Case Study

By - Pratik Ahuja

1.1 Problem Statement

The objective of this case study is the prediction of bike rental count on daily based on the environmental and seasonal settings. The dataset contains 731 observations, 15 predictors and 1 target variable. The predictors are describing various environment factors and settings like season, humidity etc. We need to build a prediction model to predict estimated count or demand of bikes on a particular day based on the environmental factors.

1.2 Dataset

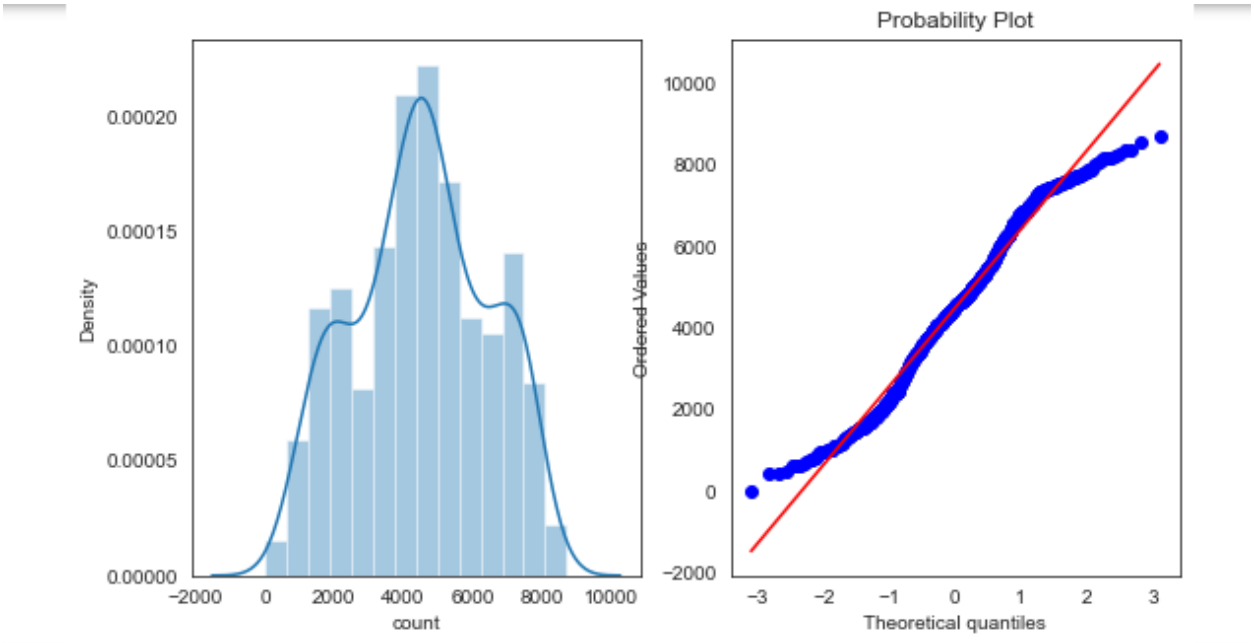
The data set consist of 731 observations recorded over a period of 2 years, between 2011 and 2012. It has 15 predictors or variables and 1 target variable. All the variables are described.

The data set consist of 7 continuous and 8 categorical variables.

2.1 Pre-Processing and EDA

2.1.1 Target Variable – 'count'

The target variable in the problem statement is the total count of registered and casual users of bikes on a single day. '*count'* is the combined value of *'registered'* and *'casual'* variables. The histogram, distribution, and summary statistics of *'count'* are as follow.



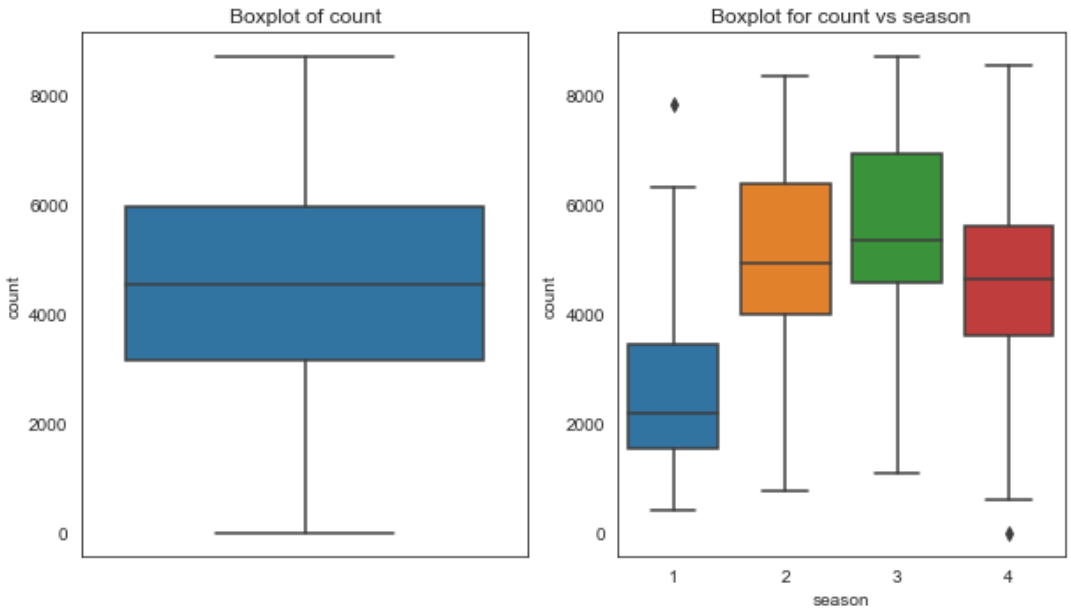
As we can see, out count variable is very close to normal distribution. Preprocessing original data and splitting into train and test data.

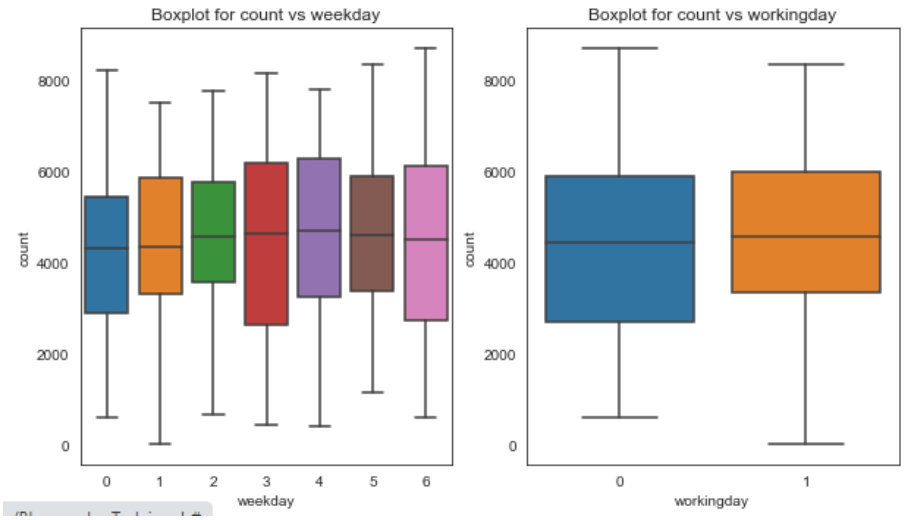
2.1.2 Missing value Analysis

Missing value analysis was performed on the dataset. No missing values were found.

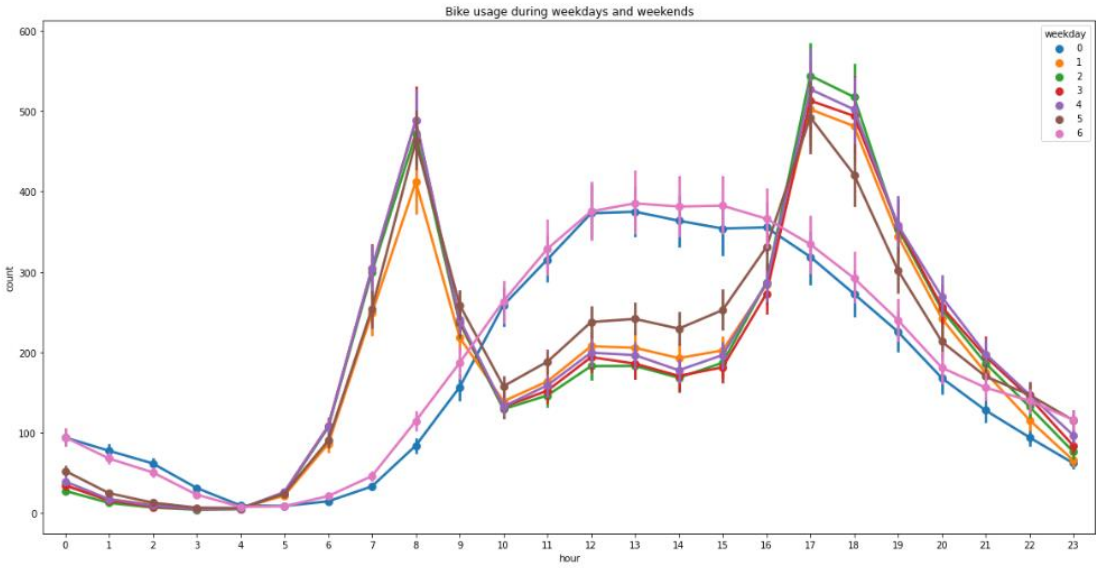
2.1.3 Outlier Analysis

After missing value analysis, we check for outliers in target variable and predictors. There were no outliers present in the dataset. Some extreme values were present in the predictors but those seems too logical. So, no observations were removed, and no imputation was performed on the dataset. Boxplot method was used to check for outliers. Below are the figures from the python implementation.

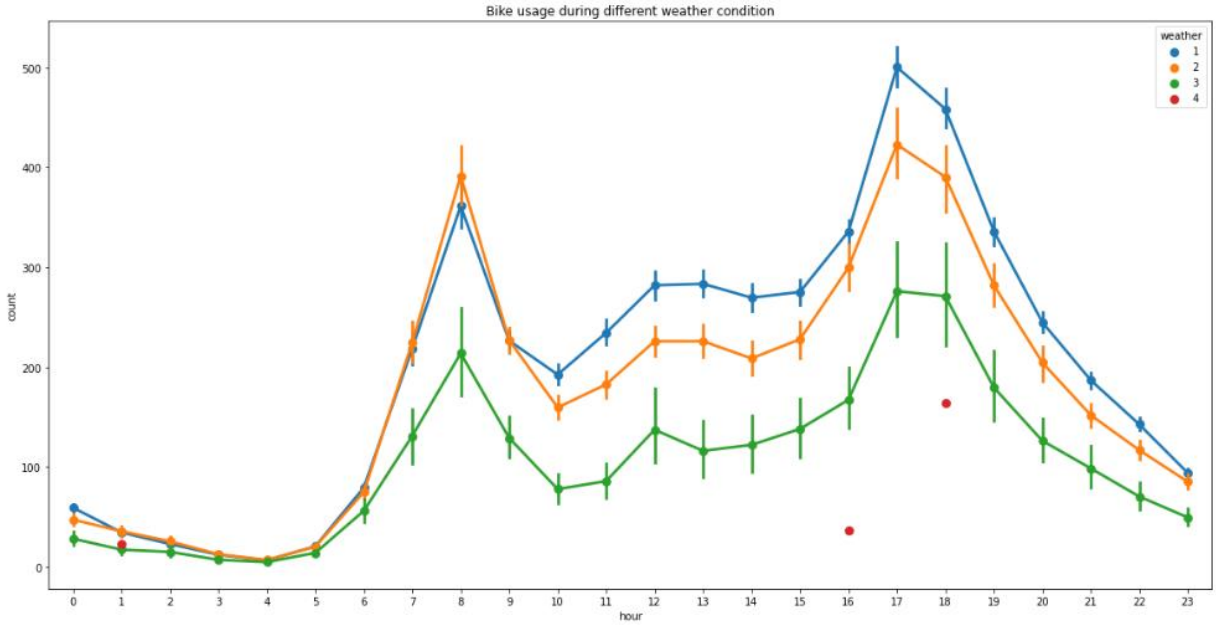




After examining the above boxplots, we can see that there are some extreme values but no outliers.
From these boxplots we can also infer that
1. Bike demand count (‘count’) is low in spring (1) season.
2. There is no effect on bike count(‘count’) due to a holiday or a working day.
3. Bikes are rented mostly in good weather (1: Clear, Few clouds, partly cloudy, partly cloudy) and least in bad (3: Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered clouds) weather.

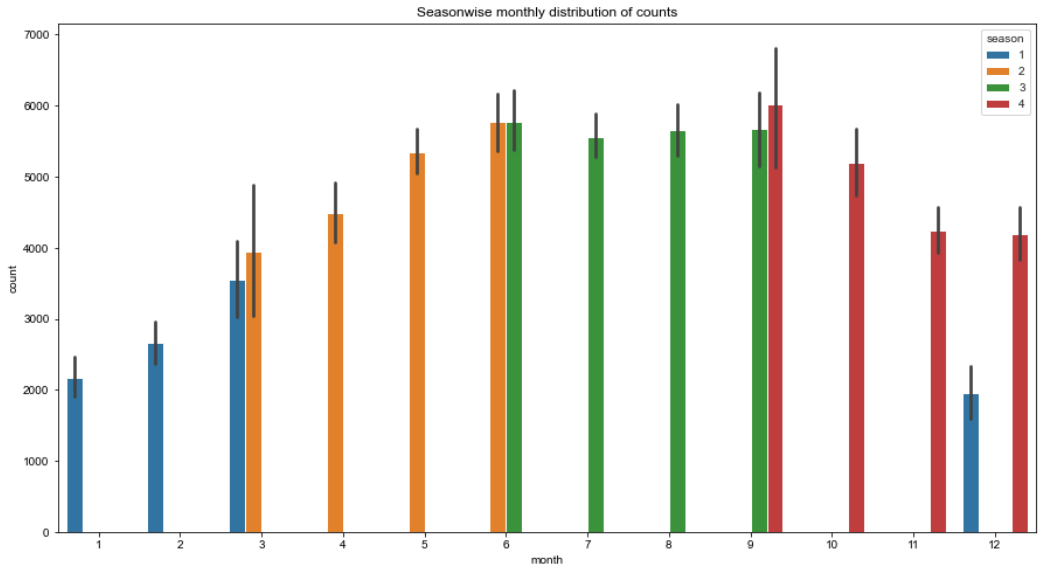


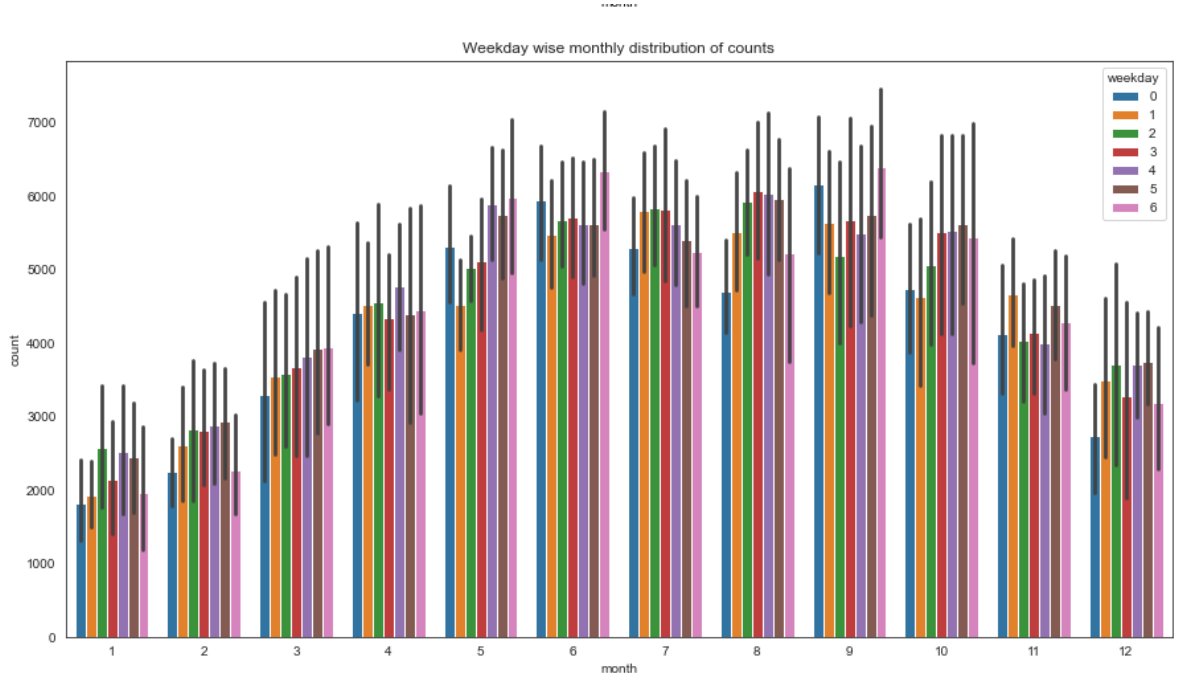
Visualizing the usage of bike during weekday and weekends and the graph shows that the bikes are rented more during weekdays than weekends. The bike is used mostly during office hours.



Visualizing the usage of bike during different weather conditions and the graph shows that the bikes are rented more during weekdays than weekends. The use of bike declines once the weather condition becomes bad and there is almost negligible data during heavy snow and thunderstorm.

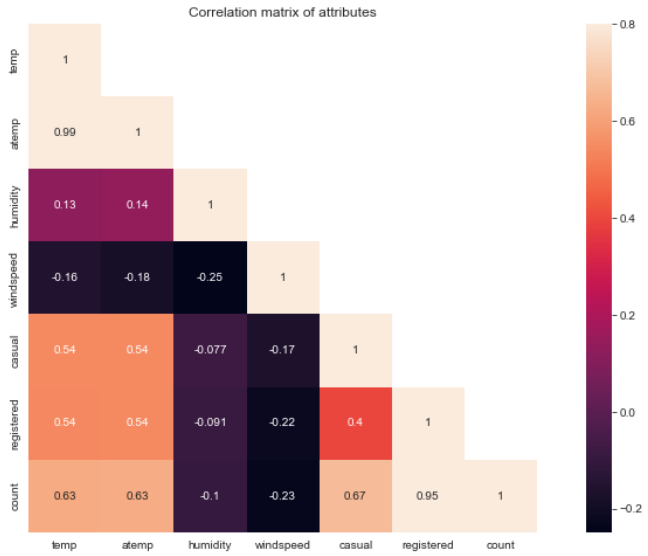
From the below plots, we can observe that increasing the bike rental count in spring and summer season and then decreasing the bike rental count in fall and winter season. Here, season 1-> spring season 2 -> summer season 3 -> fall season 4 -> winter.





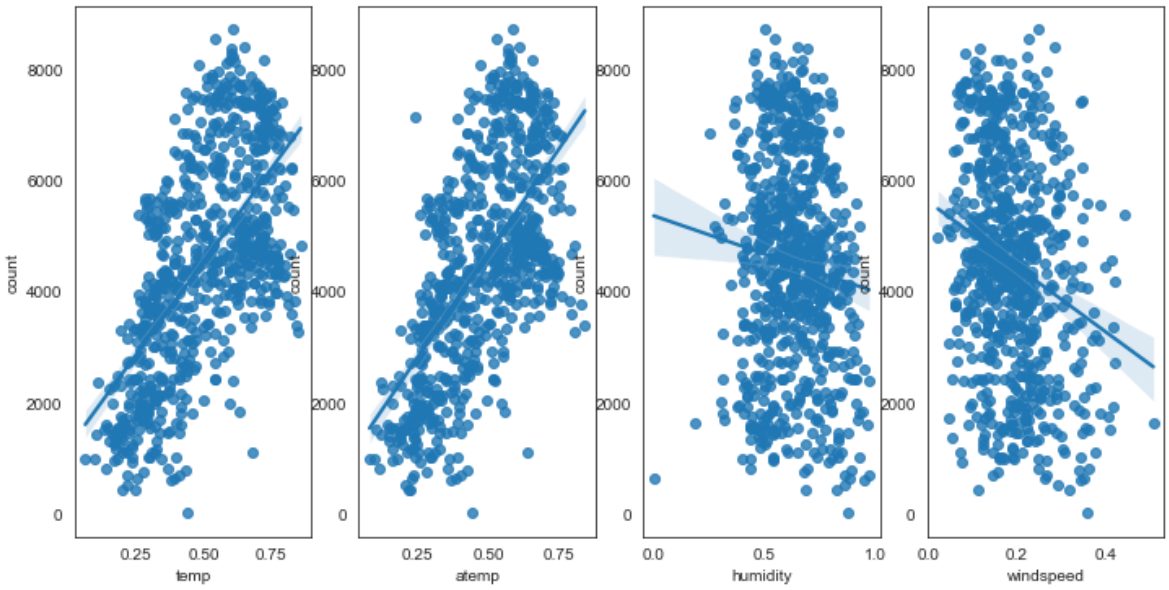
2.1.4 Correlation Analysis.

From correlation plot, we can observe that some features are positively correlated, or some are negatively correlated to each other. The temp and atemp are highly positively correlated to each other, it means that both are carrying same information. The count, casual and registered are highly positively correlated to each other. So, we are going to ignore atemp, casual and registered variable for further analysis.



2.1.5 Bivariate analysis.

From the below plot, it is evident that count has a positive linear relationship with temp and atemp. On the other hand, count has a negative linear relationship with windspeed. Humidity(hum) has a little negative linear relationship with count.



From the above scatter plots, we can see that

1. 'count' and 'temp' have strong and positive relationship. It means that as the temperature rises, the bike demand also increases.
2. 'atemp' and 'count' have strong and positive relationship. It means that as the ambient temperature rise, demand for bikes also increases.
3. 'hum' (humidity) has a negative linear relationship with 'count'. As humidity increases, count decreases.
4. 'windspeed' has negative linear relationship with 'count'. With an increase in windspeed, bike count decreases.

3. Modelling

In bike renting case study, the target variable is continuous in nature. Our task is predicting the bike demand on a single day. This makes it a regression problem. Three machine learning algorithms were used for learning. Both were implemented in python.

1. Multivariate linear regression
2. Random Forest regressor – an ensemble tree-based regression
After EDA and pre-processing steps, data was divided into training and test dataset with 70 % and 30
% Ratio.
3. Decision tree regressor

After modeling, diagnostic plots were used to check the assumptions of best performing model. For performance tuning of random forest, hyperparameter tuning was used. As a result, with proven statistics for Random Forest to be best model below the report it generated with respect to the same and comparison is made in result section.

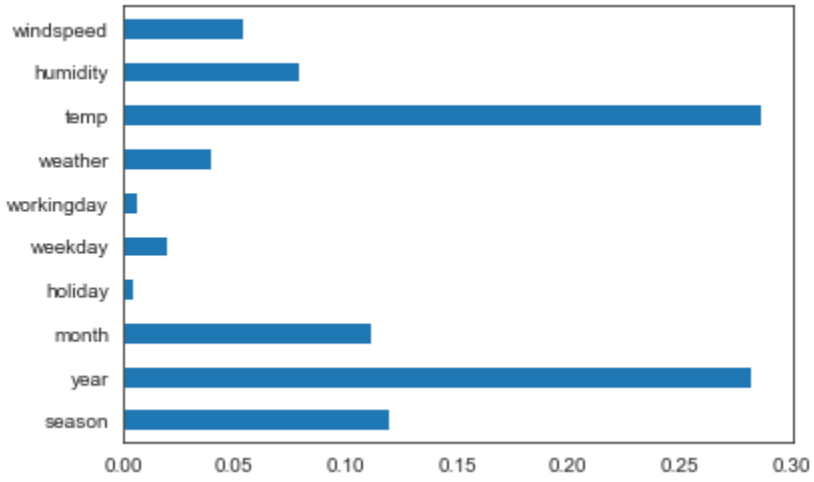
3.1 Python Implementation for Random Forest.

```
import time
np.random.seed(12)
start = time.time()

# Selecting best max_depth, maximum features, split criterion and number of trees
param_dist = {'max_depth': [2,4,6,8,10],
              'bootstrap': [True, False],
              'max_features': ['auto', 'sqrt', 'log2',None],
              "n_estimators" : [100 ,200 ,300 ,400 ,500]
             }
cv_randomForest = RandomizedSearchCV(rf, cv = 10,
                    param_distributions = param_dist,
                    n_iter = 10)

cv_randomForest.fit(training_set, train_taget)
print('Best Parameters using random search: \n',
      cv_randomForest.best_params_)
end = time.time()
print('Time taken in random search: {0: .2f}'.format(end - start))
```

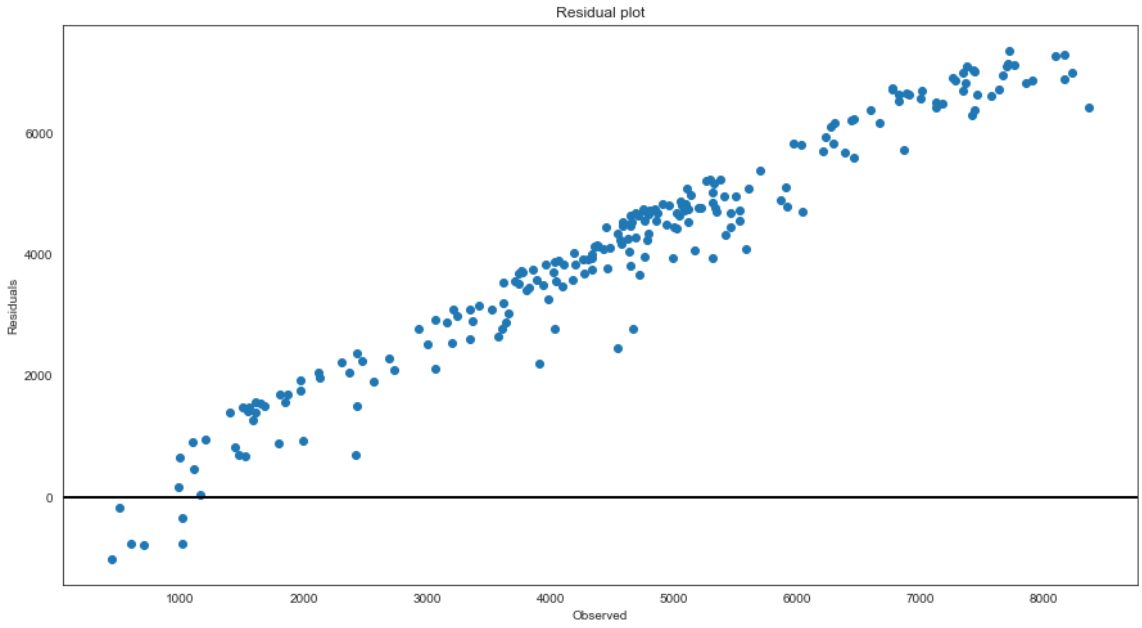
Tuned parameters were selected with hyper tuning random grid search. Best parameters selected were as follows.

```
# Setting parameters

# Set best parameters given by random search

rf.set_params( max_features = 'log2',
               max_depth =8 ,
               n_estimators = 300
                )
```

Variable importance using random forest in python.



Below are the Cross-validation prediction plot talks about finite variance between actual target value and predicted target value. In this plot, some data points have same finite variance between them and for some are not have it.



4. Result and Performance measures.

RMSE (root mean square error) and MAE (mean absolute error) were used as error metric and measuring model performance.

Both the error metric was calculated using python functions. No pre-built package or modules were used. The values for both metrics are given below for all the 3 models and is compared.

Error Metric / Algorithm	Linear Regression	Random Forest	Decision Tree Regressor
MAE	899.5	495	685
RMSE	1222	649	885

4.1 Results.

From the error metric we can see that random forest is performing better than linear regression and decision tree regressor in the implementations. The result for random forest is similar in python.
Selection of model depends on use case. If we want to study the effects of predictors in details, we will go for linear regression and look at the regression equation. If we are care about more precise prediction, we will opt for random forest based on scores for MAE metric as well as RMSE.

Final model for predicting the bike rental count on daily basis. When we compare the root mean squared error and mean absolute error of all 3 models, the random forest model has less root mean squared error and mean absolute error. So, finally I concluded Random Forest model is best for predicting the bike rental count on daily basis. Also comparing the accuracy of the model which is 96% compared with the decision tree regressor of 81% as compared in the code file.

5.Scale up solution,

5.1 Scale up properties of model.

If timeseries is quantized, we can be able to move with the same model for random forest but most probably that is not scalable that is dependent on feature engineering that was made prior to the training.

If feature space mean is reliably constant that means the small to medium datasets is a perfect subsample with same mean and variance of whole population of the dataset or terabytes of data.

If the population can be solved by using fixed window for training or that with exponential average effect of whole history or with the sliding style windows training.

For large data sets, the training data, intermediate computations, and some outputs may be cached on disk using "big. Matrix" objects. This enables random forests to be built on large data sets without hitting RAM limits, which will cause excessive virtual memory swapping by the operating system.

5.2 Random-Forest-Algorithm-using-Hadoop,COMET and Hivemall.

I propose a new algorithm called MR – Random Forest Algorithm to extract the action rules in a distributed processing environment. Hadoop splits the data and gives splits of data to several Map parts (Mappers). The combined action rules are given to the Reduce part, where I propose using a Random Forest type of algorithm to combine the output from all the Mappers. If so, it averages all supports and confidences from these Mappers for the given action rule. Then, it checks the averaged support and confidence against the minimum support and confidence thresholds specified by the user. If the support and confidence thresholds are met, the action rule is retained, and included in the final list of action rules, produced as an output from this system, and presented to the user and in proposed MR-Random Forest Algorithm, implemented in the Reduce part of MapReduce.

Hivemall is a scalable machine learning library that runs on Apache Hive. Hivemall is designed to be scalable to the number of training instances as well as the number of training features. It's mean that I can store data, build model, predict on hadoop.

COMET is particularly suited for large data sets as it splits the input files into blocks of fixed size. However, this implementation only supports classification problems, and therefore is not suitable for handling quantitative traits, and is not open source.

The Random Forest algorithm will be executed on a four-node Hadoop Cluster inputting data of varying sizes at a time. Performance of the algorithm will be measured by analyzing execution Time, Accuracy, Kappa, Reliability and Standard Deviation measures. Results will show that the execution time reduces constantly as the size of dataset increases thus making the system a key tool for processing large datasets. Our work indicates that RF performs better if fed with extracted/selected features by using appropriate feature selection methods.

5.3 Limitations.

The model can be quite challenging to interpret in comparison to a decision tree as we can select by following the tree's path. However, that's not possible in a random forest as it has multiple decision trees in Hadoop ecosystems.

Re-weighting schemes could be used to address the issue of the lack of representativeness for Random Forest and the issue of the sampling bias.

Axis-parallel splitting that may lead to suboptimal trees and the decision tree depth of two that could fail on dataset with greater depth. Because of pre-generation of many random projections, the tree is allowed to grow with more freedom.

5.4 Technology Stack

One of my project consisted with Big data for detecting the anomaly in the car parts consisted with Hadoop,Hive,Pyspark,Oozie for implementation of clustering algorithm and encoders which was counted total of 7-8 Months in the project.