edWisor
Get Skilled.Get Hired

BIKE RENTAL COUNT PREDICTION PROJECT REPORT

Using R and Python

Submitted by: Sanjeev Kumar Pandit
sanjeev110795@gmail.com

Contents

1. Pre requisite And Steps to Execute Codes

Pre requisite:

A. To run the R project in the system should have R &R-studio; Below are the versions of both which are used:
   i) R version 3.6.1 (64 bit).
   ii) R-studio IDE (Version 1.2.5001).
B. To run the Python (ipynb) file in the system should have Anaconda Navigator to be installed Below are the versions of software used:
   i) Jupyter Notebook (V0ersion – 6.0.0)
   ii) Python Version – 3.7.5
C. Download the file and save it in your local. (In my case):
   i) C:\Users\Sanjeev\Desktop\Edwisor\Bike_rent_Prediction_2nd_Project\ Bike_Rent_Count_Prediction_PythonCode.ipynb
   ii) C:\Users\Sanjeev\Desktop\Edwisor\Bike_rent_Prediction_2nd_Project\ Bike_Rent_Count_Prediction_RCode.R
   iii) C:\Users\Sanjeev\Desktop\Edwisor\Bike_rent_Prediction_2nd_Project\day.csv

Steps to execute the In Python:

   i) Python please open Anaconda prompt type Jupyter notebook, load .ipynb file and run the Code.

Steps to execute the R-code:

   i) In R-studio: For In R open R file and run the R-script to get the output in console and plots in Plot window.
   ii) Please make sure that all that library I have used in the code is installed in your system

For Complete project report Open word/ pdf document “Bike rental count prediction. Pdf or . docx”

2. Introduction

1.1. Problem Statement

The objective of this Case is to Predication of bike rental count on daily based on the environmental and seasonal settings. In the bike rental dataset, we need to predict what could be the counting of bike rental for a given season, month and year with environmental conditions like windspeed, temperature, humidity etc., based on the historical data for the two years i.e. 2011 and 2012.

1.2. Data

Understanding of data is the very first and important step in the process of finding solution of any business problem. Let's have a quick preview of data. Will discuss in detail about the data understanding in the CRISP-DM process section.

Let's understand shape of dataset:

```
print("Shape of data is: ",bike_data.shape)
```

```
Shape of data is:  (731, 16)
```

Preview of data:

	instant	dteday	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	casual	registered	cnt
0	1	2011-01-01	1	0	1	0	6	0	2	0.344167	0.363625	0.805833	0.160446	331	654	985
1	2	2011-01-02	1	0	1	0	0	0	2	0.363478	0.353739	0.696087	0.248539	131	670	801
2	3	2011-01-03	1	0	1	0	1	1	1	0.196364	0.189405	0.437273	0.248309	120	1229	1349
3	4	2011-01-04	1	0	1	0	2	1	1	0.200000	0.212122	0.590435	0.160296	108	1454	1562
4	5	2011-01-05	1	0	1	0	3	1	1	0.226957	0.229270	0.436957	0.186900	82	1518	1600

2. Process and Requirement Specification

2.1. Process:

1. Understanding the Data
2. Exploratory Data Analysis
3. Feature Engineering
4. Missing Value and Outlier Treatment
5. Standardizing the data
6. Model Development
7. Hyperparameter Tuning
8. Model selection

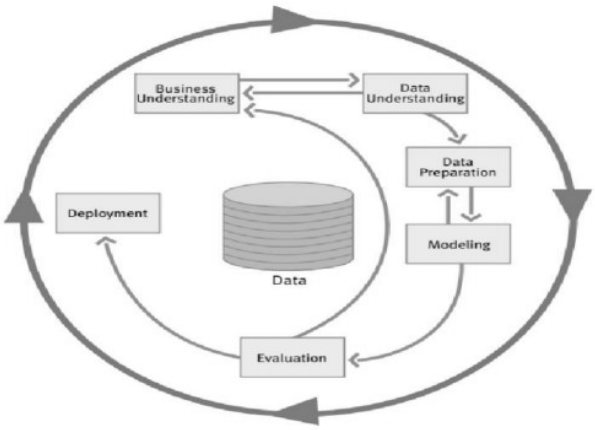
2.2. Software Requirements:

1. Python
2. Jupyter Notebook
3. R
4. R Studio

3. Methodology

CRISP-DM Process: CRISP-DM stands for cross-industry process for data mining. The CRISP-DM methodology provides a structured approach to planning a data mining works. The project follows CRISP DM process to develop the model for the given problem. It involves the following major steps:

1. Business understanding.
2. Data understanding.
3. Data preparation.
4. Modelling.
5. Evaluation.
6. Deployment



3.1. Data Understanding

DATA DICTIONARY:

1. instant: Record index
2. dteday: Date season:
3. Season (1:springer, 2:summer, 3:fall, 4:winter)
4. yr: Year (0: 2011, 1:2012)
5. mnth: Month (1 to 12)
6. hr: Hour (0 to 23)
7. holiday: weather day is holiday or not (extracted fromHoliday Schedule)
8. weekday: Day of the week
9. workingday: If day is neither weekend nor holiday is 1, otherwise is 0.
10. weathersit: (extracted fromFreemeteo) 1: Clear, Few clouds, Partly cloudy, Partly cloudy 2: Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist 3: Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered clouds 4: Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog
11. temp: Normalized temperature in Celsius. The values are derived via (t-t_min)/(t_max-t_min), t_min=-8, t_max=+39 (only in hourly scale)
12. atemp: Normalized feeling temperature in Celsius. The values are derived via (t-t_min)/(t_maxt_min), t_min=-16, t_max=+50 (only in hourly scale)
13. hum: Normalized humidity. The values are divided to 100 (max)
14. windspeed: Normalized wind speed. The values are divided to 67 (max)
15. casual: count of casual users
16. registered: count of registered users
17. cnt: count of total rental bikes including both casual and registered

3.2. Data Pre-Processing:

Data pre-processing is a data mining technique that involves transforming raw data into an understandable format. Real-world data is often incomplete, inconsistent, and/or lacking in certain behaviour trends, and is likely to contain many errors. Data pre-processing is a proven method of resolving such issues. Data pre-processing prepares raw data for further processing.

Data pre-processing is defining the quality of data and the useful information that can be derived from it directly affects the ability of our model to learn; therefore, it is extremely important that we pre-process our data before feeding it into our model. As we already know the quality of our inputs decide the quality of our output.

So, once we have got our business hypothesis ready, it makes sense to spend lot of time and efforts here. Approximately, data exploration, cleaning and preparation can take up to 60-70% of our total project time. This process is often called as Exploratory Data Analysis Listed below are data pre-processing techniques used for this Project

1) Data Exploration and Cleaning

2) Missing Value Analysis

3) Outlier analysis

4) Feature Engineering

5) Feature Selection

6) Feature Scaling

Let’s discuss /explain this technique in detail.

3.2.1. Data Exploration and Cleaning

Data Pre-processing is the very first step which comes with any data science project is data exploration and cleaning,

In Python:

	instant	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	atemp
count	731.000000	731.000000	731.000000	731.000000	731.000000	731.000000	731.000000	731.000000	731.000000	731.000000
mean	366.000000	2.496580	0.500684	6.519836	0.028728	2.997264	0.683995	1.395349	0.495385	0.474354
std	211.165812	1.110807	0.500342	3.451913	0.167155	2.004787	0.465233	0.544894	0.183051	0.162961
min	1.000000	1.000000	0.000000	1.000000	0.000000	0.000000	0.000000	1.000000	0.059130	0.079070
25%	183.500000	2.000000	0.000000	4.000000	0.000000	1.000000	0.000000	1.000000	0.337083	0.337842
50%	366.000000	3.000000	1.000000	7.000000	0.000000	3.000000	1.000000	1.000000	0.498333	0.486733
75%	548.500000	3.000000	1.000000	10.000000	0.000000	5.000000	1.000000	2.000000	0.655417	0.608602
max	731.000000	4.000000	1.000000	12.000000	1.000000	6.000000	1.000000	3.000000	0.861667	0.840896

hum	windspeed	casual	registered	cnt
731.000000	731.000000	731.000000	731.000000	731.000000
0.627894	0.190486	848.176471	3656.172367	4504.348837
0.142429	0.077498	686.622488	1560.256377	1937.211452
0.000000	0.022392	2.000000	20.000000	22.000000
0.520000	0.134950	315.500000	2497.000000	3152.000000
0.626667	0.180975	713.000000	3662.000000	4548.000000
0.730209	0.233214	1096.000000	4776.500000	5956.000000
0.972500	0.507463	3410.000000	6946.000000	8714.000000

In R:

```
> summary(bike_data)
    instant              dteday       season          yr              mnth      
 Min.   :  1.0   2011-01-01:  1   Min.   :1.000   Min.   :0.0000   Min.   : 1.00  
 1st Qu.:183.5   2011-01-02:  1   1st Qu.:2.000   1st Qu.:0.0000   1st Qu.: 4.00  
 Median :366.0   2011-01-03:  1   Median :3.000   Median :1.0000   Median : 7.00  
 Mean   :366.0   2011-01-04:  1   Mean   :2.497   Mean   :0.5007   Mean   : 6.52  
 3rd Qu.:548.5   2011-01-05:  1   3rd Qu.:3.000   3rd Qu.:1.0000   3rd Qu.:10.00  
 Max.   :731.0   2011-01-06:  1   Max.   :4.000   Max.   :1.0000   Max.   :12.00  
                 (Other)   :725                                                   
    holiday           weekday        workingday      weathersit   
 Min.   :0.00000   Min.   :0.000   Min.   :0.000   Min.   :1.000  
 1st Qu.:0.00000   1st Qu.:1.000   1st Qu.:0.000   1st Qu.:1.000  
 Median :0.00000   Median :3.000   Median :1.000   Median :1.000  
 Mean   :0.02873   Mean   :2.997   Mean   :0.684   Mean   :1.395  
 3rd Qu.:0.00000   3rd Qu.:5.000   3rd Qu.:1.000   3rd Qu.:2.000  
 Max.   :1.00000   Max.   :6.000   Max.   :1.000   Max.   :3.000  

      temp             atemp              hum          windspeed      
 Min.   :0.05913   Min.   :0.07907   Min.   :0.0000   Min.   :0.02239  
 1st Qu.:0.33708   1st Qu.:0.33784   1st Qu.:0.5200   1st Qu.:0.13495  
 Median :0.49833   Median :0.48673   Median :0.6267   Median :0.18097  
 Mean   :0.49538   Mean   :0.47435   Mean   :0.6279   Mean   :0.19049  
 3rd Qu.:0.65542   3rd Qu.:0.60860   3rd Qu.:0.7302   3rd Qu.:0.23321  
 Max.   :0.86167   Max.   :0.84090   Max.   :0.9725   Max.   :0.50746  

     casual         registered        cnt      
 Min.   :   2.0   Min.   :  20   Min.   :  22  
 1st Qu.: 315.5   1st Qu.:2497   1st Qu.:3152  
 Median : 713.0   Median :3662   Median :4548  
 Mean   : 848.2   Mean   :3656   Mean   :4504  
 3rd Qu.:1096.0   3rd Qu.:4776   3rd Qu.:5956  
 Max.   :3410.0   Max.   :6946   Max.   :8714  
```

3.2.2.Missing Value Analysis

In real world missing value is the common occurrence of incomplete observations in an asset of data. Missing values can arise due to many reasons, error in uploading data, unable to measure an observation etc. Due to presence of missing values in the form of 0, NA or NAN. These will affect the accuracy of model which we are building. Hence it is necessary to check for any missing values in the given data. Let's check of the dataset for missing values and identify what type are they? Listed below are some of the missing values in different forms:

- Values containing '0'- These will first need to be changed to NA and Nan in R and python respectively.
- Blank spaces that are taken as NA and NaN in R and Python respectively.

Depending on the percentage of missing values we can decide if we need to keep a variable or drop it based on the following conditions:

- Missing value percentage < 30% - We can include the variable
- Missing value percentage > 30 % - We need to remove those variable/s because even if we impute values, they are not the actual values, the variable will not contain actual values and hence will not contribute effectively to our model.

Let's check of the dataset for missing values and identify what type are they? In the given data there is no any missing values. So, we need not impute any missing values

In Python:

Missing Value anlysis

```
# checking for missing values in dataset
bike_data.isnull().sum()
```

```
instant         0
dteday          0
season          0
year            0
month           0
holiday         0
weekday         0
workingday      0
weather         0
temperature     0
atemp           0
humidity        0
windspeed       0
casual          0
registered      0
count           0
dtype: int64
```

In R:

```
> sum(is.na(bike_data))
[1] 0
```

3.2.3. Delete unnecessary predictors

A. Delete instant variable as it is a kind of index in dataset
B. The aim of the project is to predict the count based on environmental and seasonal settings not date basis. So, Delete the date column from dataset.
C. As we are interested on the count and count = casual + registered,
So, I am going to delete the casual and registered column from dataset to avoid redundant variable from dataset as it doesn't carry any useful condition and it may affect the model performance.

Shape of data after removing un-necessary predictors

In Python:

```
# Lets check dimensions of data after removing unnecessary variables
bike_data.shape
```

```
(731, 12)
```

In R:

```
> dim(bike_data)
[1] 731  12
```

3.2.4. Outlier Analysis and Exploratory data analysis

An Outlier is any inconsistent or abnormal observation in a variable of dataset that deviates from the rest of the observations.

These inconsistent observations can be due to manual error, poor quality/ malfunctioning equipment's, Experimental error or correct but exceptional data based on business use case.

It can cause an error in predicting the target variable/s. Hence, we need to check for the outliers and either remove the observations containing them or replace them with NA, then impute or set upper limits/lower limits or mean/median values imputation.

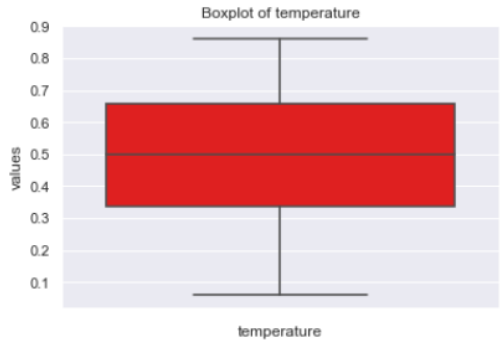
For Outlier analysis, Boxplot to visualize and summary descriptive statistics to check range of each numeric variable and sorted the variables to detect some strange values like zeros and negative values.

Exploratory data analysis is an approach to analyzing data sets to summarize their main characteristics, often with visual methods. A statistical model can be used or not, but primarily EDA is for seeing what the data can tell us beyond the formal modelling or hypothesis testing task

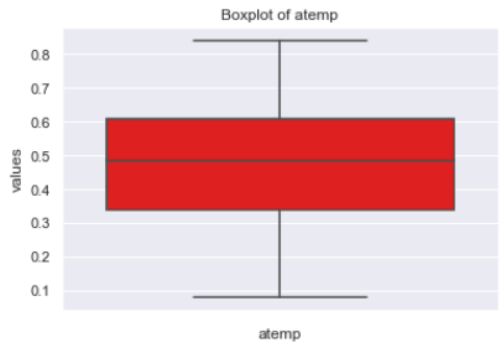
Plot boxplot to visualize outliers in continuous variable

In Python:

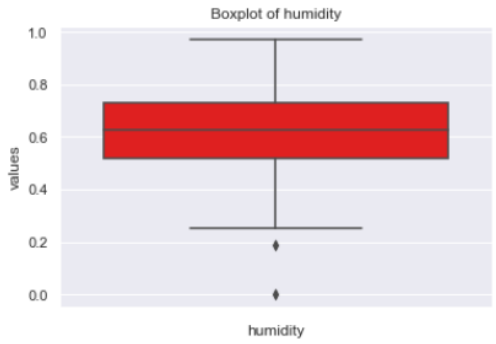
temperature



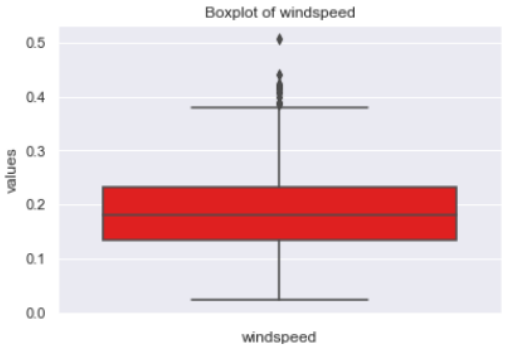
atemp



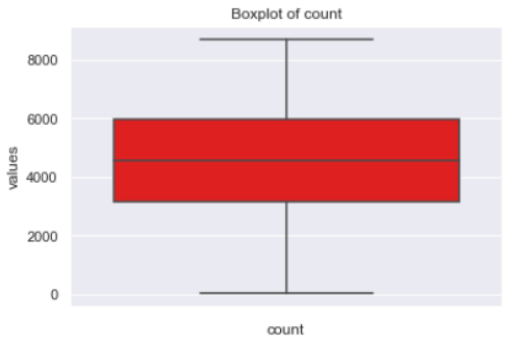
humidity



windspeed

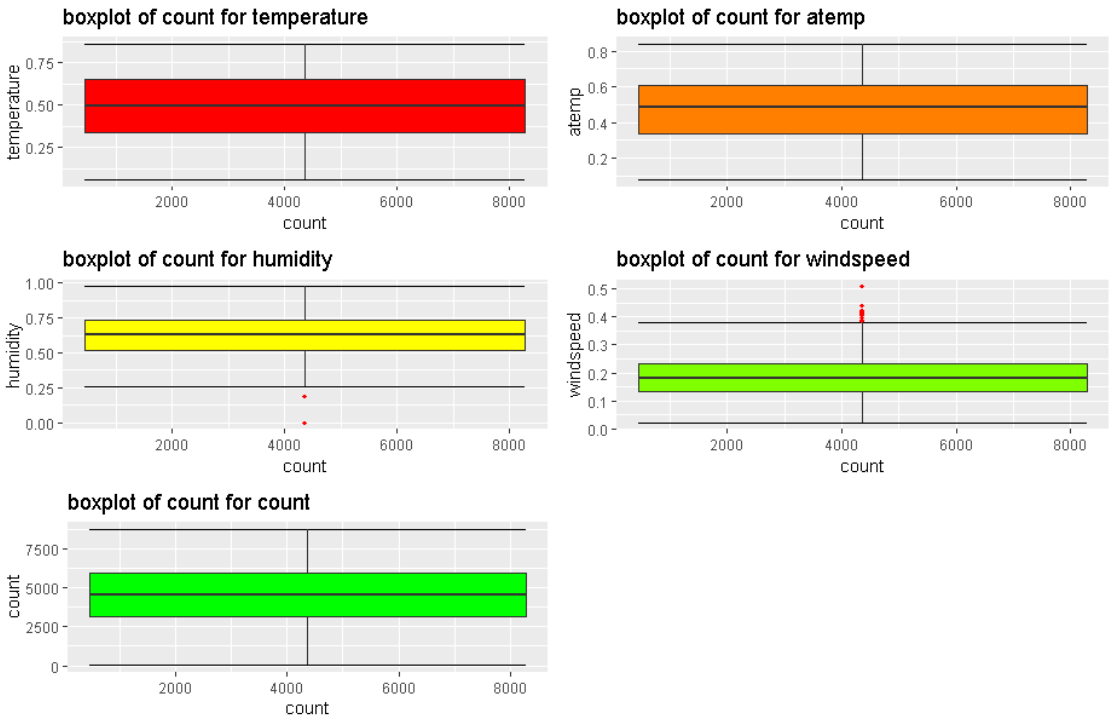


count



From boxplot we can see outliers in humidity and outliers in windspeed predictors.

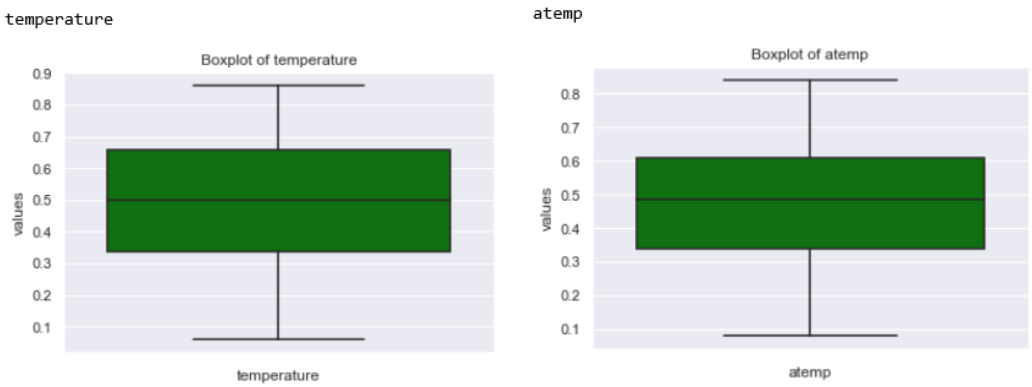
In R:



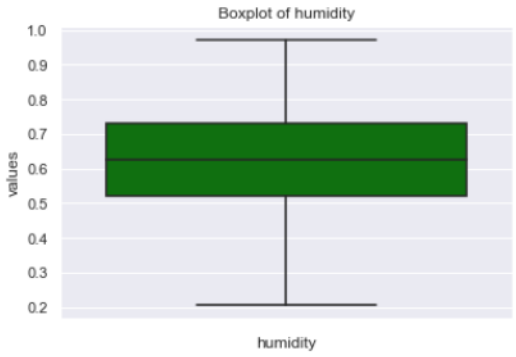
For this project I have opted for capping method in which we are going to impute outlier with upper fence and lower fence value reason behind to opt this method is we don't want to delete the observations with outliers as data collection is also a crucial step in data analytics for which client has to spend more money if don't have any past data specially for start-up companies .by keeping this point into consideration I tried to retain the data wherever possible in the pre-processing. Boxplot stat method is one of the methods to remove outlier and we also learnt we can also treat these outliers as missing values and impute them with mean or median or knn method or delete such observations.

Box Plot after removing outliers

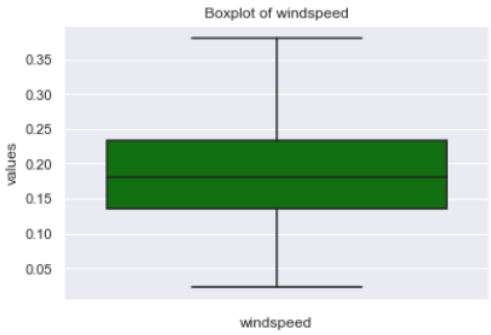
In Python:



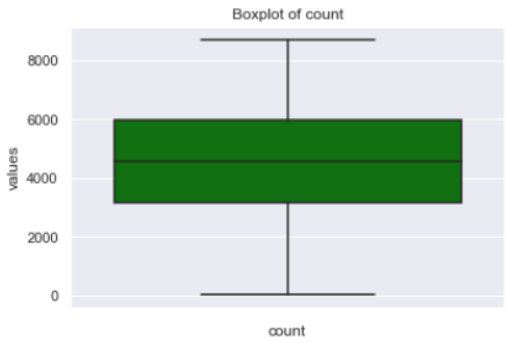
humidity



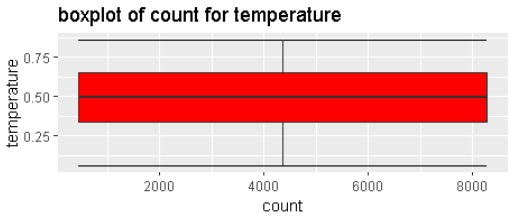
windspeed

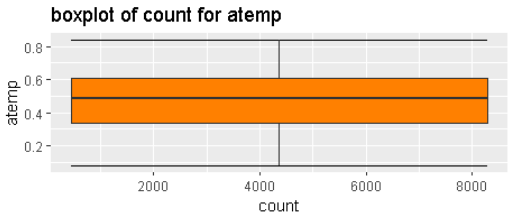


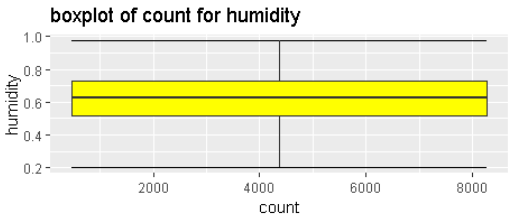
count

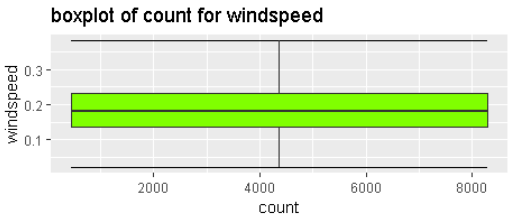


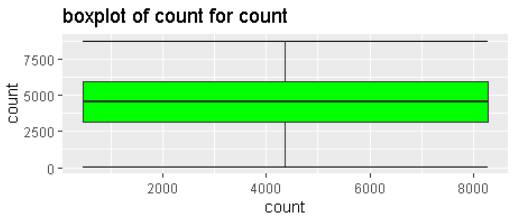
In R:











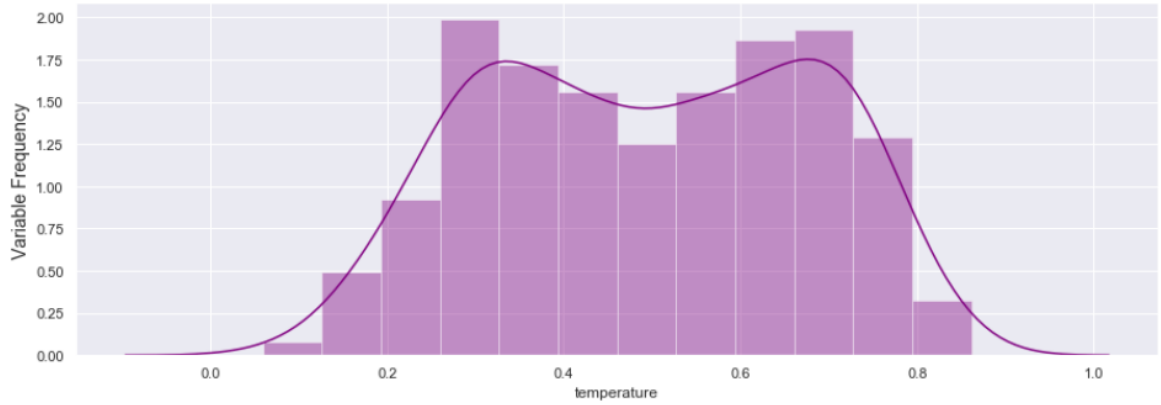
3.2.5. Feature Engineering

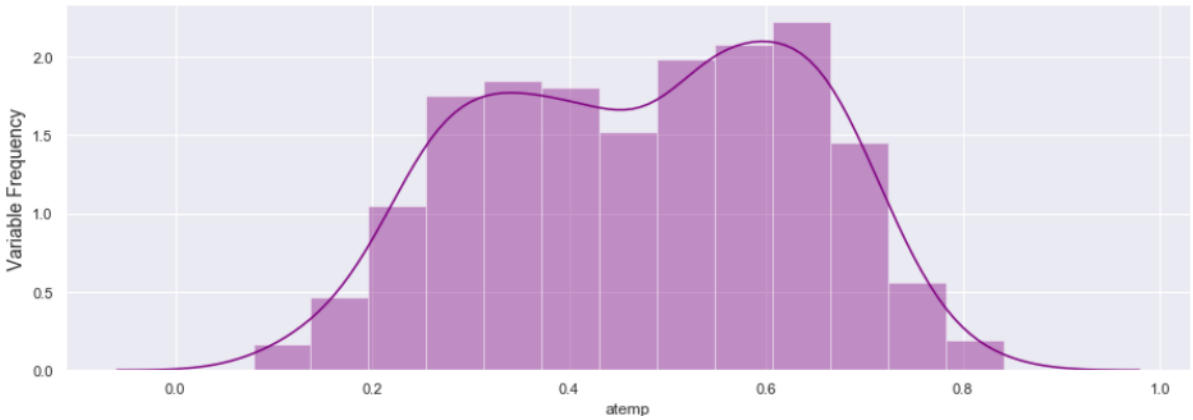
Feature engineering is the science (and art) of extracting more information from existing data, not adding any new data to it, but making the data more meaningful and usable. In our Project, No need to add any new/feature variable to our data.

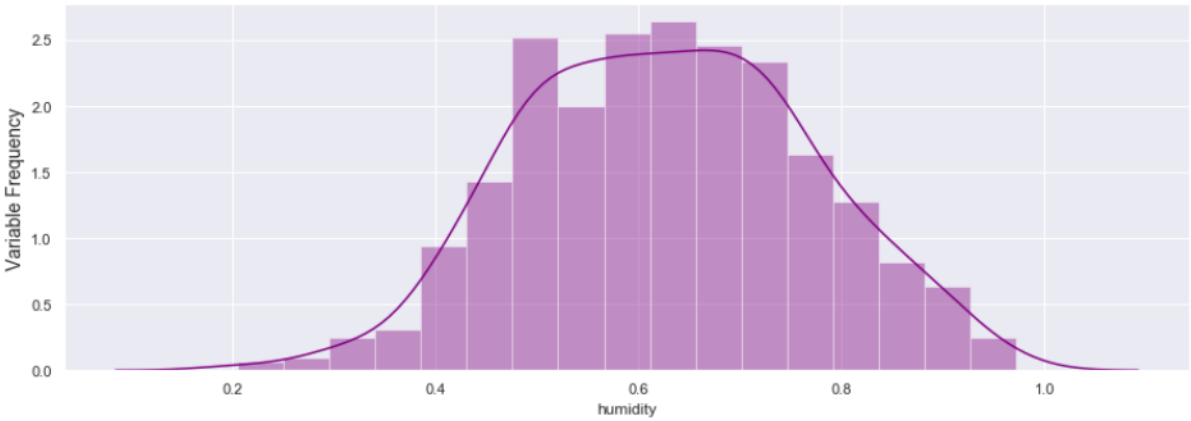
3.2.6. Frequency Distribution Visualization

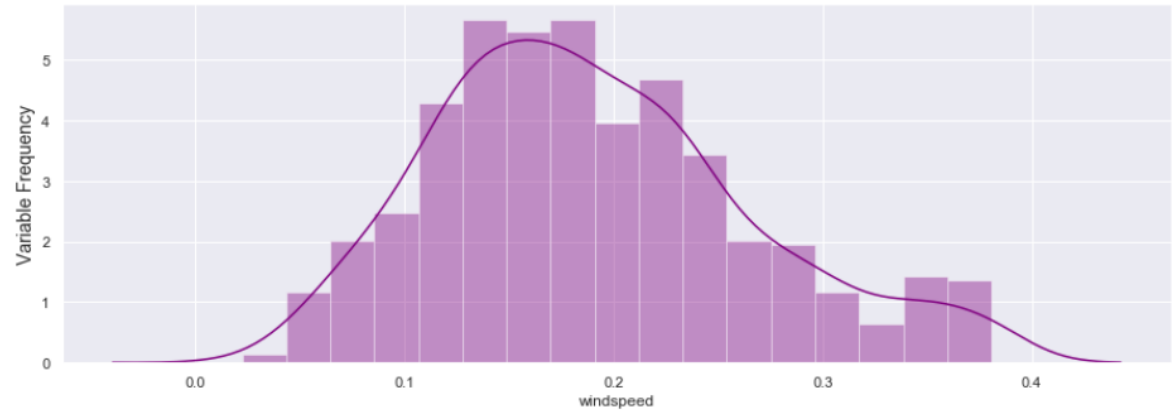
To check distribution of each continuous variable we plotted histogram for each variable Both in R and Python, we can also check distribution using summary or describe function. In our project it can be observed that from the below plot we can say predictor are normally distributed.

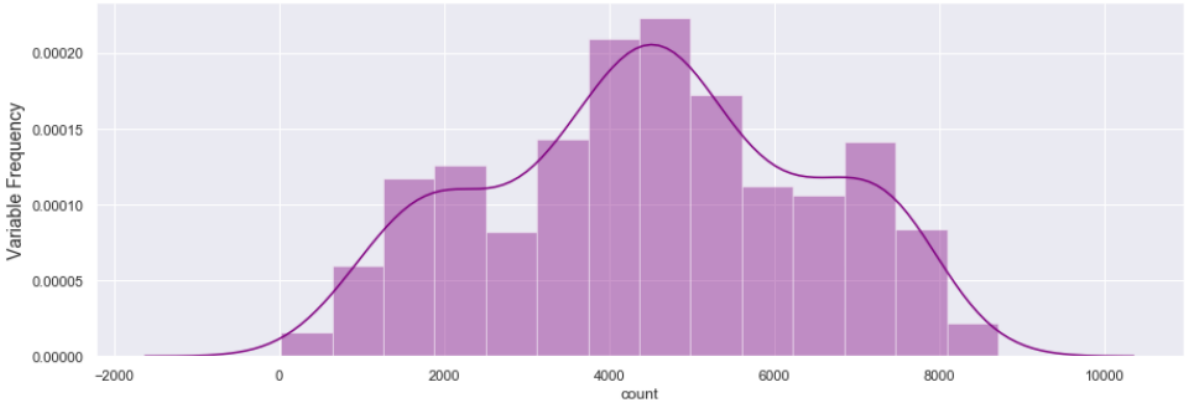
In Python:



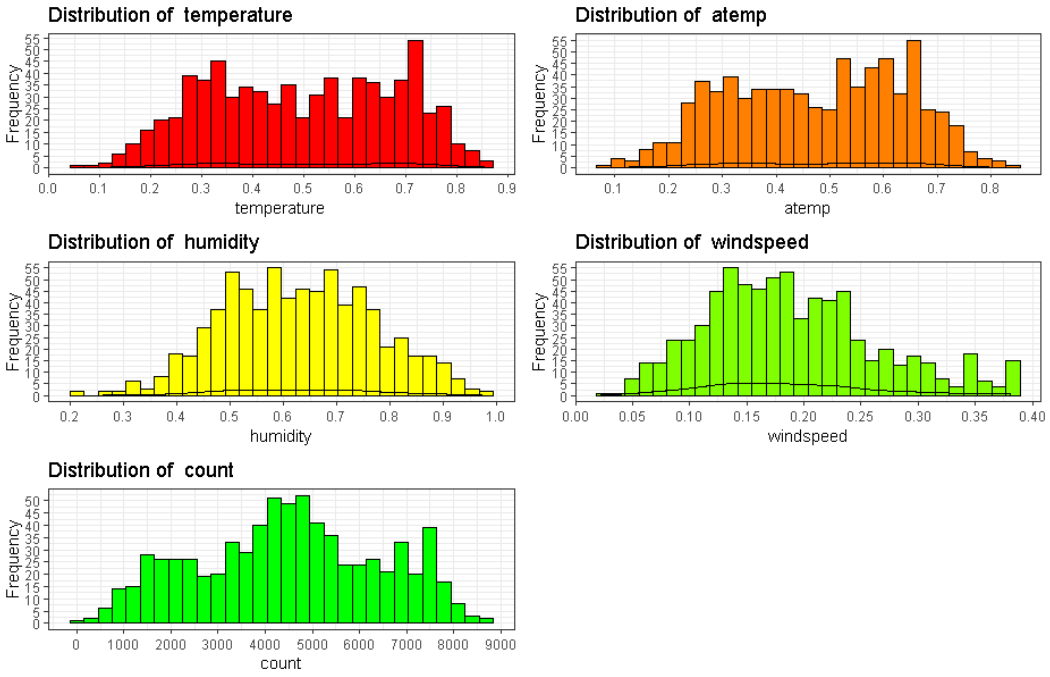





Variable Frequency
0
1
2
3
4
5
0.0
0.1
0.2
0.3
0.4
windspeed


Variable Frequency
0.00000
0.00005
0.00010
0.00015
0.00020
−2000
0
2000
4000
6000
8000
10000
count

In R:


Distribution of temperature
Frequency
temperature
Distribution of atemp
Frequency
atemp
Distribution of humidity
Frequency
humidity
Distribution of windspeed
Frequency
windspeed
Distribution of count
Frequency
count

Effect of categorical variables on target variable

To check distribution of each categorical variable with respect to target variable we used bar plot and we can also look at the frequency table

Count Vs season: Bike rent count is high in season 3(fall) and low in season1(springer)

Count Vs year: Bike rent count is high in year 1 (in 2012)

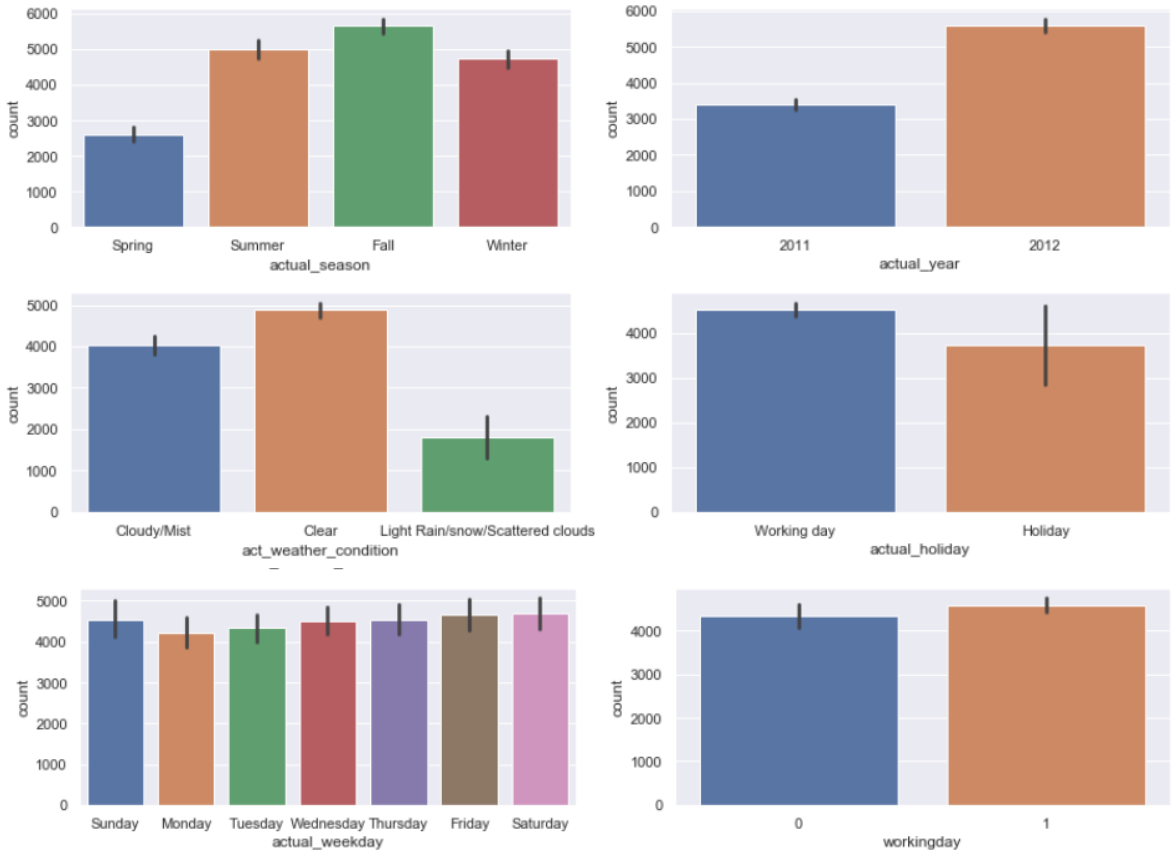
Count Vs month: Bike rent count is high in month of august and low in Jan

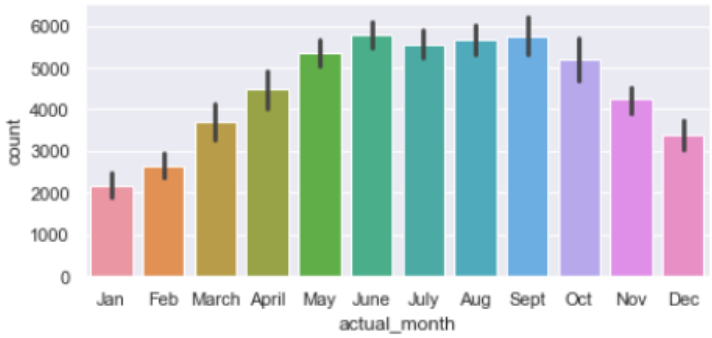
Count Vs holiday: - Bike rent count is high on holidays i.e., 0

Count Vs weekday: - From bar plot we can see maximum bikes rented on saturday and least bikes on monday.

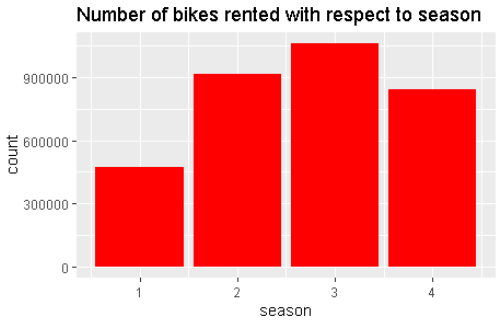
Count Vs working day: - Bike rent count is high on working day i.e., 1

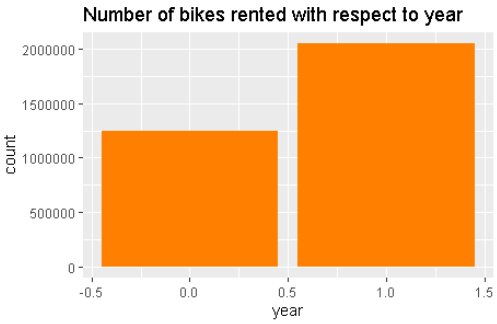
Count Vs weather: - Bike rent count is high on weather clear: i.e., when the weather is Clear, Few clouds, Partly cloudy, Partly cloudy

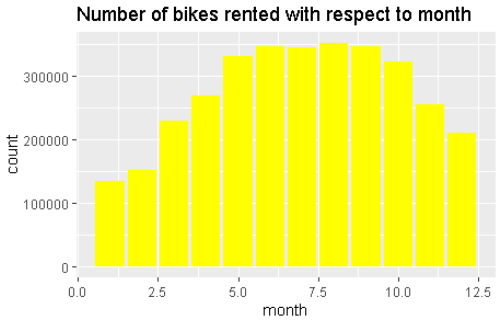


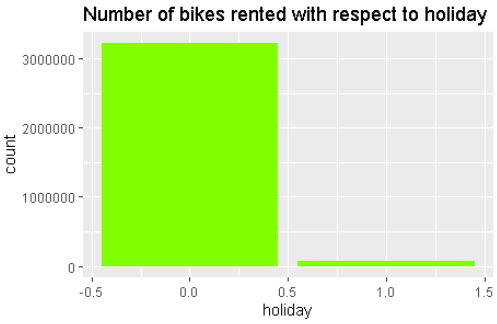

count
6000
5000
4000
3000
2000
1000
0
Jan
Feb
March
April
May
June
July
Aug
Sept
Oct
Nov
Dec
actual_month

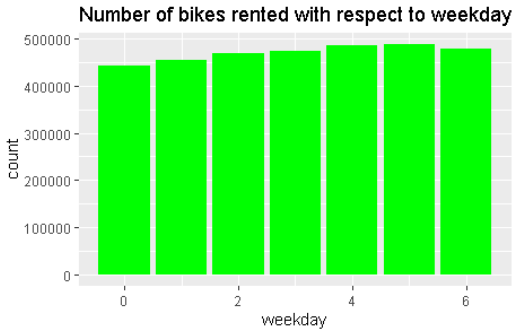
In R:

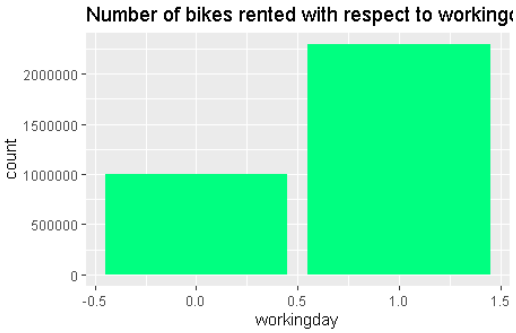

Number of bikes rented with respect to season
count
900000
600000
300000
0
1
2
3
4
season

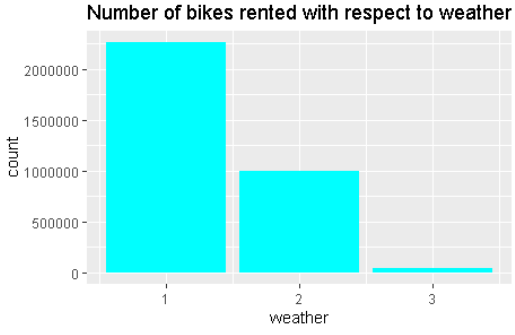

Number of bikes rented with respect to year
count
2000000
1500000
1000000
500000
0
-0.5
0.0
0.5
1.0
1.5
year


Number of bikes rented with respect to month
count
300000
200000
100000
0
0.0
2.5
5.0
7.5
10.0
12.5
month


Number of bikes rented with respect to holiday
count
3000000
2000000
1000000
0
-0.5
0.0
0.5
1.0
1.5
holiday


Number of bikes rented with respect to weekday
count
500000
400000
300000
200000
100000
0
0
2
4
6
weekday


Number of bikes rented with respect to working
count
2000000
1500000
1000000
500000
0
-0.5
0.0
0.5
1.0
1.5
workingday


Number of bikes rented with respect to weather
count
2000000
1500000
1000000
500000
0
1
2
3
weather

Summary Table:

```
frequency of count vs actual_season
actual_season
Fall      1061129
Spring     471348
Summer     918589
Winter     841613
Name: count, dtype: int64

frequency of count vs actual_holiday
actual_holiday
Holiday          78435
Working day    3214244
Name: count, dtype: int64

frequency of count vs act_weather_condition
act_weather_condition
Clear                               2257952
Cloudy/Mist                          996858
Light Rain/snow/Scattered clouds      37869
Name: count, dtype: int64
```

```
frequency of count vs actual_weekday
actual_weekday
Friday       485395
Monday       444027
Saturday     487790
Sunday       477807
Thursday     473048
Tuesday      455503
Wednesday    469109
Name: count, dtype: int64

frequency of count vs actual_year
actual_year
2011    1243103
2012    2049576
Name: count, dtype: int64
```

```
frequency of count vs actual_month
actual_month
April    269094
Aug      351194
Dec      211036
Feb      151352
Jan      134933
July     344948
June     346342
March    228920
May      331686
Nov      254831
Oct      322352
Sept     345991
Name: count, dtype: int64
```

3.2.7. Feature selection

• What is Feature Selection??

– Selecting a subset of relevant features (variables, predictors) for use in model construction

– subset of a learning algorithm's input variables upon which it should focus attention, while ignoring the rest

Before creating a model, we need to assess the importance of each predictor variable in our analysis. There is a possibility that many variables in our analysis are not important at all to the problem of prediction. Hence, feature selection can help in reducing the time for computation of model as well as the complexity of the model. Also, few models require the independent variables to be free from multicollinear effect, hence it is needed to perform various procedures to ensure that the independent variables are not collinear.

Correlation analysis includes correlation matrix and correlation plot to find out significant continuous variables and we found that there is no multi-collinearity among the predictor variables.

Using ANOVA test to find out significant categorical variable.

Correlation Analysis

• Correlation tells you the association between two continuous variables

• Ranges from -1 to 1

• Measures the direction and strength of the linear relationship between two quantitative variables

• Represented by "r"

• Correlation can be calculated as

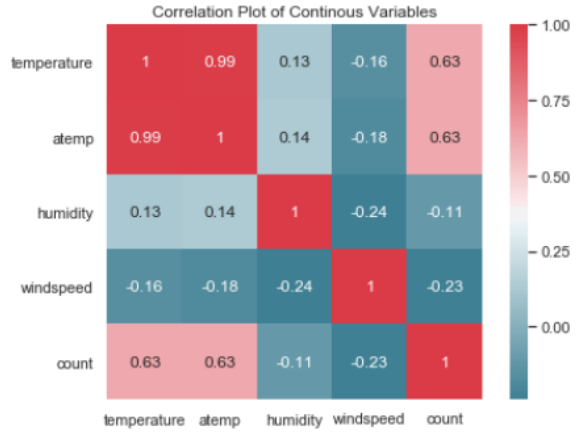
$$r = \frac{Covariance\ (x,y)}{S.D.(x)S.D.(y)}$$

- Covariance

$$Cov(X,Y) = \frac{\sum(X_i - \bar{X})*(Y_i - \bar{Y})}{n}$$

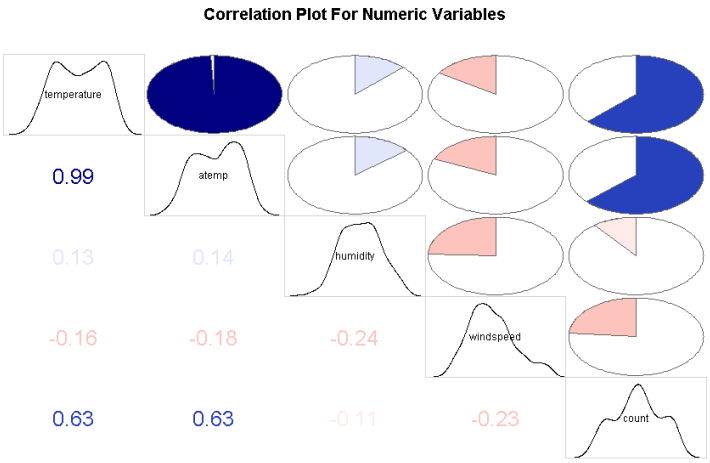
3.2.7.1. Correlation Plot and correlation matrix

In Python:



	temperature	atemp	humidity	windspeed	count
temperature	1.000000	0.991702	0.126722	-0.156916	0.627494
atemp	0.991702	1.000000	0.139924	-0.182948	0.631066
humidity	0.126722	0.139924	1.000000	-0.241160	-0.105664
windspeed	-0.156916	-0.182948	-0.241160	1.000000	-0.233657
count	0.627494	0.631066	-0.105664	-0.233657	1.000000

In R:




```
> Correlation_matrix
             temperature      atemp   humidity  windspeed      count
temperature    1.0000000  0.9917016  0.1267216 -0.1569155  0.6274940
atemp          0.9917016  1.0000000  0.1399240 -0.1829480  0.6310657
humidity       0.1267216  0.1399240  1.0000000 -0.2411599 -0.1056645
windspeed     -0.1569155 -0.1829480 -0.2411599  1.0000000 -0.2336573
count          0.6274940  0.6310657 -0.1056645 -0.2336573  1.0000000
```

From correlation plot and correlation matrix we can see ‘temperature’ variable is highly positively correlated with ‘atemp’ which means these two variable carries same information.

To avoid redundant variable in dataset and to perform model properly I am going to drop ‘atemp’ variable from the dataset.

3.2.7.2. Anova Analysis

In Python:

```
                sum_sq     df           F        PR(>F)
season    4.517974e+08    1.0  143.967653  2.133997e-30
Residual  2.287738e+09  729.0         NaN           NaN
                sum_sq     df           F        PR(>F)
year      8.798289e+08    1.0  344.890586  2.483540e-63
Residual  1.859706e+09  729.0         NaN           NaN
                sum_sq     df          F        PR(>F)
month     2.147445e+08    1.0  62.004625  1.243112e-14
Residual  2.524791e+09  729.0        NaN           NaN
                sum_sq     df         F    PR(>F)
holiday   1.279749e+07    1.0  3.421441  0.064759
Residual  2.726738e+09  729.0       NaN       NaN
                sum_sq     df         F    PR(>F)
weekday   1.246109e+07    1.0  3.331091  0.068391
Residual  2.727074e+09  729.0       NaN       NaN
                sum_sq     df         F    PR(>F)
workingday  1.024604e+07    1.0  2.736742  0.098495
Residual    2.729289e+09  729.0       NaN       NaN
                sum_sq     df          F        PR(>F)
weather   2.422888e+08    1.0  70.729298  2.150976e-16
Residual  2.497247e+09  729.0        NaN           NaN
```

In R:

```
[1] "season"
                Df     Sum Sq   Mean Sq F value              Pr(>F)
Bike_Rent[, i]   1  451797359 451797359     144 <0.0000000000000002 ***
Residuals      729 2287738033   3138187
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
[1] "year"
                Df     Sum Sq   Mean Sq F value              Pr(>F)
Bike_Rent[, i]   1  879828893 879828893   344.9 <0.0000000000000002 ***
Residuals      729 1859706499   2551038
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
[1] "month"
                Df     Sum Sq   Mean Sq F value              Pr(>F)
Bike_Rent[, i]   1  214744463 214744463   62.01 0.0000000000000124 ***
Residuals      729 2524790929   3463362
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
[1] "holiday"
                Df     Sum Sq  Mean Sq F value Pr(>F)
Bike_Rent[, i]   1   12797494 12797494   3.421 0.0648 .
Residuals      729 2726737898  3740381
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

[1] "weekday"
                Df     Sum Sq  Mean Sq F value Pr(>F)
Bike_Rent[, i]   1   12461089 12461089   3.331 0.0684 .
Residuals      729 2727074303  3740843
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
[1] "workingday"
                Df     Sum Sq  Mean Sq F value Pr(>F)
Bike_Rent[, i]   1   10246038 10246038   2.737 0.0985 .
Residuals      729 2729289354  3743881
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
[1] "weather"
                Df     Sum Sq   Mean Sq F value              Pr(>F)
Bike_Rent[, i]   1  242288753 242288753   70.73 <0.0000000000000002 ***
Residuals      729 2497246639   3425578
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

From the anova result, we can observe working day, weekday and holiday has p value > 0.05, so delete this variable not consider in model.

After Correlation and ANOVA Analysis we have remaining variables are

In Python:

```
# Lets check column names after dimension reduction
Bike_Rent.columns
```

```
Index(['season', 'year', 'month', 'weather', 'temperature', 'humidity',
       'windspeed', 'count'],
      dtype='object')
```

```
Bike_Rent.head()
```

	season	year	month	weather	temperature	humidity	windspeed	count
0	1	0	1	2	0.344167	0.805833	0.160446	985.0
1	1	0	1	2	0.363478	0.696087	0.248539	801.0
2	1	0	1	1	0.196364	0.437273	0.248309	1349.0
3	1	0	1	1	0.200000	0.590435	0.160296	1562.0
4	1	0	1	1	0.226957	0.436957	0.186900	1600.0

In R:

```
> names(Bike_Rent)
[1] "season"      "year"        "month"       "weather"     "temperature"
[6] "humidity"    "windspeed"   "count"
>
```

```
> head(Bike_Rent)
  season year month weather temperature humidity windspeed count
1      1    0     1       2    0.344167 0.805833 0.1604460   985
2      1    0     1       2    0.363478 0.696087 0.2485390   801
3      1    0     1       1    0.196364 0.437273 0.2483090  1349
4      1    0     1       1    0.200000 0.590435 0.1602960  1562
5      1    0     1       1    0.226957 0.436957 0.1869000  1600
6      1    0     1       1    0.204348 0.518261 0.0895652  1606
```

3.2.7.3. Feature Scaling

Feature scaling is a method used to normalize the range of independent variables or features of data. In data processing, it is also known as data normalization and is generally performed during the data preprocessing step. Since the range of values of raw data varies widely, in some machine learning algorithms, objective functions will not work properly without normalization. Widely used feature scaling methods are min max scaling and Standardization.

Our data may be at different levels or may contain different units. It will not be suitable to move ahead and use this data without solving this problem. This can be done by standardizing the data.

- Calculate the mean absolute deviation:
- Calculate the standardized measurement (z-score)
- Using mean absolute deviation is more robust than using standard deviation. Since the deviations are not squared the effect of outliers is somewhat reduced but their z-scores do not become too small; therefore, the outliers remain detectable.

The idea behind StandardScaler is that it will transform the data such that its distribution will have a mean value 0 and standard deviation of 1.

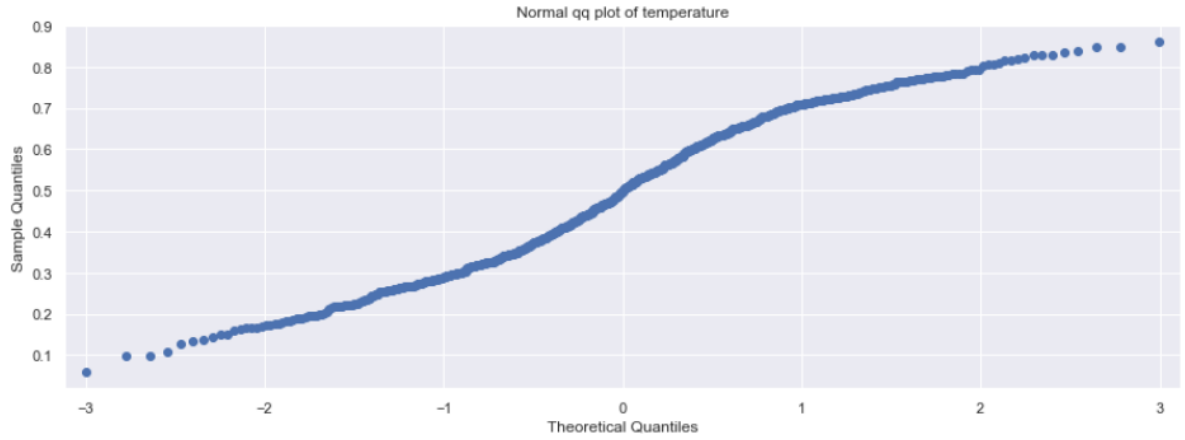
In the given dataset, for continuous variables data is already Normalized, so we need not to scale the data. We can check normality of data using qqplot and histogram plot.

In Python:

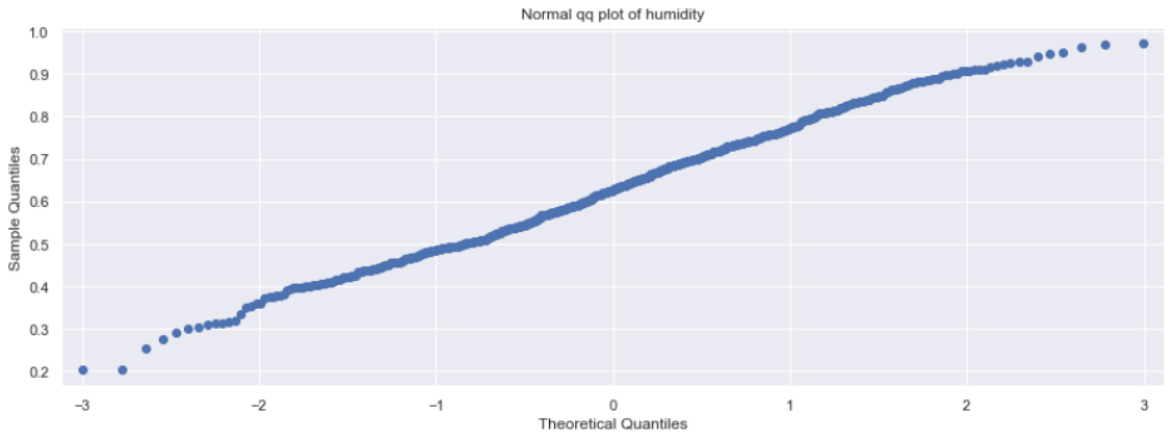
Let's check the Normality of numeric variables

Qqplot

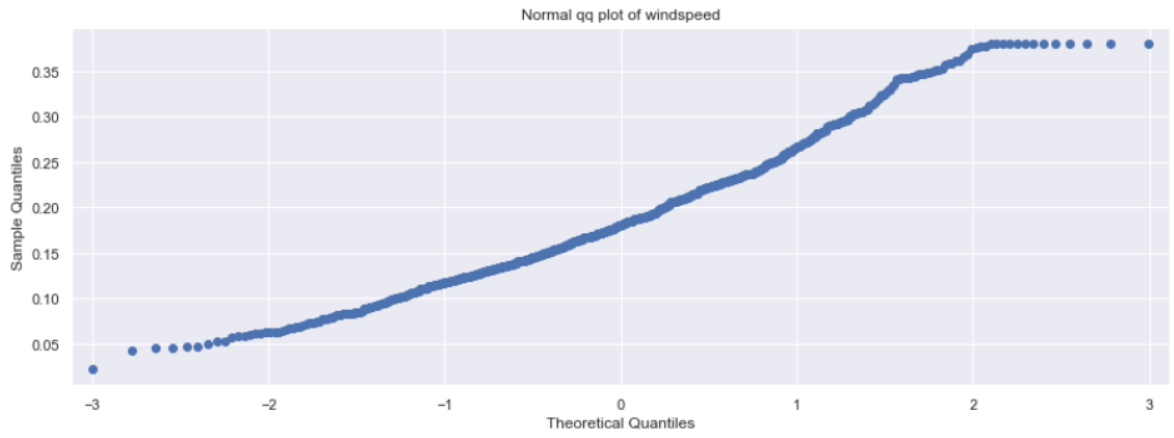
`temperature`



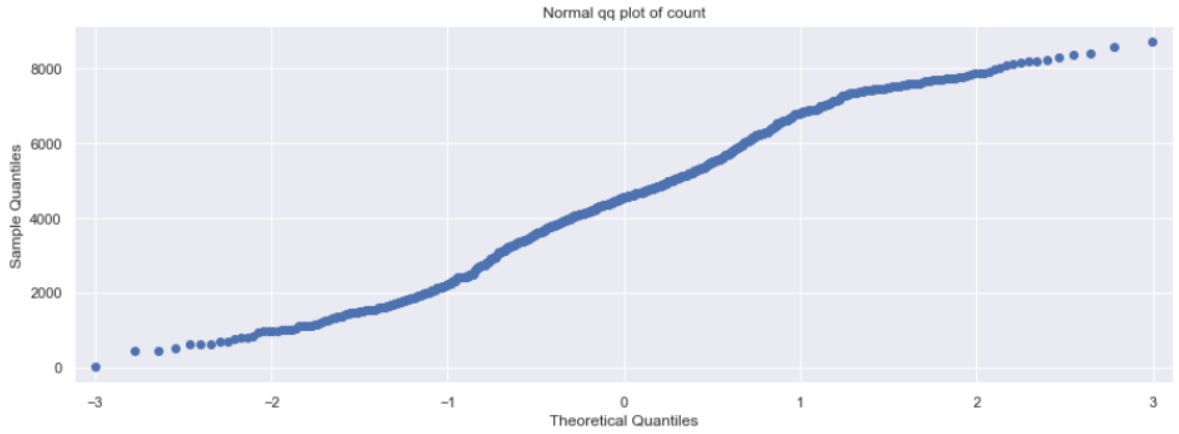
`humidity`



`windspeed`

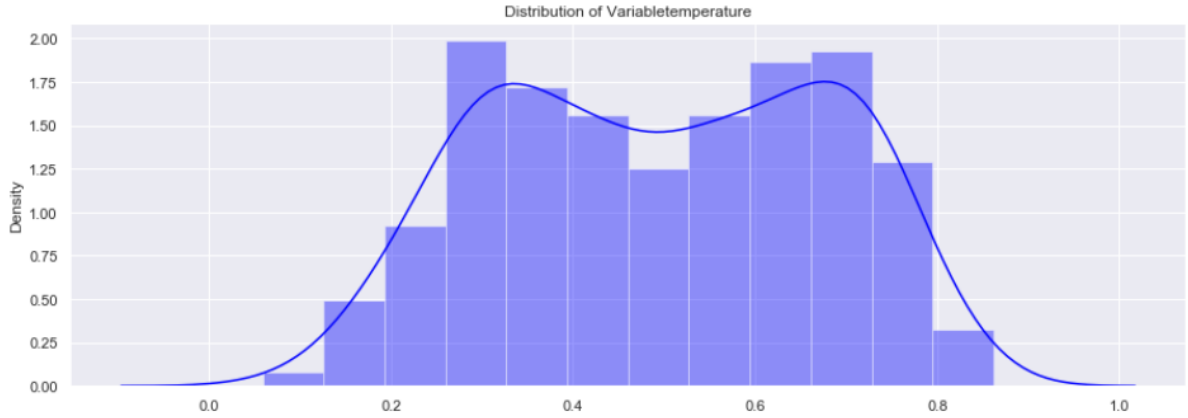


```
count
```

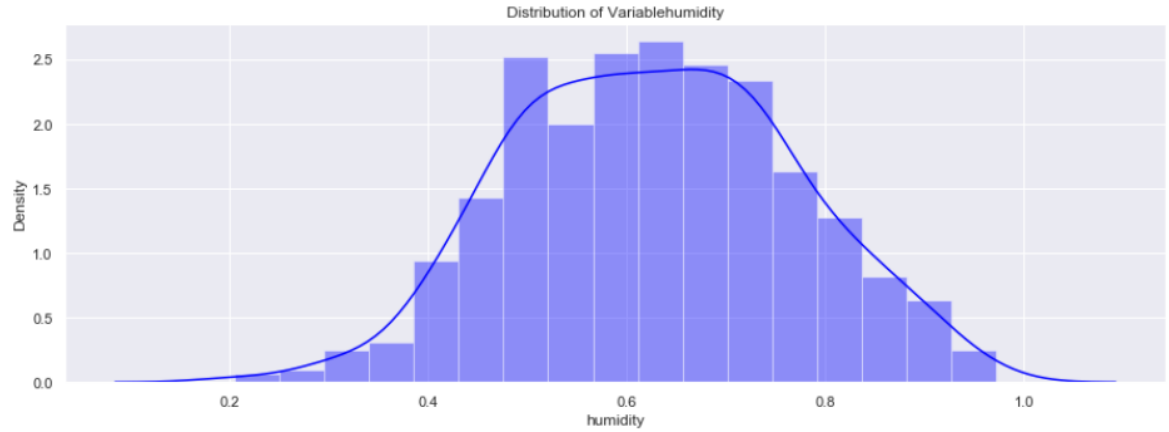


Histogram Plot:

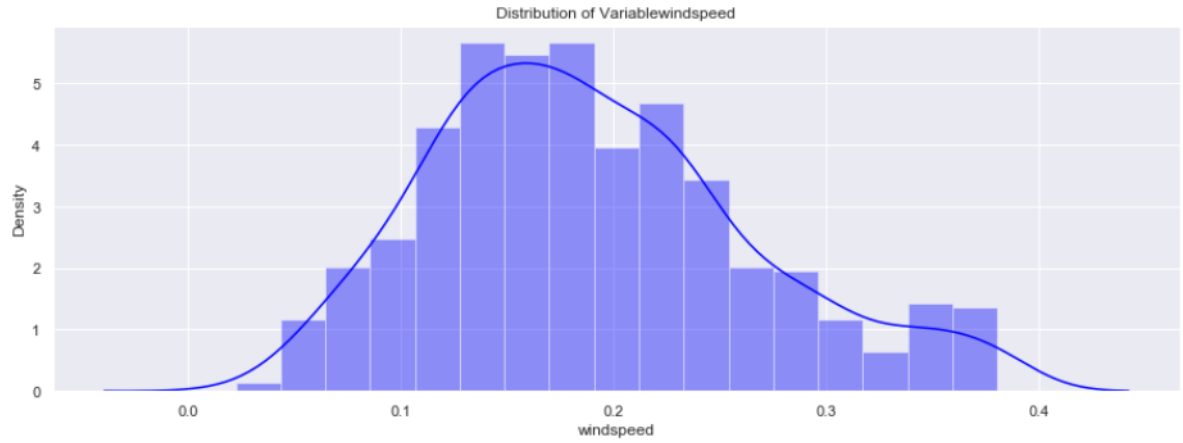
```
temperature
```



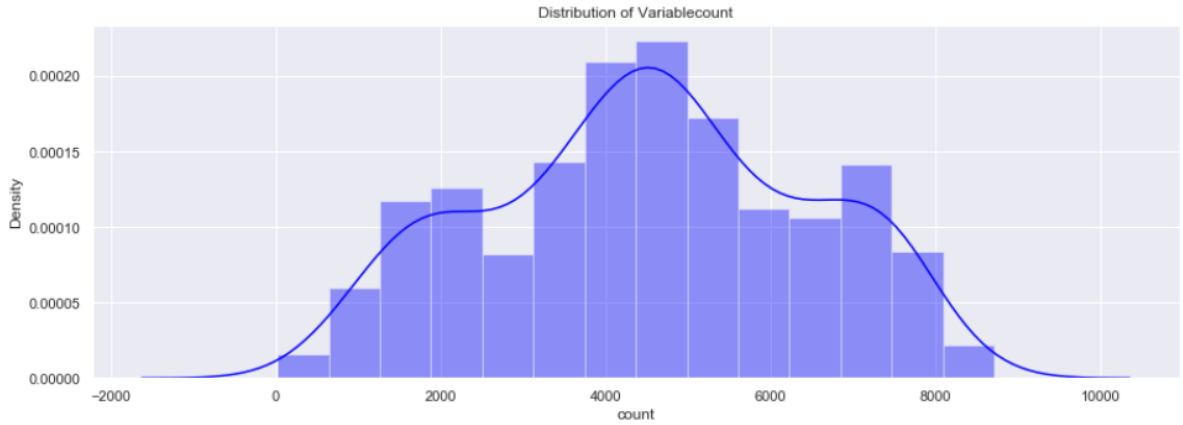

```
humidity
```



windspeed

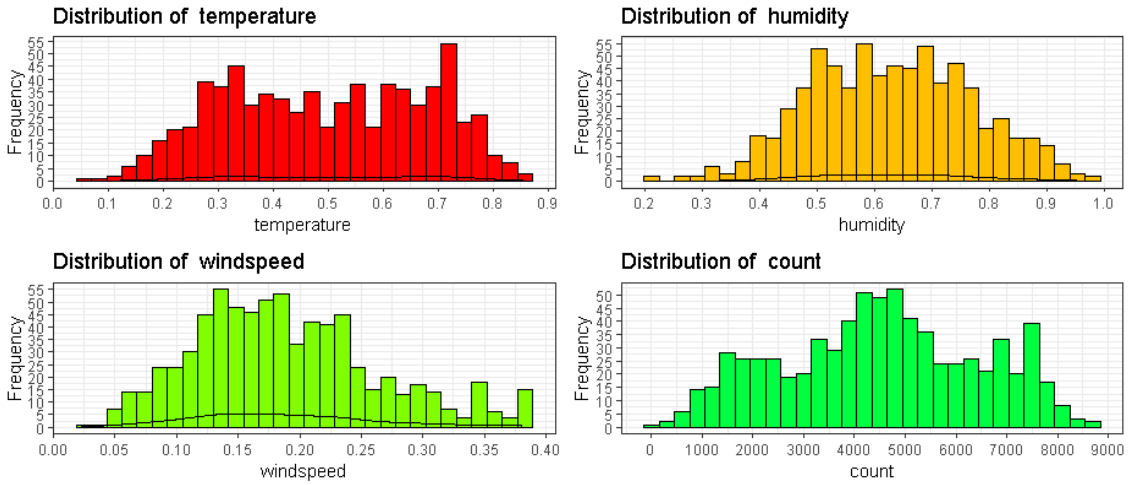


count



In R:

Histogram Plot



4. Predictive modelling

Predictive modelling is a commonly used statistical technique to predict future behavior. Predictive modelling solutions are a form of data-mining technology that works by analyzing historical and current data and generating a model to help predict future outcomes. It's a technique that uses mathematical and computational methods to predict an event or outcome. A mathematical approach uses an equation-based model that describes the phenomenon under consideration. The model is used to forecast an outcome at some future state or time based upon changes to the model inputs. The model parameters help explain how model inputs influence the outcome.

Predictive modelling is a process that uses data mining and probability to forecast outcomes. Each model is made up of a number of predictors, which are variables that are likely to influence future results. Once data has been collected for relevant predictors, a statistical model is formulated. The model may employ a simple linear equation, or it may be a complex neural network, mapped out by sophisticated software. As additional data becomes available, the statistical analysis model is validated or revised.

4.1. Model Selection

It is the task of selecting a statistical model from a set of candidate models, given data. In the simplest cases, a pre-existing set of data is considered. However, the task can also involve the design of experiments such that the data collected is well-suited to the problem of model selection.

Once completing data cleaned next process is model selection it is based on problem statement. In bike rental count prediction problem statement understood that it comes under supervised machine learning because it has both input and output variables and its regression problem as out target variable is count which is of numeric / continuous type. So, we can consider linear regression, Decision Tree, Random Forest etc.

In our project used linear regression, Decision Tree, Random Forest and Gradient Boosting models. Error matrix chosen for the given problem statement is Root Mean Squared Error (RMSE) and R2(R-Squared). Before building an any model we divided the preprocessed Bike_Rent data set in to train and test set. Data was divided into 80:20 ratio, 80% of data was used as 'train' set and rest of the 20% was used as 'test' set. The training set is used to fit the model and the test set is used to estimate the model prediction accuracy.

4.2. Hyper Parameter Tuning

In statistics, hyperparameter is a parameter from a prior distribution; it captures the prior belief before data is observed. In any machine learning algorithm, these parameters need to be initialized before training a model. Choosing appropriate hyperparameters plays a crucial role in the success of good model. Since it makes a huge impact on the learned model. For example, if the learning rate is too low, the model will miss the important patterns in the data. If it is high, it may have collisions.

We used two techniques of Hyperparameter for our models:

- Random Search
- Grid Search

Random Search : Random search is a technique where random combinations of the hyperparameters are used to find the best solution for the built model. In this search pattern, random combinations of parameters are considered in every iteration. The chances of finding the optimal parameter are

comparatively higher in random search because of the random search pattern where the model might end up being trained on the optimised parameters without any aliasing.

Grid Search : Grid search is a technique which tends to find the right set of hyperparameters for the particular model. Hyperparameters are not the model parameters and it is not possible to find the best set from the training data. Model parameters are learned during training when we optimize a loss function using something like a gradient descent. In this tuning technique, we simply build a model for every combination of various hyperparameters and evaluate each model. The model which gives the highest accuracy wins. The pattern followed here is similar to the grid, where all the values are placed in the form of a matrix. Each set of parameters is taken into consideration and the accuracy is noted. Once all the combinations are evaluated, the model with the set of parameters which give the top accuracy is considered to be the best. We used these two techniques for Decision tree, Random Forest, Gradient boosting models.

4.3.Multiple Linear Regression

Multiple linear regression (MLR), also known simply as multiple regression, is a statistical technique that uses several explanatory variables to predict the outcome of a response variable. The goal of multiple linear regression (MLR) is to model the linear relationship between the explanatory (independent) variables and response (dependent) variable.

In essence, multiple regression is the extension of ordinary least-squares (OLS) regression that involves more than one explanatory variable.

The Formula for Multiple Linear Regression Is

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_p x_{ip} + \epsilon$$

where, for $i = n$ observations:

y_i = dependent variable

x_i = expanatory variables

β_0 = y-intercept (constant term)

β_p = slope coefficients for each explanatory variable

ϵ = the model's error term (also known as the residuals)

Explaining Multiple Linear Regression

A simple linear regression is a function that allows an analyst or statistician to make predictions about one variable based on the information that is known about another variable. Linear regression can only be used when one has two continuous variables— an independent variable and a dependent variable. The independent variable is the parameter that is used to calculate the dependent variable or outcome. A multiple regression model extends to several explanatory variables.

The multiple regression model is based on the following assumptions:

There is a linear relationship between the dependent variables and the independent variables.

The independent variables are not too highly correlated with each other.

Yi observations are selected independently and randomly from the population.

Residuals should be normally distributed with a mean of 0 and variance σ.

The coefficient of determination (R-squared) is a statistical metric that is used to measure how much of the variation in outcome can be explained by the variation in the independent variables. R2 always increases as more predictors are added to the MLR model even though the predictors may not be related to the outcome variable. R2 by itself can't thus be used to identify which predictors should be included in a model and which should be excluded. R2 can only be between 0 and 1, where 0 indicates that the outcome cannot be predicted by any of the independent variables and 1 indicates that the outcome can be predicted without error from the independent variables.

Let's check the assumptions of linear regression:

a) Error should follow normal distribution

b) Error should follow Homoscadacity or No Heteroscadacity

c) No multicollinearity among the independent variables

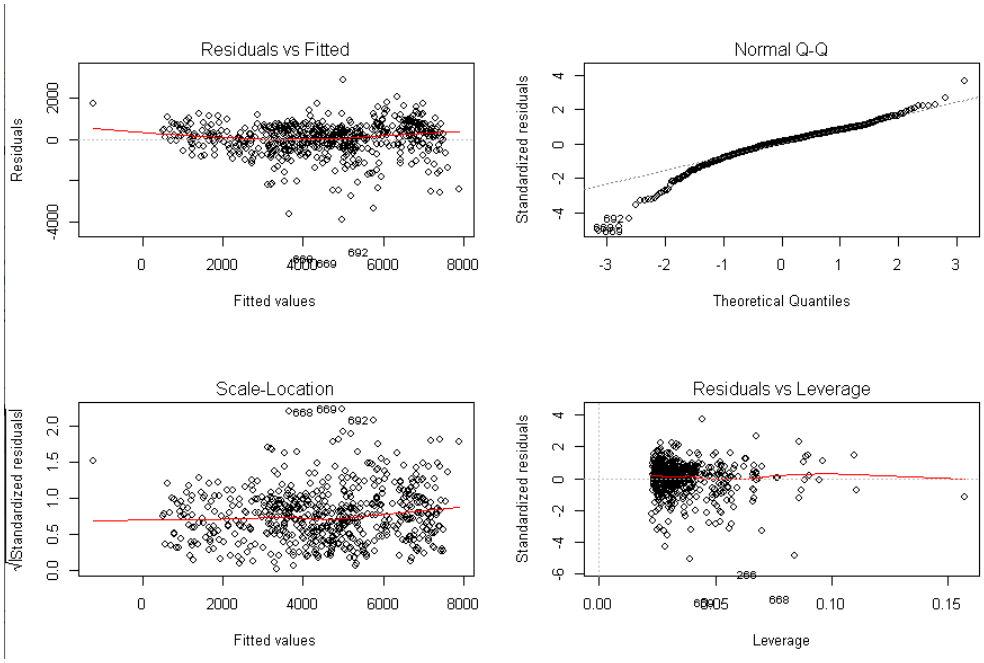
d) No serial / autocorrelation in error

a) Error should follow normal distribution:

To check this assumption, we plot normal QQ plot. The normal QQ plot helps us to determine if our target variable is normally distributed by plotting quantiles (i.e. percentiles) from our distribution against a theoretical distribution. If our data is normally distributed, it will be plotted in a generally straight line on the QQ plot. We can also plot histogram where error should show a curvy bell shape for our project this assumption satisfied.

b) Error should follow Homoscadacity or No Heteroscadacity:

To check this assumption, we can use residual plot Residual plots help us evaluate and improve our regression model. A residual is the difference between the observed value of the dependent variable (y) and the predicted value (ŷ). A "good" residual vs. fitted plot should be relatively shapeless without clear patterns in the data, no obvious outliers, and be generally symmetrically distributed. For our Project this assumption is violated (residuals following a pattern not scattered) we can see it in Residuals vs fitted plot.



c) No multicollinearity among the independent variables:

To check this assumption, we are going to use Variance Inflation Factor (VIF) Variance inflation factor is a measure of the amount of multicollinearity in a set of multiple regression variables The Variance Inflation Factor (VIF) is 1/Tolerance, it is always greater than or equal to 1. There is no formal VIF value for determining presence of multicollinearity. Values of VIF that exceeds 10 are often regarded as indicating multicollinearity, but in weaker models values above 2.5 may be a cause for concern for our project VIF values are within the range and this assumption is satisfied.

```
     Variables        VIF
0  temperature   1.034137
1     humidity   1.070959
2    windspeed   1.080362
3    Intercept  41.553921
```

None of the variables from the 3 input variables has collinearity problem.

e) No serial / autocorrelation in error:

Errors of all observation independent each other we can check this assumption using dwt test or Durbin Watson test The Durbin Watson (DW) statistic is a test for autocorrelation in the residuals from a statistical regression analysis. The Durbin - Watson statistic will always have a value between 0 and 4. A value of 2.0 means that there is no autocorrelation detected in the sample. Values from 0 to less than 2 indicate positive autocorrelation and values from 2 to 4 indicate negative autocorrelation. So, our model is fine with this assumption.

```
        Durbin-Watson test

data:  LR_Model
DW = 1.2978, p-value < 0.00000000000000022
alternative hypothesis: true autocorrelation is greater than 0
```

Using linear regression for our project in python the accuracy is 79.15 % and in R the model the accuracy is 80.51%. Let's check other Algorithms.

In Python:

```
Mean Absolute Precentage Error for test data=20.841930379420212
R^2_score for test data=0.7819292243796828
RMSE for test data=846.3506302699261
Accuracy :=79.1580696205798
```

	Model Name	Accuracy	MAPE_Test	R-squared_Test	RMSE_test
0	Linear Regression	79.15807	20.84193	0.781929	846.35063

In R:

Model	Test_Accuracy	MAPE_Test	Rsquare_Test	Rmse_Test
Linear Regression	80.5134373	19.4865627	0.83090035	808.3312216

4.4. Decision Tree

Decision Tree is a supervised machine learning algorithm, which is used to predict the data for classification and regression. It accepts both continuous and categorical variables. A decision tree is a decision support tool that uses a tree like graph or model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. Each branch connects nodes with “and” and multiple branches are connected by “or”. Extremely easy to understand by the business users. It provides the output in the form of rule, which can easily understand by a non - technical person also. Output of Decision tree regression model is as below

In Python:

```
Mean Absolute Precentage Error for test data=18.61043341675327
R^2_score for test data=0.7357324122047093
RMSE for test data=931.6945135969551
Accuracy :=81.38956658324673
```

	Model Name	Accuracy	MAPE_Test	R-squared_Test	RMSE_test
0	Desicision Tree	81.389567	18.610433	0.735732	931.694514

A) Random Search CV in Decision Tree:

```
Best Parameter={'max_depth': 11}
Best Model=DecisionTreeRegressor(ccp_alpha=0.0, criterion='mse', max_depth=11,
                      max_features=None, max_leaf_nodes=None,
                      min_impurity_decrease=0.0, min_impurity_split=None,
                      min_samples_leaf=1, min_samples_split=2,
                      min_weight_fraction_leaf=0.0, presort='deprecated',
                      random_state=0, splitter='best')
Mean Absolute Precentage Error for test data=16.74813754859453
R^2_score for test data=0.8213647699359325
RMSE for test data=766.0112442164559
Accuracy :=83.25186245140547
```

	Model Name	Accuracy	MAPE_Test	R-squared_Test	RMSE_test
0	Random Search CV In Decision Tree	83.251862	16.748138	0.821365	766.011244

B) Grid Search CV in Decision Tree:

```
Best Parameter={'max_depth': 5}
Best Model=DecisionTreeRegressor(ccp_alpha=0.0, criterion='mse', max_depth=5,
                      max_features=None, max_leaf_nodes=None,
                      min_impurity_decrease=0.0, min_impurity_split=None,
                      min_samples_leaf=1, min_samples_split=2,
                      min_weight_fraction_leaf=0.0, presort='deprecated',
                      random_state=0, splitter='best')
Mean Absolute Precentage Error for test data=18.005792343718657
R^2_score for test data=0.8258697500492262
RMSE for test data=756.290594763957
Accuracy :=81.99420765628135
```

	Model Name	Accuracy	MAPE_Test	R-squared_Test	RMSE_test
0	Decision Tree Grid Search CV	81.994208	18.005792	0.82587	756.290595

In R:

Model	Test_Accuracy	MAPE_Test	Rsquare_Test	Rmse_Test
Decision Tree for Regression	76.21868822	23.78131178	0.775647954	929.5915896

A) Random Search CV in Decision Tree:

```
RMSE was used to select the optimal model using the smallest value.
The final value used for the model was maxdepth = 10.
```

Model	Test_Accuracy	MAPE_Test	Rsquare_Test	Rmse_Test
Random Search in Decision Tree	76.21868822	23.78131178	0.775647954	929.5915896

B) Grid Search CV in Decision Tree:

```
RMSE was used to select the optimal model using the smallest value.
The final value used for the model was maxdepth = 9.
```

Model	Test_Accuracy	MAPE_Test	Rsquare_Test	Rmse_Test
Gird Search in Decision Tree	76.21868822	23.78131178	0.775647954	929.5915896

4.5. Random Forest

Random Forest is an ensemble technique that consists of many decision trees. The idea behind Random Forest is to build N number of trees to have more accuracy in dataset. It is called random forest as we are building n no. of trees randomly. In other words, to build the decision trees it selects randomly N no of variables and n no of observations. It means to build each decision tree on random forest we are not going to use the same data. The higher no of trees in the random forest will give higher no of accuracy, so in random forest we can go for multiple trees. It can handle large no of independent variables without variable deletion and it will give the estimates that what variables are important. The Number of trees used in R and Python for random forest model are 200 no's

Using Random Forest model

In Python:

```
Mean Absolute Precentage Error for test data=13.072015930173622
R^2_score for test data=0.912575964213138
RMSE for test data=535.8793394705832
Accuracy :=86.92798406982638
```

	Model Name	Accuracy	MAPE_Test	R-squared_Test	RMSE_test
0	Random Forest	86.927984	13.072016	0.912576	535.879339

In Python:

A) Random Search CV in Random Forest

```
Best Parameter={'n_estimators': 73, 'max_depth': 13}
Best Model=RandomForestRegressor(bootstrap=True, ccp_alpha=0.0, criterion='mse',
                      max_depth=13, max_features='auto', max_leaf_nodes=None,
                      max_samples=None, min_impurity_decrease=0.0,
                      min_impurity_split=None, min_samples_leaf=1,
                      min_samples_split=2, min_weight_fraction_leaf=0.0,
                      n_estimators=73, n_jobs=None, oob_score=False,
                      random_state=0, verbose=0, warm_start=False)
Mean Absolute Precentage Error for test data=13.190494050417833
R^2_score for test data=0.9137450792434485
RMSE for test data=532.2841424661989
Accuracy :=86.80950594958216
```

	Model Name	Accuracy	MAPE_Test	R-squared_Test	RMSE_test
0	Random Forest Random Search CV	86.809506	13.190494	0.913745	532.284142

B) Grid Search CV in Random Forest

```
Best Parameter={'n_estimators': 73, 'max_depth': 13}
Best Model=RandomForestRegressor(bootstrap=True, ccp_alpha=0.0, criterion='mse',
                      max_depth=13, max_features='auto', max_leaf_nodes=None,
                      max_samples=None, min_impurity_decrease=0.0,
                      min_impurity_split=None, min_samples_leaf=1,
                      min_samples_split=2, min_weight_fraction_leaf=0.0,
                      n_estimators=19, n_jobs=None, oob_score=False,
                      random_state=0, verbose=0, warm_start=False)
Mean Absolute Precentage Error for test data=13.878527400149524
R^2_score for test data=0.9014935658909338
RMSE for test data=568.8318257385259
Accuracy :=86.12147259985048
```

	Model Name	Accuracy	MAPE_Test	R-squared_Test	RMSE_test
0	Random Forest Grid Search CV	86.121473	13.878527	0.901494	568.831826

In R:

Model	Test_Accuracy	MAPE_Test	Rsquare_Test	Rmse_Test
Random Forest	82.36845278	17.63154722	0.876851593	692.5009965

A) Random Search CV in Random Forest

```
RMSE was used to select the optimal model using the smallest value.
The final value used for the model was mtry = 8.
```

Model	Test_Accuracy	MAPE_Test	Rsquare_Test	Rmse_Test
Random Search in Random Forest	82.42978925	17.57021075	0.869208927	706.6425686

B) Grid Search CV in Random Forest

```
RMSE was used to select the optimal model using the smallest value.
The final value used for the model was mtry = 10.
```

Model	Test_Accuracy	MAPE_Test	Rsquare_Test	Rmse_Test
Grid Search in Random Forest	82.62419513	17.37580487	0.871111485	702.695805

4.6.Gradient Boosting

Gradient boosting is a machine learning technique for regression and classification problems, It produces a prediction model in the form of an ensemble of weak learner models and produce a strong learner with less misclassification and higher accuracy. It feed the error from one decision tree to another decision tree and generates a strong classifier or Regressor

In Python:

```
Mean Absolute Precentage Error for test data=12.06622739477786
R^2_score for test data=0.9118654090521587
RMSE for test data=538.0526620396487
Accuracy :=87.93377260522215
```

	Model Name	Accuracy	MAPE_Test	R-squared_Test	RMSE_test
0	Gradient Boosting	87.933773	12.066227	0.911865	538.052662

A) Random Search CV in Gradient Boosting

```
Best Parameter={'n_estimators': 23, 'max_depth': 7}
Best Model=GradientBoostingRegressor(alpha=0.9, ccp_alpha=0.0, criterion='friedman_mse',
                          init=None, learning_rate=0.1, loss='ls', max_depth=7,
                          max_features=None, max_leaf_nodes=None,
                          min_impurity_decrease=0.0, min_impurity_split=None,
                          min_samples_leaf=1, min_samples_split=2,
                          min_weight_fraction_leaf=0.0, n_estimators=23,
                          n_iter_no_change=None, presort='deprecated',
                          random_state=0, subsample=1.0, tol=0.0001,
                          validation_fraction=0.1, verbose=0, warm_start=False)
Mean Absolute Precentage Error for test data=14.112654958720256
R^2_score for test data=0.9022301320272976
RMSE for test data=846.3506302699261
Accuracy :=85.88734504127974
```

	Model Name	Accuracy	MAPE_Test	R-squared_Test	RMSE_test
0	Gradient Boosting Random Search CV	85.887345	14.112655	0.90223	846.35063

B) Grid Search CV in Gradient Boosting

```
Best Parameter={'max_depth': 5, 'n_estimators': 19}
Best Model=GradientBoostingRegressor(alpha=0.9, ccp_alpha=0.0, criterion='friedman_mse',
                          init=None, learning_rate=0.1, loss='ls', max_depth=5,
                          max_features=None, max_leaf_nodes=None,
                          min_impurity_decrease=0.0, min_impurity_split=None,
                          min_samples_leaf=1, min_samples_split=2,
                          min_weight_fraction_leaf=0.0, n_estimators=19,
                          n_iter_no_change=None, presort='deprecated',
                          random_state=0, subsample=1.0, tol=0.0001,
                          validation_fraction=0.1, verbose=0, warm_start=False)
Mean Absolute Precentage Error for test data=15.833943071987505
R^2_score for test data=0.8896970392618515
RMSE for test data=601.92886875878
Accuracy :=84.1660569280125
```

	Model Name	Accuracy	MAPE_Test	R-squared_Test	RMSE_test
0	Gradient Boosting Grid Search CV	84.166057	15.833943	0.889697	601.928869

In R:

Model	Test_Accuracy	MAPE_Test	Rsquare_Test	Rmse_Test
Gradient Boosting	82.26832083	17.73167917	0.850642096	754.9006985

A) Random Search CV in Gradient Boosting

```
RMSE was used to select the optimal model using the smallest value.
The final values used for the model were n.trees = 92, interaction.depth =
 3, shrinkage = 0.0947087 and n.minobsinnode = 22.
```

Model	Test_Accuracy	MAPE_Test	Rsquare_Test	Rmse_Test
Random Search in Gradient Boosting	82.4247365	17.5752635	0.877470574	702.695805

B) Grid Search CV in Gradient Boosting

```
RMSE was used to select the optimal model using the smallest value.
The final values used for the model were n.trees = 4349, interaction.depth =
 2, shrinkage = 0.01 and n.minobsinnode = 2.
```

Model	Test_Accuracy	MAPE_Test	Rsquare_Test	Rmse_Test
Grid Search in Gradient Boosting	82.16427984	17.83572016	0.876459908	695.6098248

5. Conclusion

5.1. Accuracy, RMSE, R-squared, MAPE

I have calculated RMSE, R-Square, MAPE (Mean Absolute Percentage Error) and Accuracy for all the models

Root Mean Square Error (RMSE) is the standard deviation of the residuals (prediction errors) Residuals are a measure of how far from the regression line data points are, RMSE is a measure of how spread out these residuals are. In other words, it tells you how concentrated the data is around the line of best fit. Whereas R squared is a relative measure of fit, RMSE is an absolute measure of fit. As the square root of a variance, RMSE can be interpreted as the standard deviation of the unexplained variance and has the useful property of being in the same units as the response variable. Lower values of RMSE indicate better fit.

R- squared is basically explains the degree to which input variable explain the variation of the output. In simple words R-Squared tells how much variance of dependent variable explained by the independent variable. It is a measure if goodness of fit in regression line.

We also said about MAPE (Mean Absolute Percent Error) value for all the models which measures the size of the error in percentage terms.

Selected Model Results:

In Python:

Summary:

	Model Name	Accuracy	MAPE_Test	R-squared_Test	RMSE_test
0	Linear Regression	79.158070	20.841930	0.781929	846.350630
1	Desicision Tree	81.389567	18.610433	0.735732	931.694514
2	Random Search CV In Decision Tree	83.251862	16.748138	0.821365	766.011244
3	Decision Tree Grid Search CV	81.994208	18.005792	0.825870	756.290595
4	Random Forest	86.877675	13.122325	0.916267	524.446203
5	Random Forest Random Search CV	86.590618	13.409382	0.910967	540.788449
6	Random Forest Grid Search CV	86.121473	13.878527	0.901494	568.831826
7	Gradient Boosting	87.978539	12.021461	0.911900	537.946072
8	Gradient Boosting Random Search CV	85.883365	14.116635	0.902096	846.350630
9	Gradient Boosting Grid Search CV	84.166057	15.833943	0.889697	601.928869

From above table we can see Predicted Bike Rental count accuracy is highest using Gradient Boosting Model (87.93%) in Python

Actual count vs Predicted count in ascending order (only few predictions is reported):

Please refer appendix for all predicted test value

	Predicted_Bike_Rental_Count	Actual_Bike_Rental_Count
139	977.0	605.0
113	1999.0	623.0
118	1333.0	1349.0
25	1243.0	1360.0
114	1297.0	1406.0
...	...	...
73	7868.0	7466.0
87	7119.0	7498.0
3	7244.0	7582.0
122	7172.0	7605.0
1	7520.0	7691.0

*

Please refer appendix for all predictions

In R:

Summary:

	Model	Test_Accuracy	MAPE_Test	Rsquare_Test	Rmse_Test
1	Linear Regression	80.51344	19.48656	0.8309003	808.3312
2	Decision Tree for Regression	76.21869	23.78131	0.7756480	929.5916
3	Random Search in Decision Tree	76.21869	23.78131	0.7756480	929.5916
4	Gird Search in Decision Tree	76.21869	23.78131	0.7756480	929.5916
5	Random Forest	82.36845	17.63155	0.8768516	692.5010
6	Random Search in Random Forest	82.42979	17.57021	0.8692089	706.6426
7	Grid Search in Random Forest	82.62420	17.37580	0.8711115	702.6958
8	Gradient Boosting	82.26832	17.73168	0.8506421	754.9007
9	Random Search in Gradient Boosting	82.42474	17.57526	0.8774706	702.6958
10	Grid Search in Gradient Boosting	82.16428	17.83572	0.8764599	695.6098

From above table we can see Predicted Bike Rental count accuracy is highest using Grid search in Random Forest Model (82.62%) in R

Actual count vs Predicted count in ascending order (only few predictions is reported):

Please refer appendix for all predicted test value

Predicted_Bike_Rental_Count	Actual_Bike_Rental_Count
1786	605
2380	705
1239	822
3694	1027
1129	1167
1116	1263
1271	1321
2302	1341
1559	1360
1235	1406
2052	1446
1545	1526
1646	1550
1382	1600
1499	1605
2890	1607
1958	1623
2663	1749
2644	1796
1547	1807
1848	1812
5733	6883
7115	6891
6566	6904
7089	6998
7454	7006
7001	7105
6810	7132
7085	7148
6808	7216
6625	7282
6543	7350
6955	7384
6240	7444
7112	7499
7156	7534
7577	7641
7448	7702
7057	7965
7754	8009
7641	8555

6. Brief Insights about the Bike Rental Count prediction Project

From this bike rental count project, we analyzed and found below observations.

1) Temperature variable has major impact on bike rental count

2) Bike rent count is highest in month of august and low in Jan

3) Bike rent count is high on weather clear: i.e., when the weather is Clear, Few clouds, Partly cloudy, Partly cloudy

4) Bike rent count is high in season fall and low in springer.

5) Windspeed and humidity don't have any impact on bike rental count and these two Variables are negatively correlated with all variables

6) Maximum bikes rented on saturday and least bikes on Monday

7) Bike rent count is high on working day

8) Bike rent count is high in year 2012

7. References

1. https://www.analyticsvidhya.com/blog/2017/09/machine-learning-models-as-apis-using-flask/

2. For Model deployment refer https://www.saedsayad.com/model_deployment.htm

3. https://www.analyticsvidhya.com/blog/2016/01/guide-data-exploration/

4. For Model evaluation refer https://www.saedsayad.com/model_evaluation.htm

5.https://www.youtube.com/watch?v=mrExsjcvF4o&list=PLZoTAELRMXVOAvUbePX1lTdxQR8EY35Z1&ind ex=2 # Model Deployment Reference

6. https://trainingdatascience.com/workshops/histograms-and-skewed-data/

7. Reference Blog: https://machinelearningmastery.com/tune-machine-learning-algorithms-in-r/

8. For parameter Tuning https://uc-r.github.io/random_forestsp

9. http://www.knowru.com/blog/how-create-restful-api-for-machine-learning-credit-model-in-r/

8. Appendix


Bike Rental Problem Statement.pdf


day_input_data.csv


Bike_Rent_Count_Prediction_PythonCod


Bike_Rent_Count_R Code.R


Bike_Rental_Count_RF_results_R.csv


Bike_Rental_Count_summary_R.csv


Bike_Rental_Count_GB_results_py.csv


Bike_Rental_Count_Model_Summary_py.