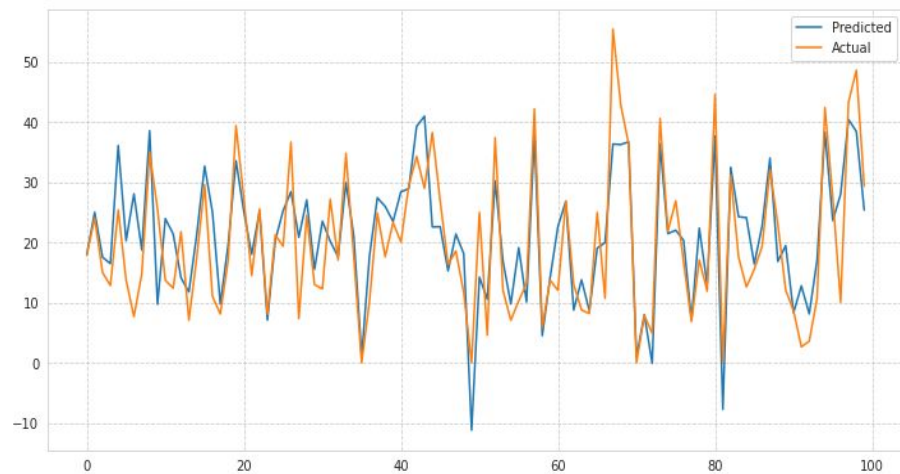
R2 : 0.6501226505752834

Adjusted R2 : 0.6485170948609061

Heteroskedasticity



Graph of actual and predicted values



Lasso Regression

Parameters = {'alpha': [1e-15, 1e-10, 1e-8, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1, 5, 10, 20, 30, 40, 45, 50, 55, 60, 100]}

The best fit alpha value is found out to be : {'alpha': 0.0001}

Train set Metrics

MSE : 53.80611534623119

RMSE : 7.3352651858151106

MAE : 5.63804777968967

R2 : 0.6540967679051862

Adjusted R2 : 0.6525094490580972

Test set Metrics

MSE : 53.74016699398597

RMSE : 7.330768513190549

MAE : 5.661284867852185

R2 : 0.6501247609565992

Adjusted R2 : 0.6485192149265722

Graph of actual and predicted values



Ridge Regression

Parameters = {'alpha': [1e-10, 1e-5, 1e-4, 1e-3, 1e-2, 0.5, 1, 1.5, 5, 10, 20, 30, 35, 40, 45, 50, 55, 60, 100]}

The best fit alpha value is found out to be : {'alpha': 1e-10}

Train set Metrics

MSE : 53.80611459663623

RMSE : 7.335265134719823

MAE : 5.638051729952847

R2 : 0.6540967727241054

Adjusted R2 : 0.65250945389913

Test set Metrics

MSE : 53.74049114451563

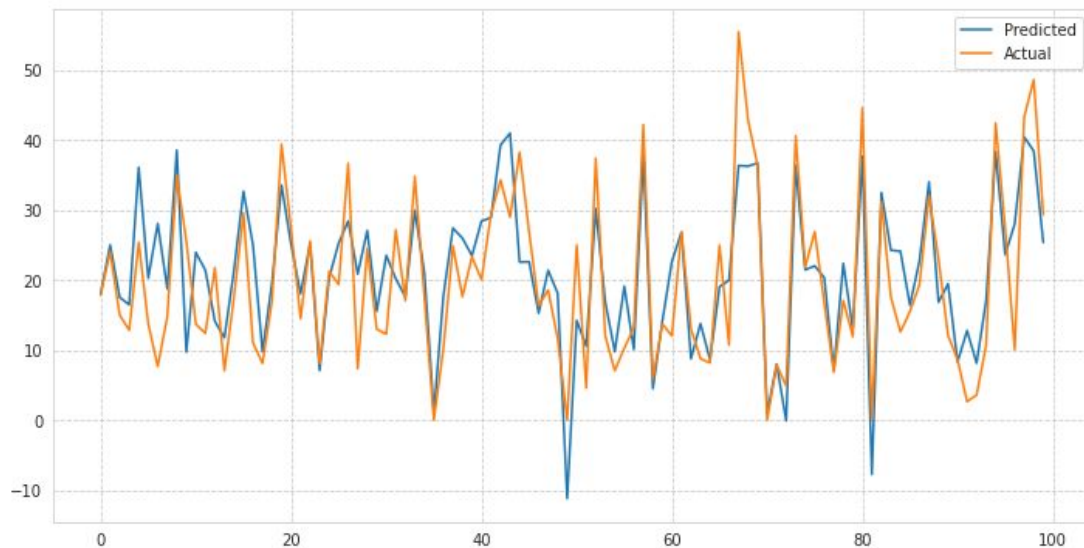
RMSE : 7.330790622062237

MAE : 5.661287586240932

R2 : 0.6501226505752886

Adjusted R2 : 0.6485170948609114

Graph of actual and predicted values



Elastic Net Regression

```
Parameters = {'alpha': [1e-15, 1e-10, 1e-5, 1e-3, 1e-2, 1e-1, 1, 5, 10, 20, 30, 40, 50, 100],  
             'l1_ratio': [0.1, 0.2, 0.3, 0.4, 0.5]}
```

The best parameters found out : {'alpha': 1e-15, 'l1_ratio': 0.2}

Train set Metrics

MSE : 53.80611459663623

RMSE : 7.335265134719823

MAE : 5.638051729952843

R2 : 0.6540967727241054

Adjusted R2 : 0.65250945389913

Test set Metrics

MSE : 53.740491144516405

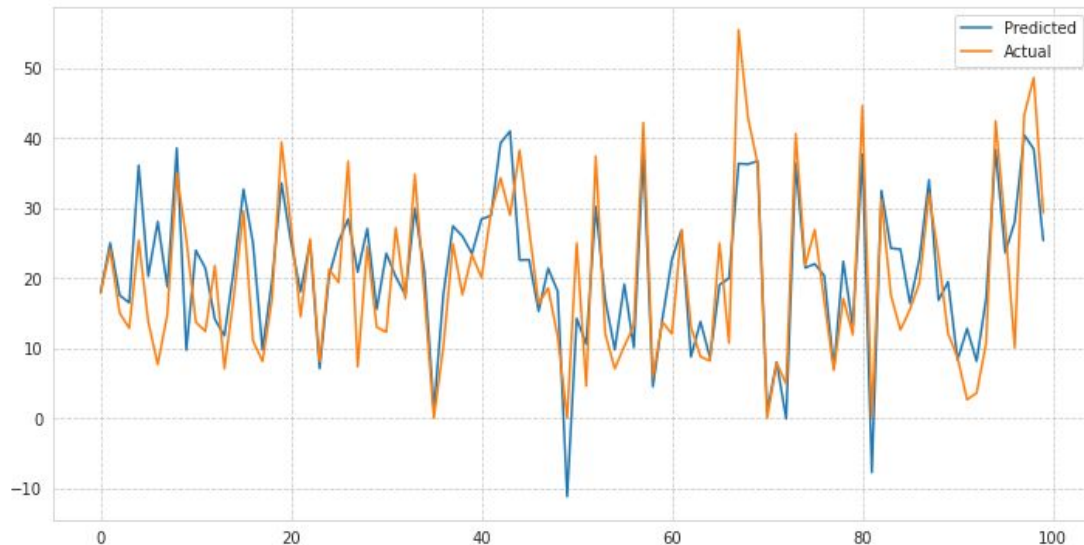
RMSE : 7.330790622062289

MAE : 5.661287586240901

R2 : 0.6501226505752835

Adjusted R2 : 0.6485170948609063

Graph of actual and predicted values



Decision Tree Regression

```
Grid = {"criterion": ["mse", "mae"],  
        "max_depth": [5, 6, 7, 8],  
        "min_samples_split": [10, 20, 40],  
        "min_samples_leaf": [20, 40, 100]}
```

```
Best Parameters = {'criterion': 'mse',  
                  'max_depth': 8,  
                  'min_samples_split': 10,  
                  'min_samples_leaf': 20}
```

Train set Metrics

MSE : 21.56339216605

RMSE : 4.643639969468994

MAE : 3.218932170884279

R2 : 0.8613754775424959

Adjusted R2 : 0.8607393420665914

Test set Metrics

MSE : 24.623417452837717

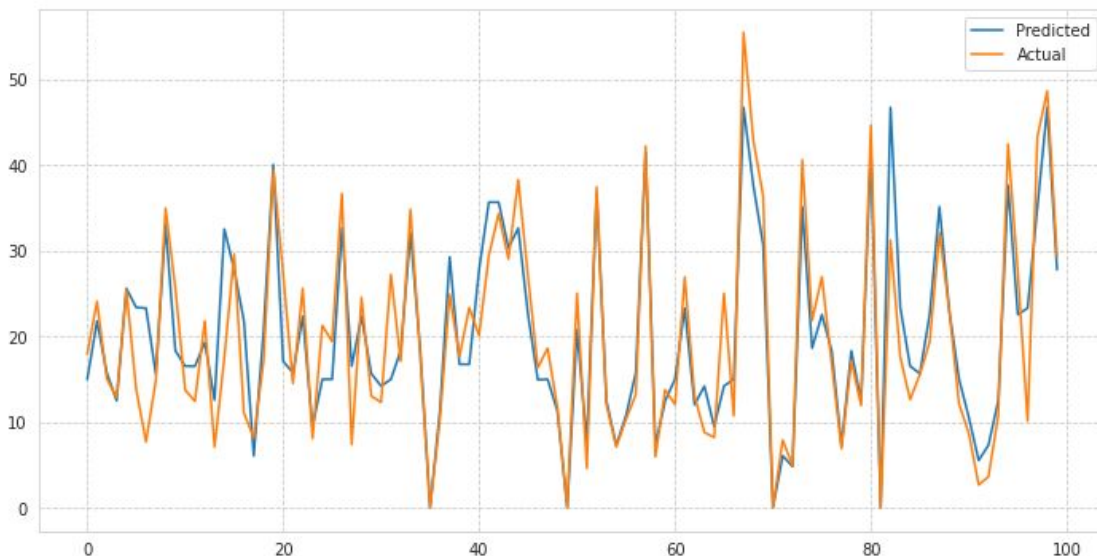
RMSE : 4.962198852609367

MAE : 3.4556336932227647

R2 : 0.8396892948185077

Adjusted R2 : 0.8389536433989369

Graph of actual and predicted values



Random Forest Regression

```
Grid = {'n_estimators': [50, 100, 150],  
       'max_depth': [5, 6, 7, 8],  
       'min_samples_split': [50, 100, 150],  
       'min_samples_leaf': [40, 50]}
```

```
Best Parameters = {'n_estimators': 150,  
                  'max_depth': 8,  
                  'min_samples_split': 100,  
                  'min_samples_leaf': 40}
```

Train set Metrics

MSE : 21.09551559724364

RMSE : 4.592985477578134

MAE : 3.3044192561703136

R2 : 0.8643833144088102

Adjusted R2 : 0.8637609816259826

Test set Metrics

MSE : 24.422199390470553

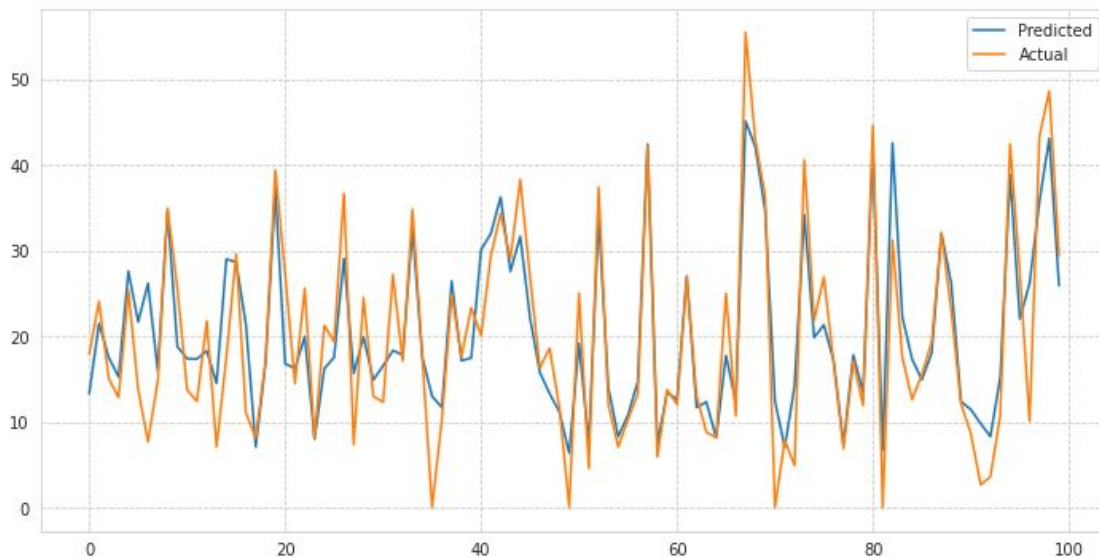
RMSE : 4.941882170840433

MAE : 3.5404879225490267

R2 : 0.8409993245710845

Adjusted R2 : 0.8402696847603208

Graph of actual and predicted values



Gradient Boosting Regression

```
Grid = {'n_estimators': [50, 100, 150],  
       'max_depth': [5, 6, 7, 8],  
       'min_samples_split': [50, 80],  
       'min_samples_leaf': [40, 50]}
```

```
Best Parameters = {'max_depth': 8,  
                  'min_samples_leaf': 40,  
                  'min_samples_split': 50,  
                  'n_estimators': 150}
```

Train set Metrics

MSE : 4.917277342679263

RMSE : 2.217493481992509

MAE : 1.4896685104851668

R2 : 0.9683883120906536

Adjusted R2 : 0.9682432488956585

Test set Metrics

MSE : 10.055935052912352

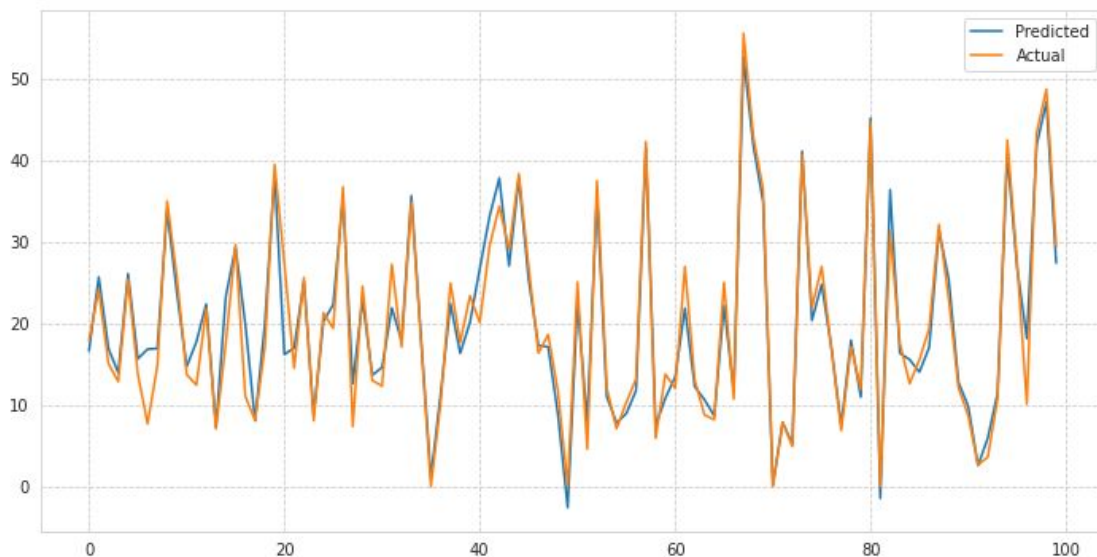
RMSE : 3.1711094356569203

MAE : 2.080602955504996

R2 : 0.9345308569503262

Adjusted R2 : 0.9342304249363316

Graph of actual and predicted values



XGBoost Regression

```
Grid = {'max_depth': [5,6,7,8],  
        'learning_rate': [0.05, 0.1, 0.3],  
        'n_estimators': [80,100,150],  
        'colsample_bytree': [0.5,0.7]}
```

```
Best Parameters = {'colsample_bytree': 0.7,  
                  'learning_rate': 0.1,  
                  'max_depth': 7,  
                  'n_estimators': 150}
```

Train set Metrics

MSE : 2.9450939077909637

RMSE : 1.7161275907667715

MAE : 1.1572887392163702

R2 : 0.9810668825472272

Adjusted R2 : 0.9809800001726829

Test set Metrics

MSE : 9.46132555407195

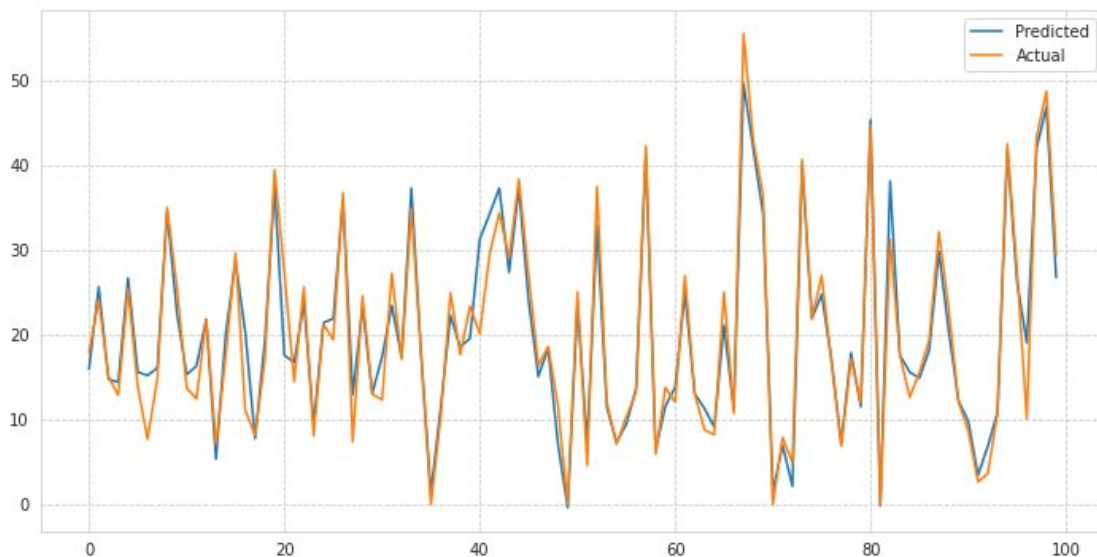
RMSE : 3.0759267796994045

MAE : 2.0349535013806364

R2 : 0.9384020607850212

Adjusted R2 : 0.9381193933775337

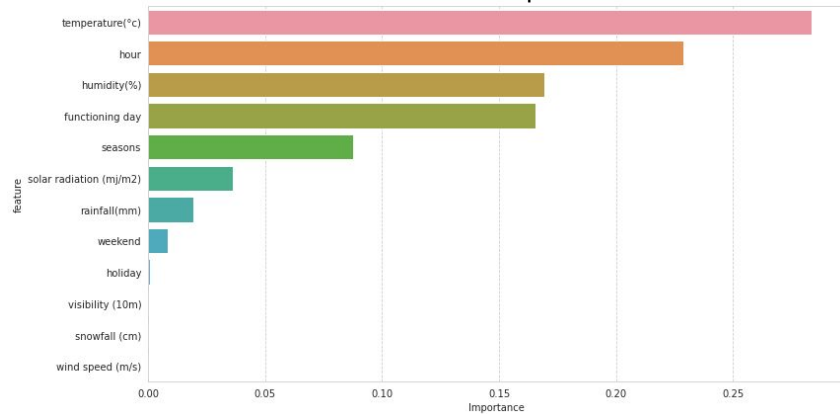
Graph of actual and predicted values



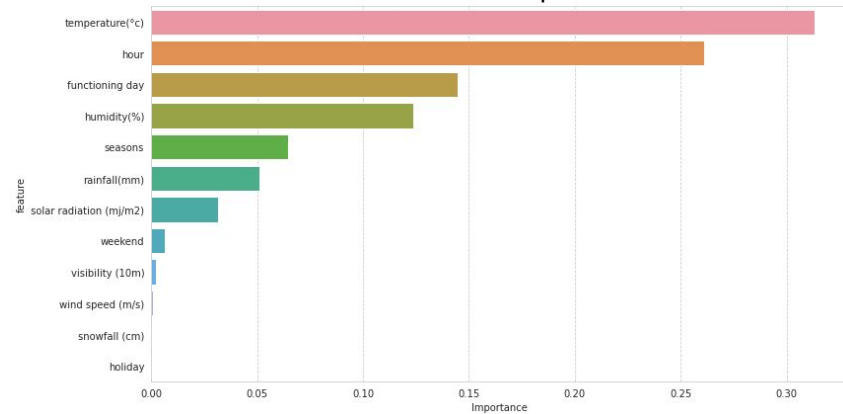
Feature Importance



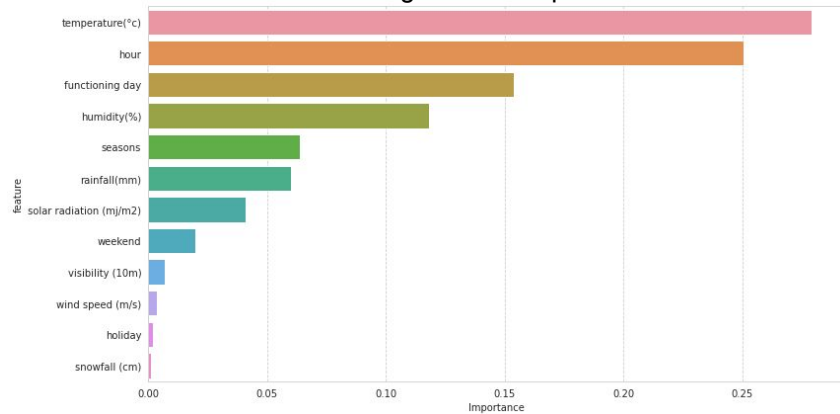
Decision Tree Feature Importance



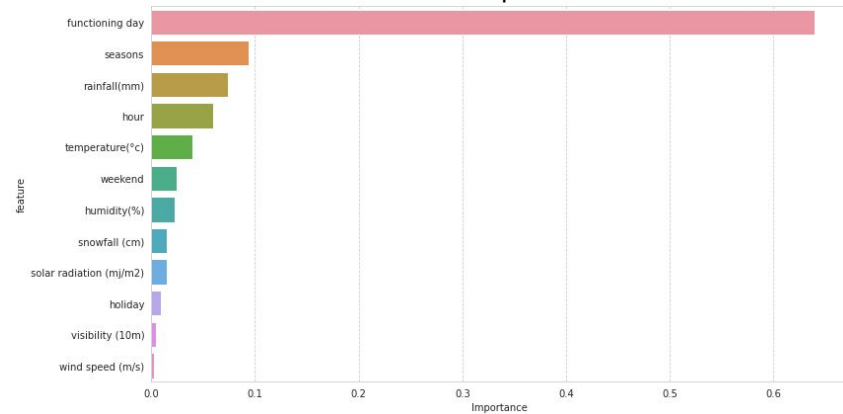
Random Forest Feature Importance



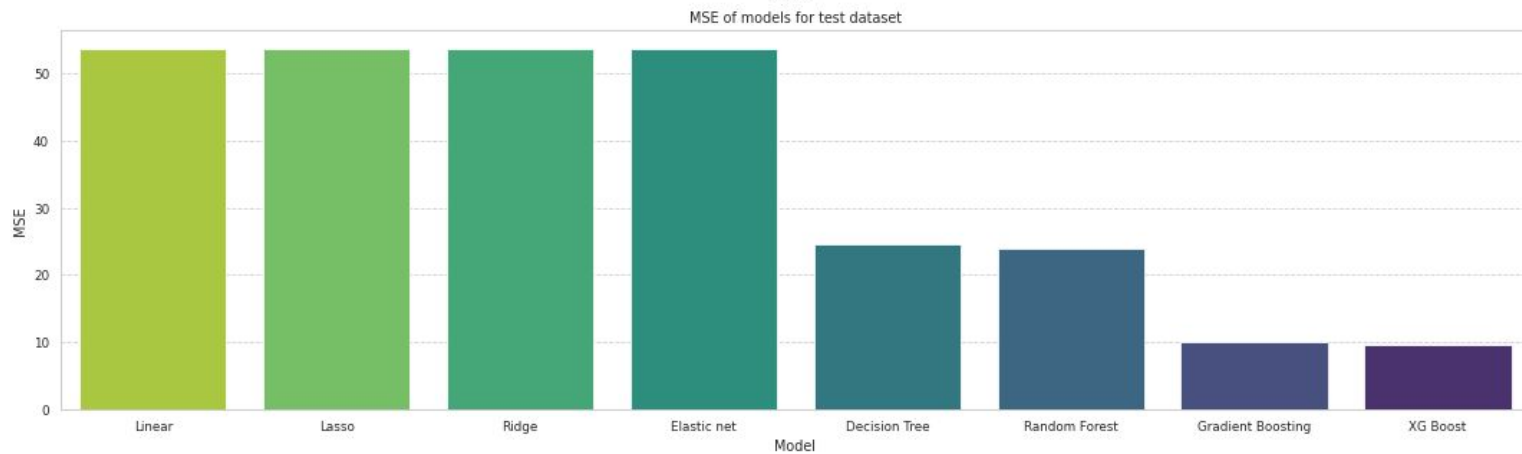
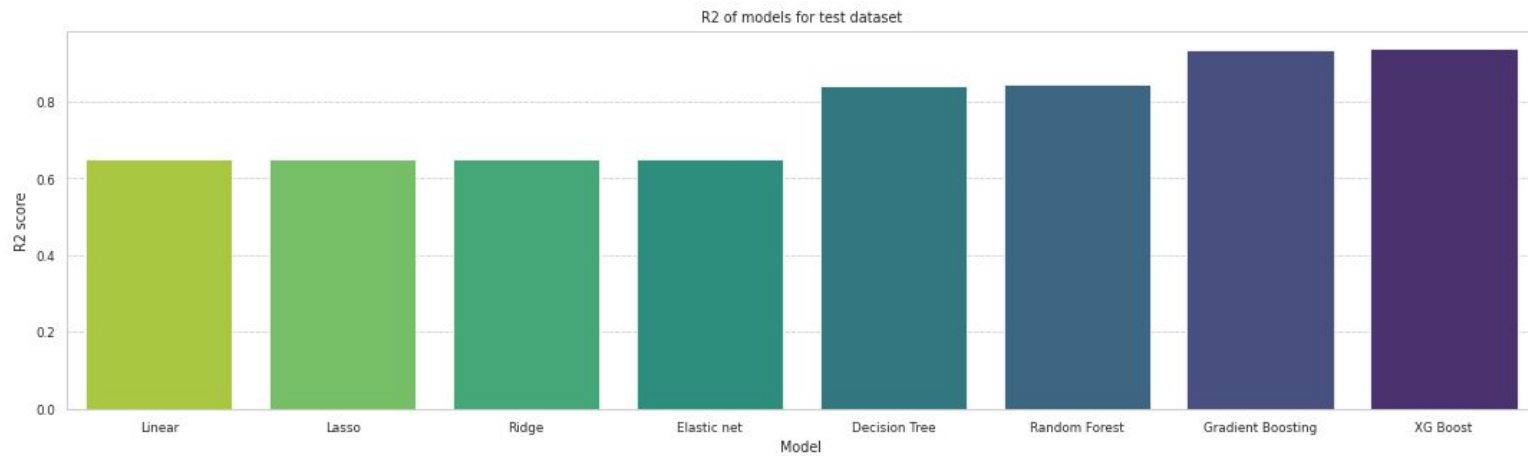
Gradient Boosting Feature Importance



XGBoost Feature Importance



Result



Challenges

- It was time series type data so little difficult to handle.
- Dependent variable had outliers so removing them or not was good point of discussion
- Plotting lots of charts and graphs were needed for data analysis
- Feature selection was important
- Randomly selecting hyperparameters and tuning them was also difficult task.

Conclusion

- There is a huge demand for bike rents in the summer season while the least bike rents occur in winter.
- The demand for rental bikes is at peak at 8am and 6pm so we can say that demand is more during office opening and closing time.
- Temperature and hour are the most influential features to predict the rental bike count.
- Gradient Boosting and XGBoost models are found to be the best models.
- Therefore, either Gradient Boosting or XGBoost model can be used to predict the number of bikes required at each hour.

Q & A