# Bike Sharing Demand Prediction

## Technical Document

**Mohammad Jibran**
**Siddhi Thakur**
| Data Science Trainees |

**ALMABETTER**

# Abstract

This project is about to give prediction of number of bikes that will be rented in a particular hour of a day, after analysing with several weathers condition.

In order to comprehend the data in this project, we will apply various supervised Machine learning libraries and algorithms for better accuracy of Model.

## 1. Problem Statement:

Currently for the purpose of improving transportation facility, rented bike have been introduced in several urban and metropolitan cities. It is necessary in order to make rented bike accessible and availability to local public of that city at the appropriate time. Eventually, maintain the steady supply of rented bikes in the cities emerges as a top priority. Predicting the number of bikes needed to maintain a steady supply of rental bikes at each hour's interval is essential.

## 2. Introduction:

Seoul, one of the most populous cities in the world. With more than 10 million residents and a building density of 33,000 people per square kilometres, Seoul has the perfect mix of urban sprawl and cozy neighborhoods. It's also home to one of the largest concentrations of bicycles in all the world—in other words, there's usually an empty bike for you when you need it. The city has been relying on a bike sharing system for years now to provide transportation services to its citizens.

The transportation is essential part of day to day activities, in urban cities most of the work depends on transportation. For covering short distance bike is much preferable than other vehicles because it became one of the most environment friendly and pocket friendly businesses that attract several tourism and local public of the cities. Who won't love the beauty of nature with the sense of helping world's pollution rate.

In many urban cities we can see bike station are providing bike facility for short distance. Riding bike every day also improves cholesterol level.

The dataset contain numerical, categorical type of data, here we have two years of record of bike rented 2017 and 2018,and some other information like functioning day, humidity, temperature level, our project is we deal with various weather related fact to make accurate prediction without fail.

## 3. Understanding the data

In the datasets, there are 8760 entries and 14 columns. The bike rented are mostly found between December 1, 2017 and November 30 2018, with the successfully receiving bike rent demand.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8760 entries, 0 to 8759
Data columns (total 14 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   Date                       8760 non-null   object
 1   Rented Bike Count          8760 non-null   int64
 2   Hour                       8760 non-null   int64
 3   Temperature(°C)            8760 non-null   float64
 4   Humidity(%)                8760 non-null   int64
 5   Wind speed (m/s)           8760 non-null   float64
 6   Visibility (10m)           8760 non-null   int64
 7   Dew point temperature(°C)  8760 non-null   float64
 8   Solar Radiation (MJ/m2)    8760 non-null   float64
 9   Rainfall(mm)               8760 non-null   float64
 10  Snowfall (cm)              8760 non-null   float64
 11  Seasons                    8760 non-null   object
 12  Holiday                    8760 non-null   object
 13  Functioning Day            8760 non-null   object
dtypes: float64(6), int64(4), object(4)
memory usage: 958.2+ KB
```

In this dataset we have different types of columns which help to make good prediction, but before actual work begin we have to observe dataset and its data types.
Convert date column from object type to date-time type and the extract month, weekdays, year from date column.

```
df['Date']=pd.to_datetime(df['Date'])
```

```
# Split day of week, month and year in three column

df['weekday'] = df['Date'].dt.day_name()
df['month'] = df['Date'].dt.month_name()
df['year'] = df['Date'].map(lambda x: x.year).astype("object")
```

```
# Drop the Date column as we extract necessary details

df.drop(columns=['Date'],inplace=True)
```

After extracting necessary data we drop ‘date’ column from dataset.

These are updated columns ----

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8760 entries, 0 to 8759
Data columns (total 16 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   Rented Bike Count          8760 non-null   int64
 1   Hour                       8760 non-null   int64
 2   Temperature(°C)            8760 non-null   float64
 3   Humidity(%)                8760 non-null   int64
 4   Wind speed (m/s)           8760 non-null   float64
 5   Visibility (10m)           8760 non-null   int64
 6   Dew point temperature(°C)  8760 non-null   float64
 7   Solar Radiation (MJ/m2)    8760 non-null   float64
 8   Rainfall(mm)               8760 non-null   float64
 9   Snowfall (cm)              8760 non-null   float64
 10  Seasons                    8760 non-null   object
 11  Holiday                    8760 non-null   object
 12  Functioning Day            8760 non-null   object
 13  weekday                    8760 non-null   object
 14  month                      8760 non-null   object
 15  year                       8760 non-null   object
dtypes: float64(6), int64(4), object(6)
memory usage: 1.1+ MB
```

| Columns Name | Columns Description |
|---|---|
| Rented Bike Count | It give total counts of bike rented in each hour. |
| Hour | Hours in numeric 0 to 23. |
| Temperature | Person see temperature before renting the bike and temperature is in between -5.2 to 1.9 c. |
| Humidity | Humidity lies between 37% to 43%. |
| wind_speed | Speed of wind (0.0-7.4). |
| Visibility | Visibility (0,2000) |
| Dew_point_temperature | The dew point is the temperature at which air is saturated with water vapour (-17.6,-9.3) |
| Solar Radiation | It’s an electromagnetic radiation emitted by the sun. |
| Rainfall | Rainfall rate in millimetre. |

| | |
|---|---|
| snowfall | Snowfall rate in cm. |
| Seasons | four season type<br>• Summer<br>• Autumns<br>• Winter<br>• Spring |
| Holiday | Two type of holiday<br>1. Holiday<br>2. No Holiday. |
| Functional_Day | It segregate data into two<br>• Functional_Day (Yes)<br>• Non_Functional_Day (No) |
| Weekday | It has days from Monday to Sunday. |
| Month | It has months from January to December. |
| Year | It has records since 2017 to 2018. |

## 4. Data Wrangling

Data wrangling is a term often used to describe during data analysis. It helps to transform data from one format into another. The aim is to make data more accessible. These include things like data collection, exploratory analysis, data cleansing, creating data structures, and storage.

# Steps involved

- **Importing important libraries**

Our major goal in this step was to import all of the necessary libraries to aid us in exploring the issue statement and doing EDA to draw conclusions based on the data collection.

# Libraries we used:

## 1. NumPy

NumPy is the python library used for programming languages, adding support for large, multidimensional arrays and matrices with large collections.

## 2. Pandas

It is a software library written for python programming, flexible, and expressive data structures designed to make working with relational or labelled data both easy and intuitive. Pandas allow us to access many of Matplotlibs and NumPy's methods with fewer codes.

## 3. Matplotlib

It is a pyplot collection of functions that make matplotlib work like MATLAB. e.g., creates a figure, lines a plotting area.

## 4. Seaborn

It is an open-source python library built on top of matplotlib. It is used for data visualization and EDA.seaborn works easily with Data frames and pandas' libraries. The graphs can also be customized.

## 5. Scikit-learn (Sklearn)

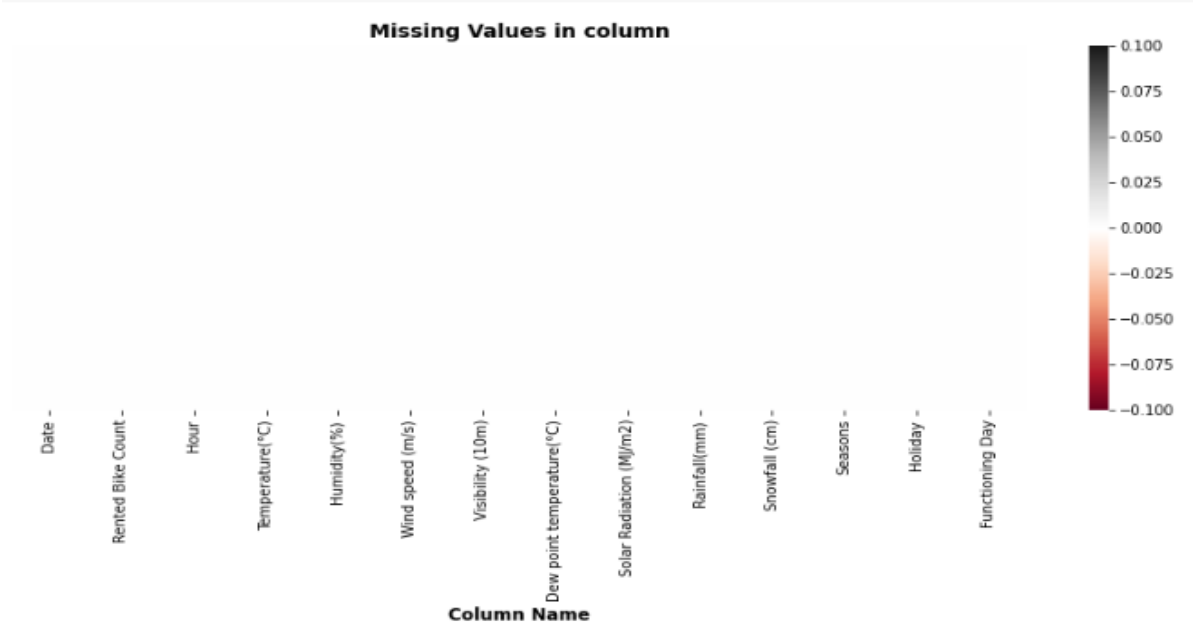
Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modelling including classification, regression, and clustering and dimensionality reduction via a consistence interface in Python.

# Outliers Treatment

Our dataset contains outliers which might tend to disturb our insights. We get rid of them for better understanding. Columns named 'Rented Bike Count' have huge positive skewedness, we remove outliers by doing square Transformation method on these columns.

# Pre-processing of dataset

Our dataset does not contain any null values and duplicated values. We plot a heat-map for better understanding. As we can see in below plot the outcome is 0.0.




# Exploratory Data Analysis

After treating the null values, we started with the EDA. When we observe the data we realize that Hour column is a numerical column but it is a time stamp so we have to treat Hour as a categorical feature**.** Here we make two different datasets by using 'select_dtypes' method. First dataset contains numerical columns and second categorical columns.

```
numeric_features= df.select_dtypes(exclude='object')
categorical_features=df.select_dtypes(include='object')
```

```
[21] numeric_features.head()
```

| | Rented Bike Count | Temperature(°C) | Humidity(%) | Wind speed (m/s) | Visibility (10m) | Dew point temperature(°C) | Solar Radiation (MJ/m2) | Rainfall(mm) | Snowfall (cm) |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 254 | -5.2 | 37 | 2.2 | 2000 | -17.6 | 0.0 | 0.0 | 0.0 |
| **1** | 204 | -5.5 | 38 | 0.8 | 2000 | -17.6 | 0.0 | 0.0 | 0.0 |
| **2** | 173 | -6.0 | 39 | 1.0 | 2000 | -17.7 | 0.0 | 0.0 | 0.0 |
| **3** | 107 | -6.2 | 40 | 0.9 | 2000 | -17.6 | 0.0 | 0.0 | 0.0 |
| **4** | 78 | -6.0 | 36 | 2.3 | 2000 | -18.6 | 0.0 | 0.0 | 0.0 |

```
categorical_features.head()
```

| | Hour | Seasons | Holiday | Functioning Day | weekday | month | year |
|---|---|---|---|---|---|---|---|
| **0** | 0 | Winter | No Holiday | Yes | Thursday | January | 2017 |
| **1** | 1 | Winter | No Holiday | Yes | Thursday | January | 2017 |
| **2** | 2 | Winter | No Holiday | Yes | Thursday | January | 2017 |
| **3** | 3 | Winter | No Holiday | Yes | Thursday | January | 2017 |
| **4** | 4 | Winter | No Holiday | Yes | Thursday | January | 2017 |

# Categorical columns value count

First, we check counts of each sub-categories of categorical data. Here the results.

## Seasons

```
Column name  :  Seasons


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

## Holiday

```
Column name  :  Holiday


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

## Functioning_day

```
Column name  :  Functioning Day


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

## Weekday

```
Column name  :  weekday


Sunday       1296
Wednesday    1272
Tuesday      1272
Thursday     1248
Saturday     1248
Friday       1224
Monday       1200
Name: weekday, dtype: int64
```

## Month

```
Column name  :  month


January      744
March        744
May          744
July         744
August       744
October      744
December     744
April        720
June         720
September    720
November     720
February     672
Name: month, dtype: int64
```

## Hours

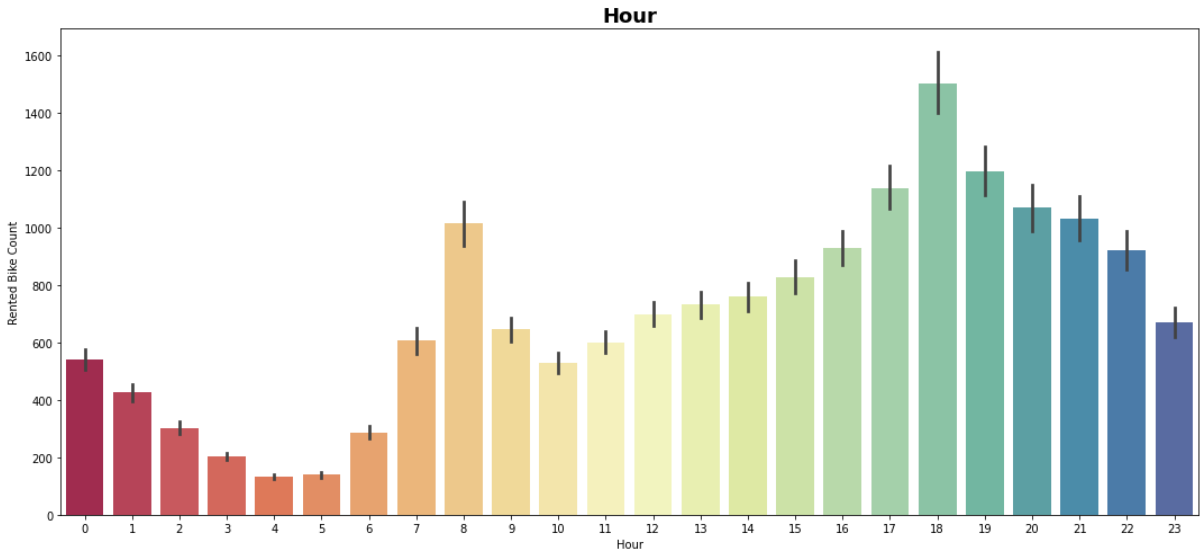
```
Column name  :  Hour


0     365
1     365
22    365
21    365
20    365
19    365
18    365
17    365
16    365
15    365
14    365
13    365
12    365
11    365
10    365
9     365
8     365
7     365
6     365
5     365
4     365
3     365
2     365
23    365
Name: Hour, dtype: int64
```

## Year

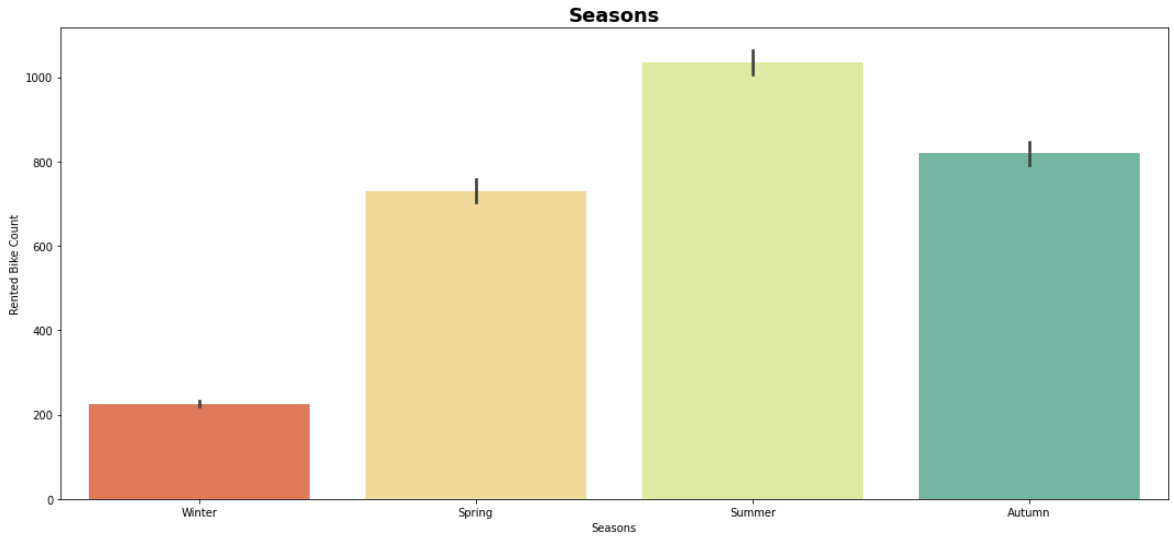
```
Column name  :  year


2018    8016
2017     744
```

# Visualization of Value count of Categorical column

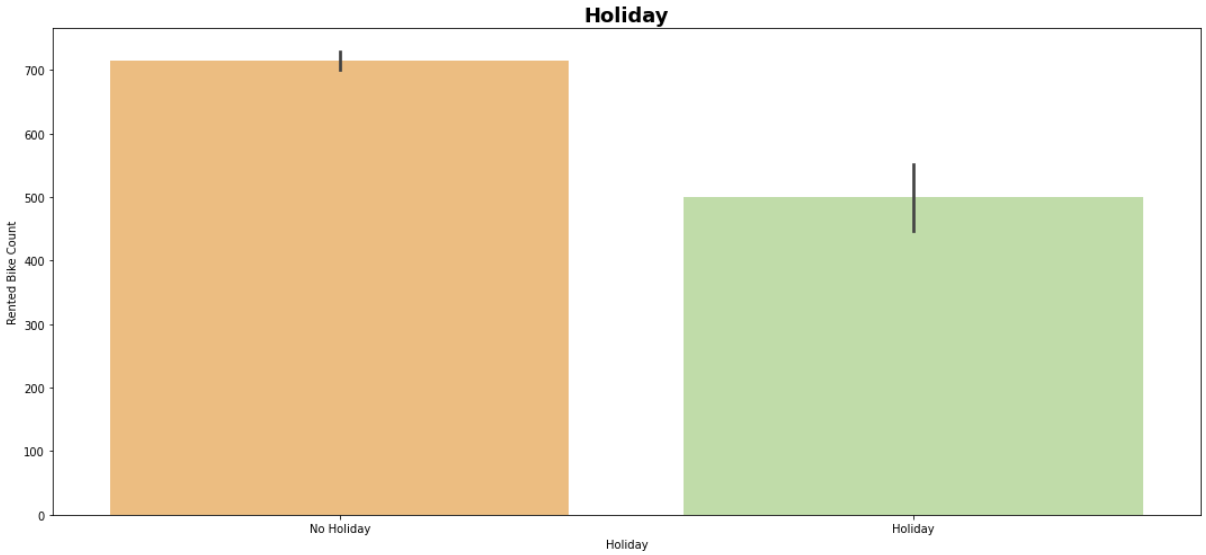
We visualized categorical columns value counts using barplot, here we compare 'Rented Bike Count' column with each categorical columns.



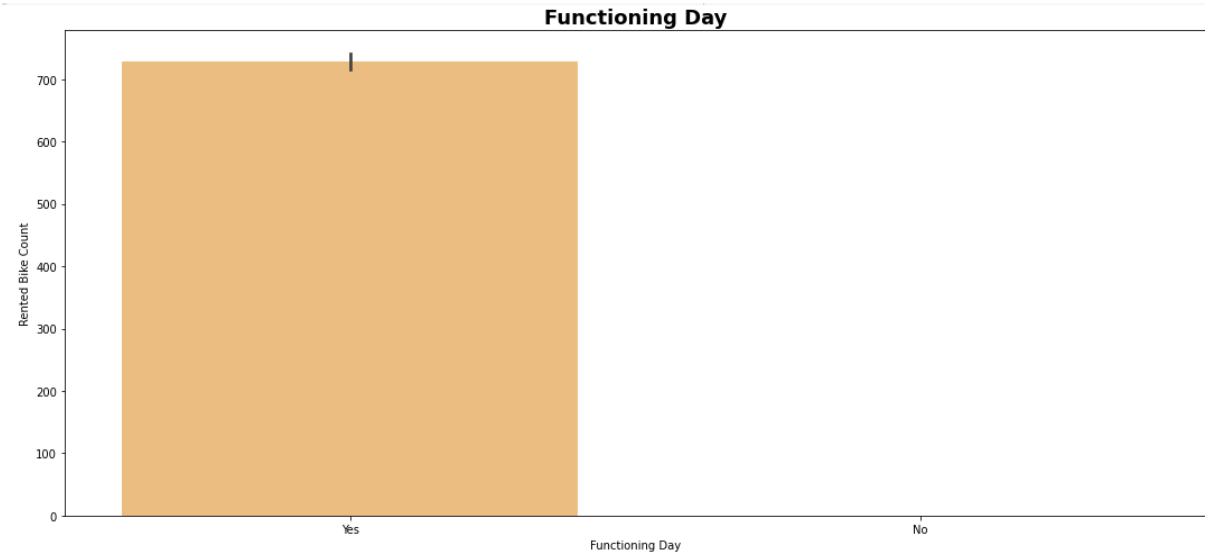

This graph shows that demand of bike is more during pick hours like in morning 8'o clock and in Evening 18'o clock. After busy hours, demands of bikes are fluctuating. Least demand of bike is at mid night.



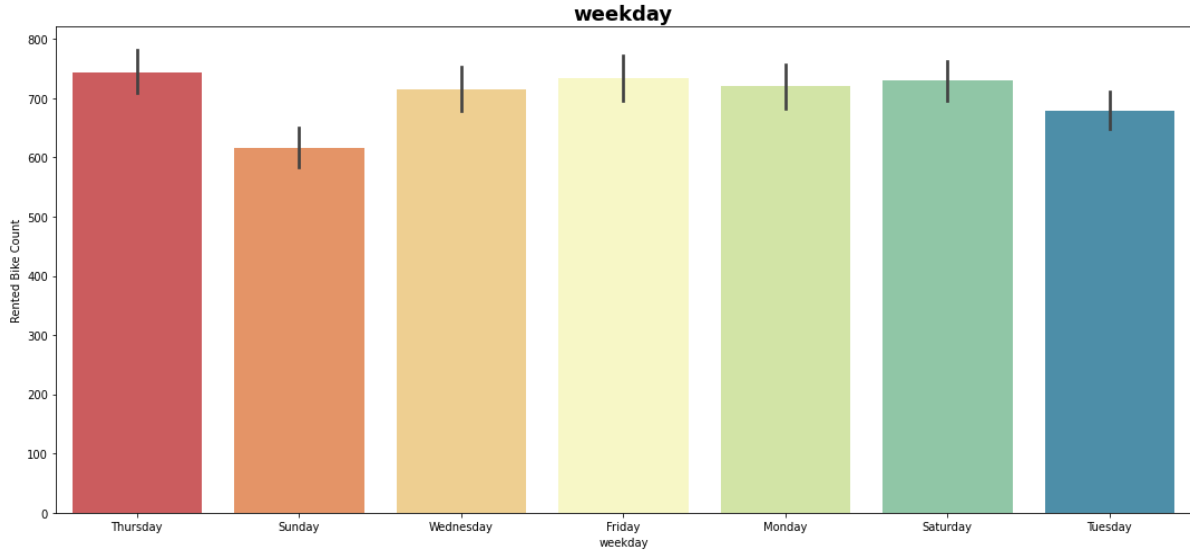

This graph show four types of seasons in a year. And each season records of bike rent. Summer seasons are vacation seasons so most of the peoples have rented bike that's why this season records highest bike rent counts, autumn and spring seasons balance their counts between (700-800) and winter season have least number of bike rent counts.



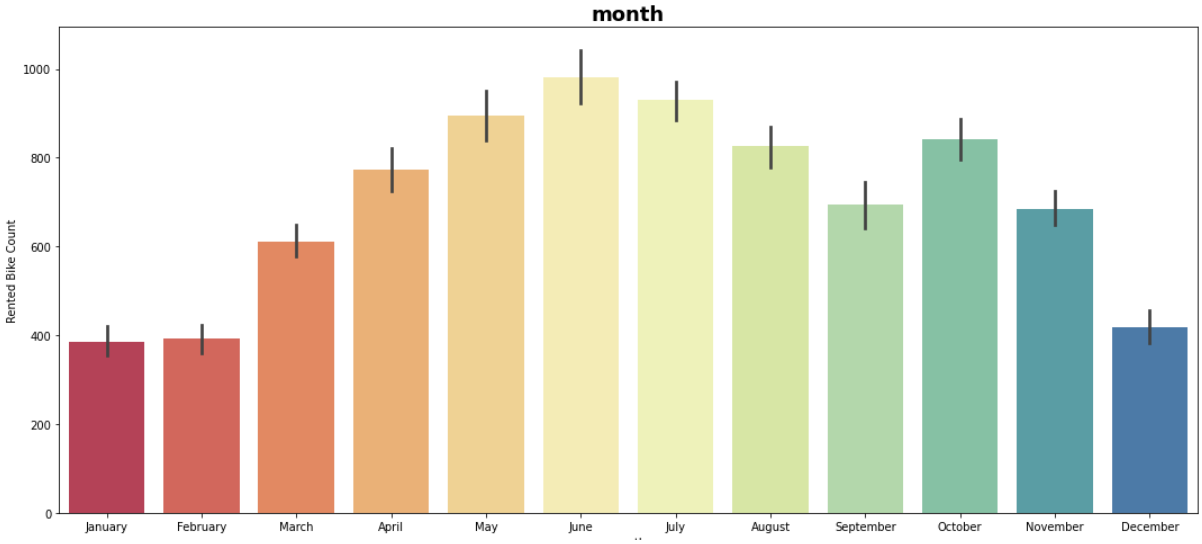

This graph shows Holiday's in a year. As we can see working days are more than non-working days. So may be office people are using bicycle for travelling.



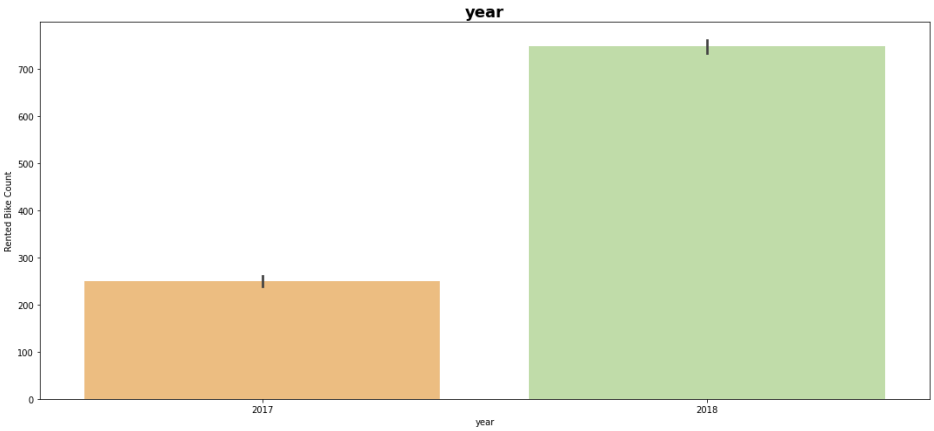

This graph shows functioning days [Yes] and non-functioning days [No]. this bar graph have very less counts on non-functioning days so if we compare holiday and functioning day together we can conclude that office people have more demand rather than other people.




In this weekday barplots Thursdays have highest bike rented count. Other days are also fluctuating around highest counts, Sunday had lesser count than other days.



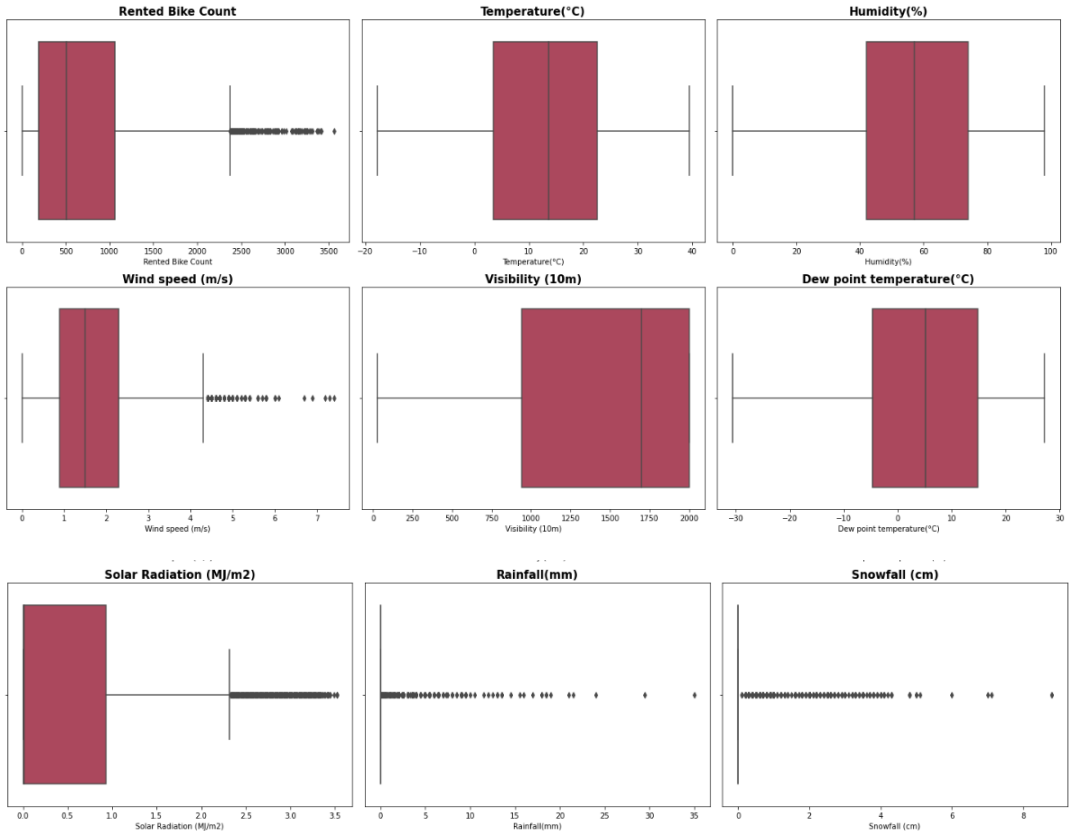

In this barplot, June month have highest and January month have least bike rent count. We observed that since December to February have a winter season so this may be the reason of less demand of bikes. Summer season have begun from March so demand of bike constantly increasing throughout June. Then demand of bike fluctuating till November.




This plot shows two years (2017-2018) of bike rented counts record.2017 had very less count of bike rent but as market grows in 2018 the counts of bike rent also increase, and records bike rents are double than previous year.

# Outliers of Numerical feature

We check outliers in Numerical features using seaborn's boxplot.

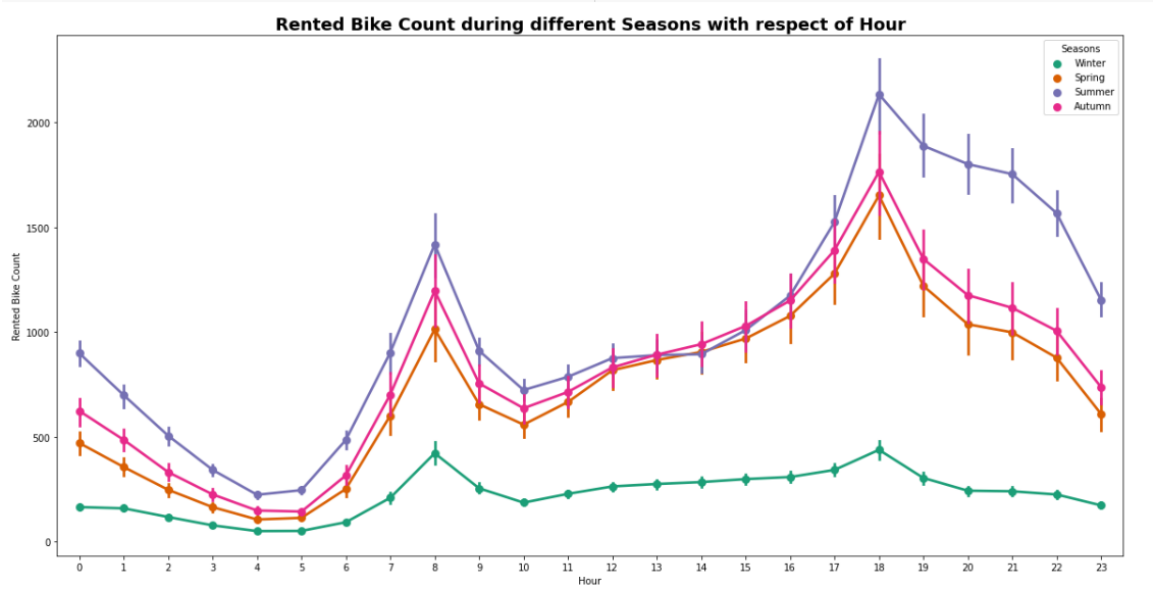



From this boxplot we observed that Rented bike count (Dependent Feature), wind speed, solar radiation, rainfall, and snowfall these columns have outliers. Temperature, humidity, visibility and dew point temperature are normally distributed columns.

# Rented Bike count during different Categorical features with respect to Hour

We fixed hours on x-axis and rented bike count on y-axis, then apply categorical features as hue one by one and observer each graphs.

## Seasons

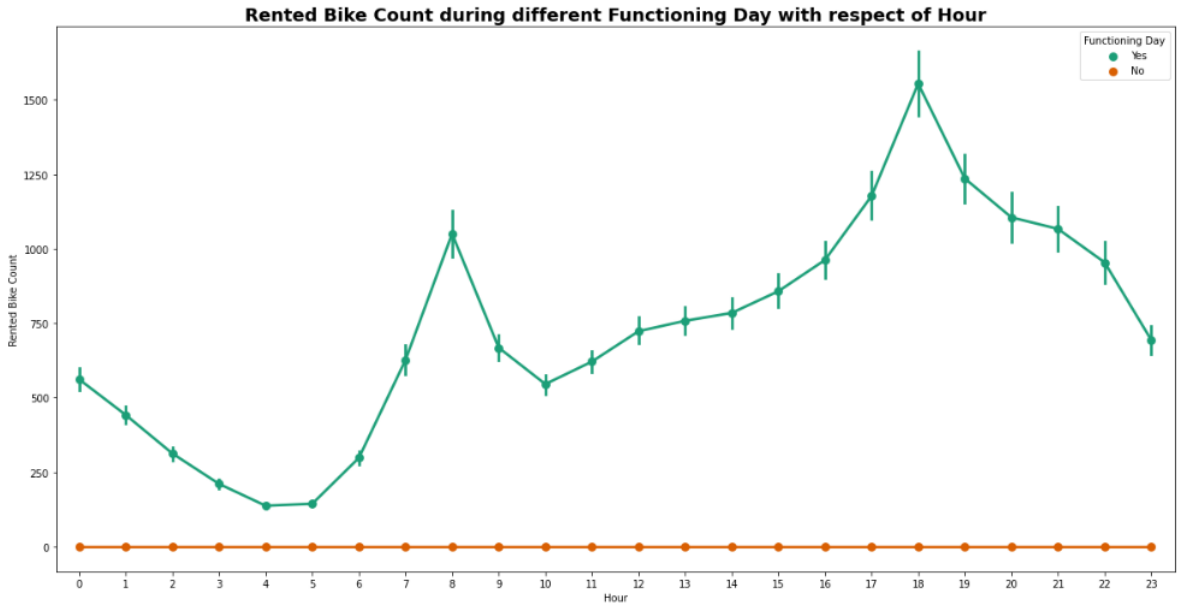



This graph shows rented bike counts during different seasons with respect to hours. In legend Denoting seasons with different colours. During summer season demand of bike rent is always higher only drop for an hour in afternoon time. Winter season have least count then remaining seasons.

## Holiday

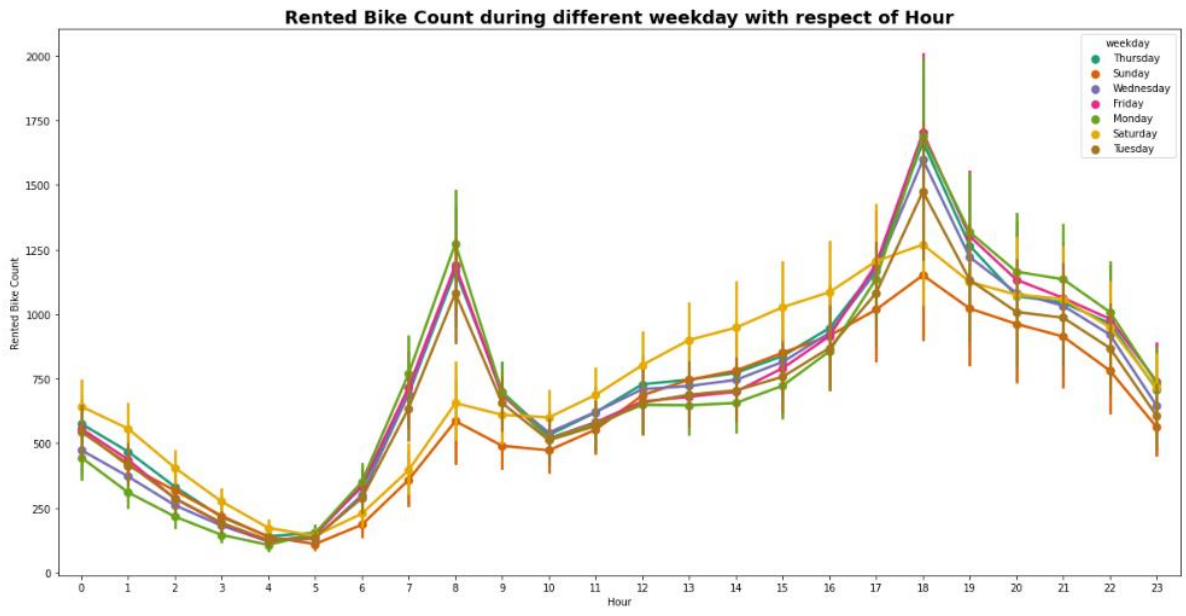

We can see demand of bike is increasing only on pick hours of working days. may be users of bikes are more form office.

## Functioning Days



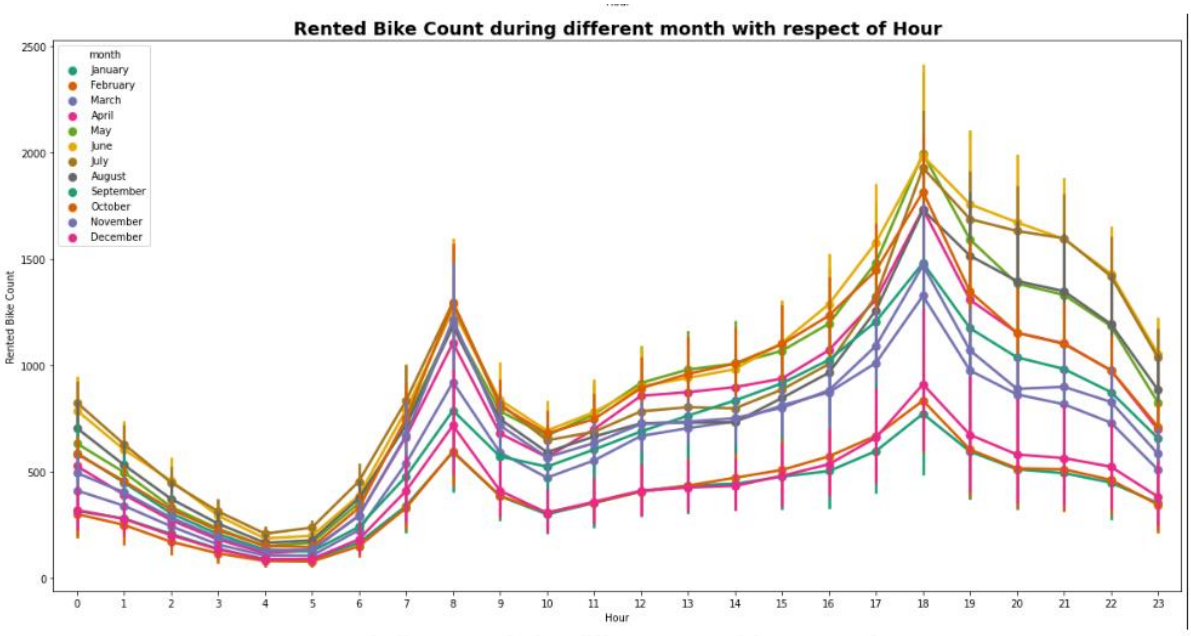

In this graph we can see demand of bike rented is more during functioning day of pick hours. There was no bike demand on a non-functioning day.

## Weekday



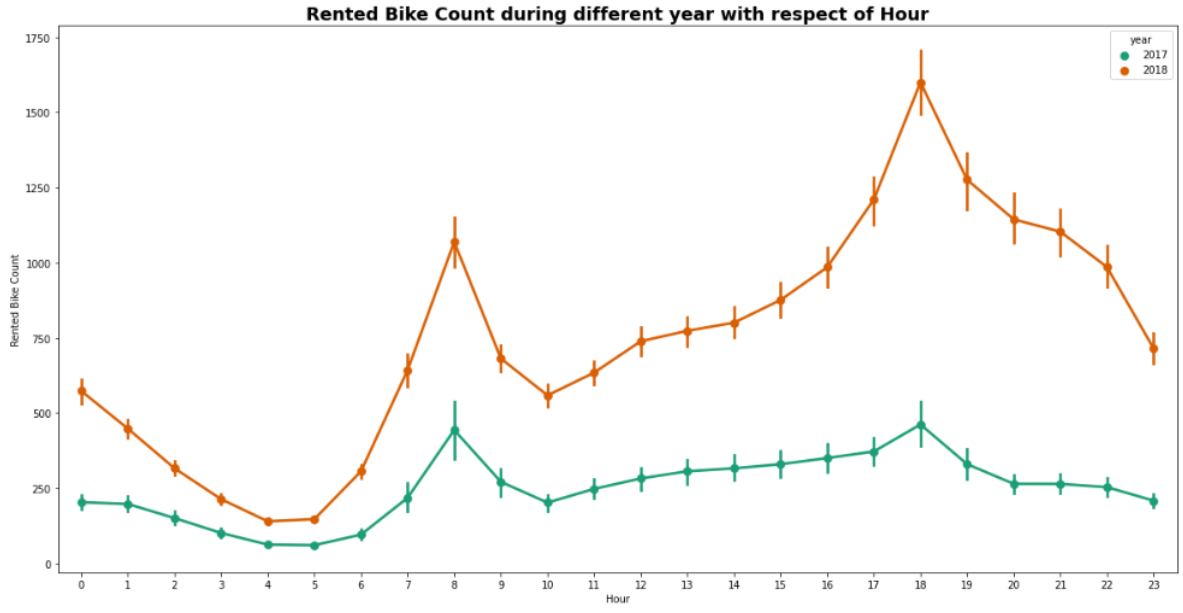

This graph shows counts of bike rented is high during hours of working days. Saturday afternoon and mid-night time counts of bike is higher than other days.

## Month




In this pointplot we can see that June month have highest bike demand through-out 24/7.january and December month have least counts in each hours.

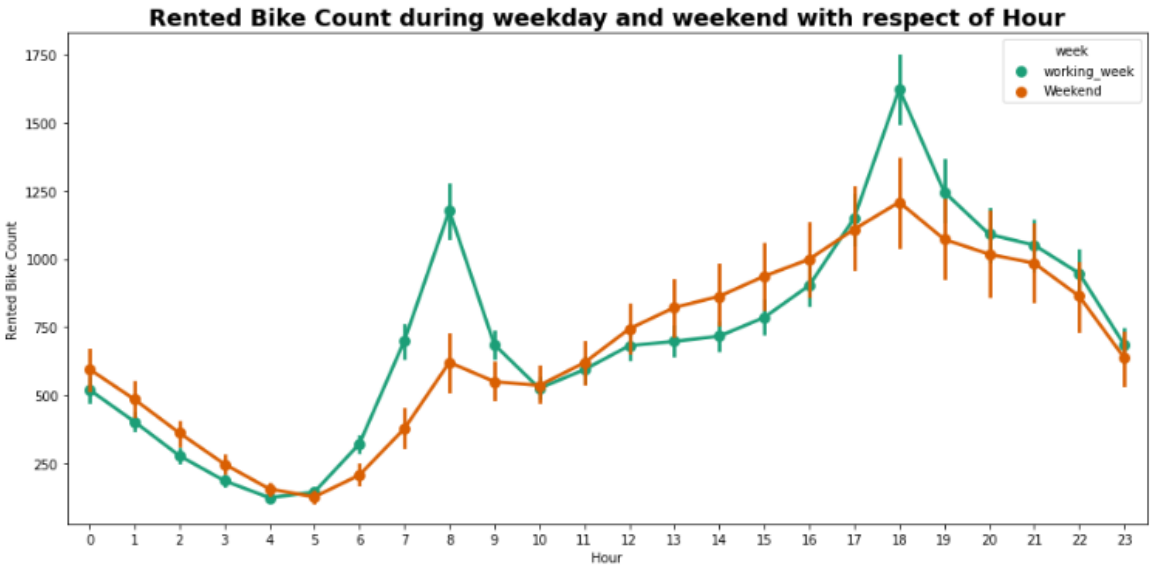
## Year




This graph shows huge difference of records where initial year we saw very less demand of bike, but as popularity of rented bike increase people also increase demand of bike in next year.

## Extracting Weekday column

We extract working-week (Mon-Fri) and weekend (Sat-Sun) from Weekday column. We get 6216 is weekday and 2544 is weekend.

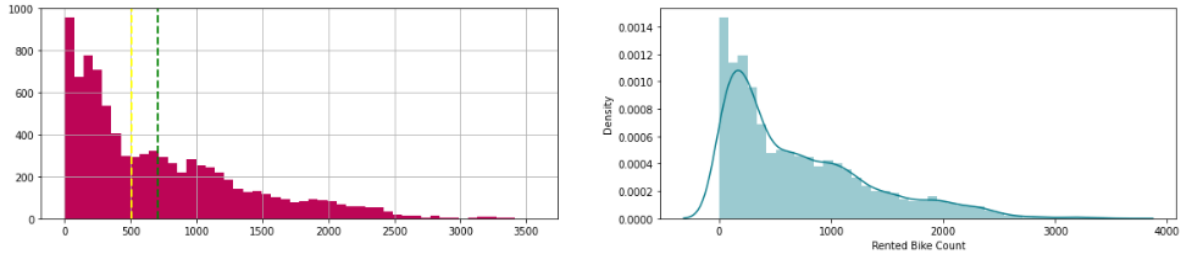



As we can see in this pointplot demand of bike during working week is high in pick hours like in morning 6-8 and evening 5 to till night. And weekend demand of bike have in mid-night and afternoon.

# Operation on Numerical Feature

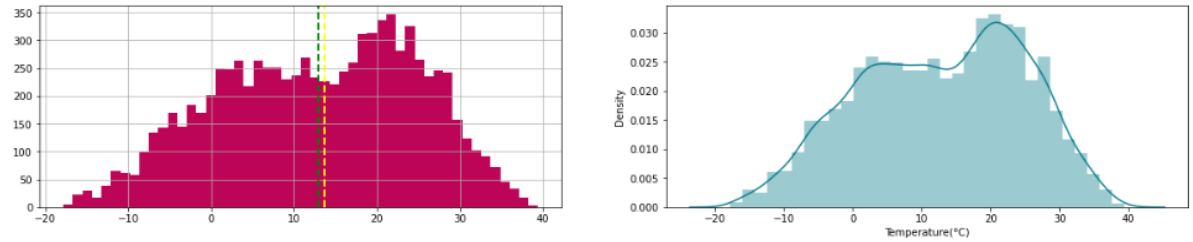
Here we plot histogram and distplot together to find skewedness or mean and median of columns.

## Rented Bike Count



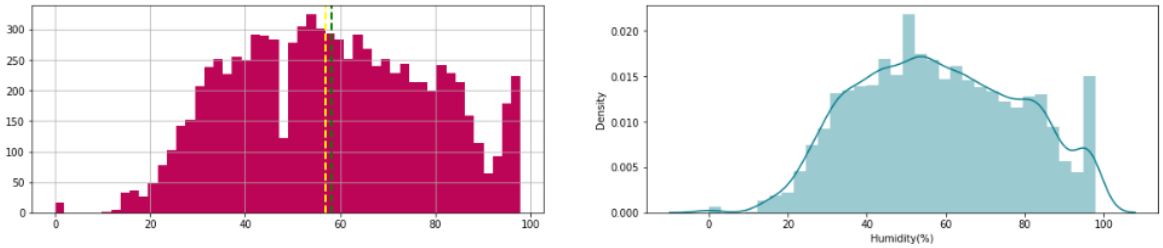

Rented Bike Count is dependent feature in dataset. The distplot graph shows rented bike count column is right skewed. From the histogram we can see because of skewedness in column their mean and median also skewed.

## Temperature (°C)



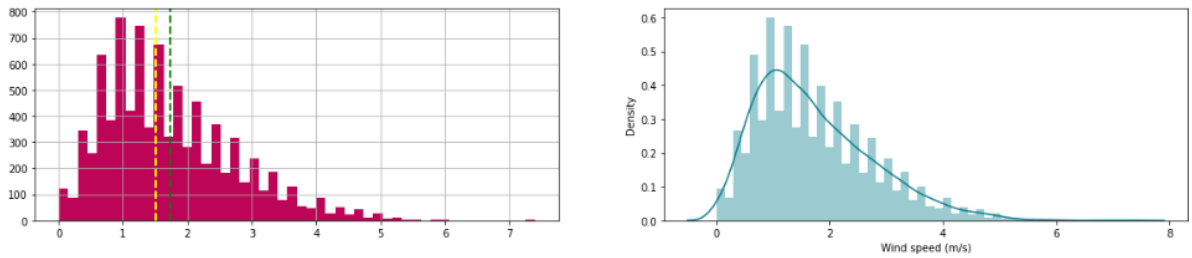

Temperature is independent feature in dataset.as we can see in both graph this column is normally distributed. And their mean and median is also normal.

## Humidity (%)



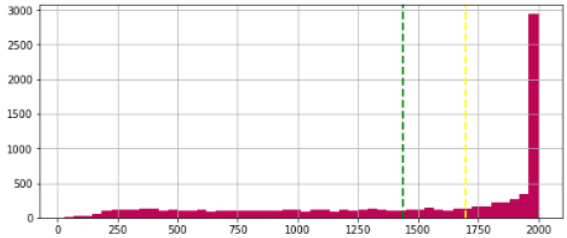

Humidity is independent feature in dataset.as we can see in both graph this column is normally distributed. And their mean and median is also normal.

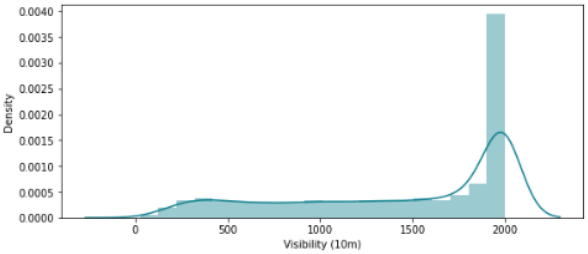
## Wind Speed (m/s)




Wind Speed is also Independent feature in dataset. The distplot graph shows Wind speed column is right skewed. from the histogram we can clearly see because of skewedness in column their mean and median also skewed.

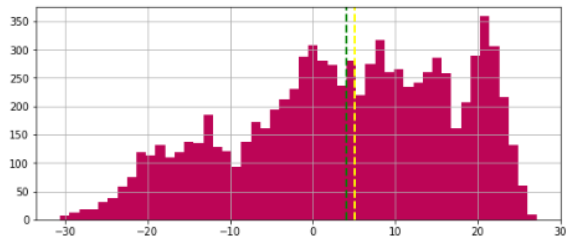
## Visibility (10mm)

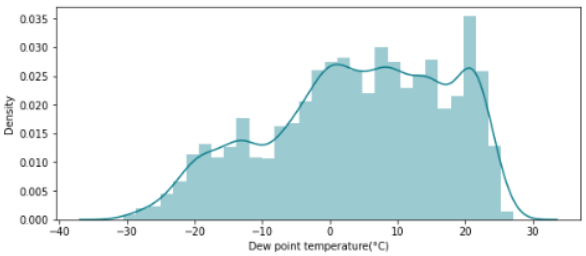





Visibility also Independent feature in dataset. The distplot graph shows visibility column is left skewed. With the histogram we are clearly see that visibility rate from 0 to 1999mm is very less as compare to highest visibility rate is 2000mm.so this is reason of skewedness because of skewedness in column their mean and median also skewed.

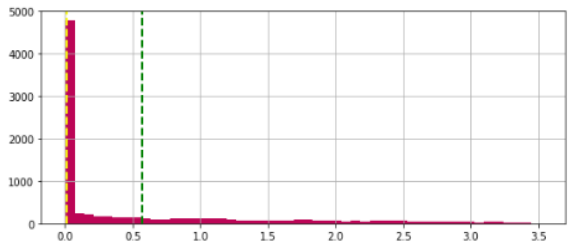
## Dew point Temperature (°C)

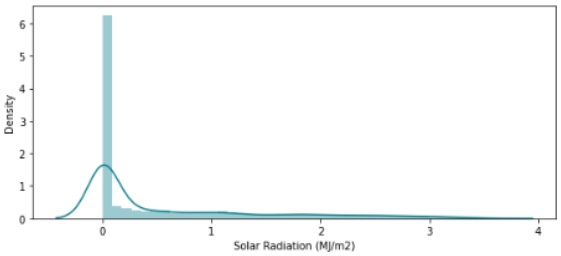





Dew Point Temperature is independent feature in dataset.as we can see in both graph this column is normally distributed. And their mean and median is also normal.

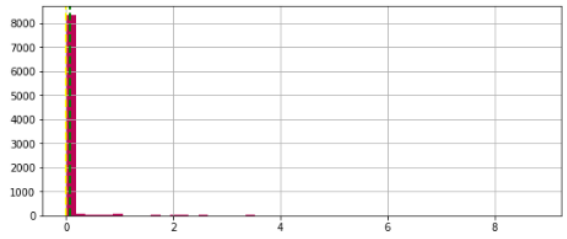
## Solar Radiation (MJ/m2)

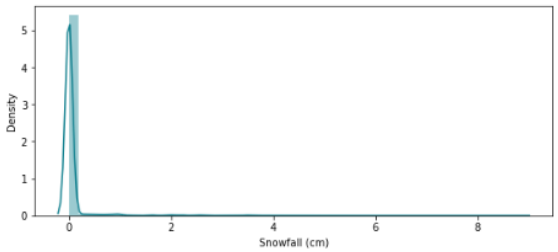





In the dataset,Solar Radiation is an independent feature.The displot it clearly shows that,this column is strongly right skewed. From the histogram we can see because of skewedness in column their mean and median also skewed.

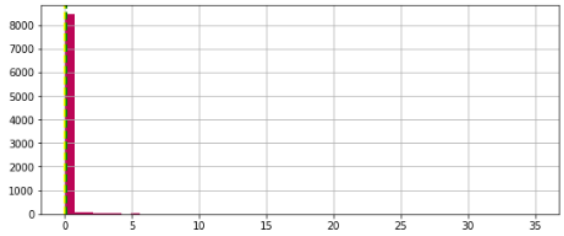
## Snowfall (cm)

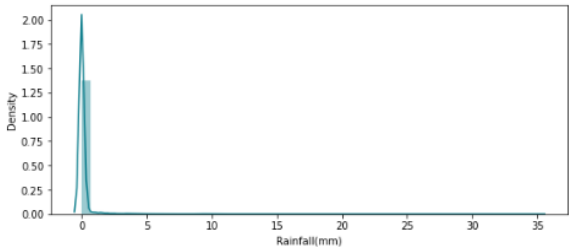



In the dataset, snowfall is an independent feature. The display makes it pretty obvious that this column is highly right-skewed. Because of the skewedness in the column, we can see by the histogram that mean and median are equally skewed.

## Rainfall (mm)



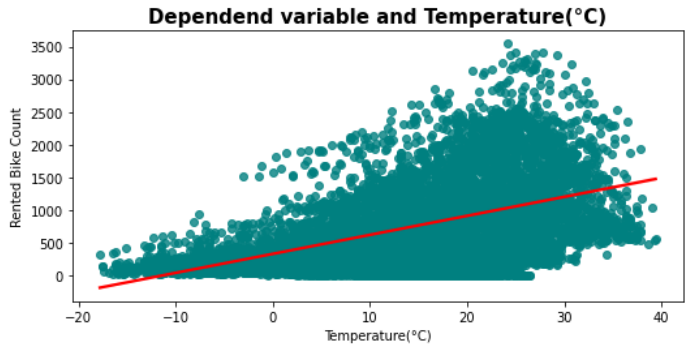




Rainfall is an independent feature in dataset.with displot it clearly see that,this column is major right skewed. From the histogram we can see because of skewedness in column their mean and median also skewed.

# Relation of Numerical features with Dependent variable

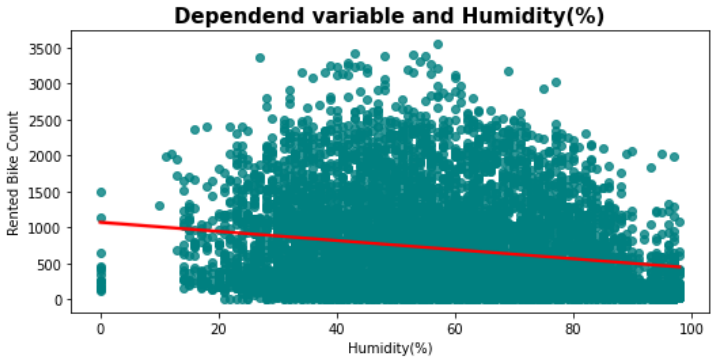
Here we see linear relationship between dependent and independent features. These regression plots shows that some of our features are positive linear and some are negative linear in relation to our target variable.

## Temperature (°C)



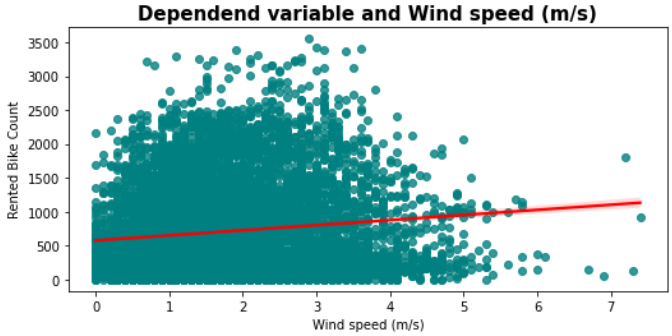

Here regplot shows linear relationship between rented bike count and temperature. We can say that, the data is linearly correlated with each other because as temperature increases the demand of bike rent is also increasing and it find a best fit positive line.

## Humidity (%)



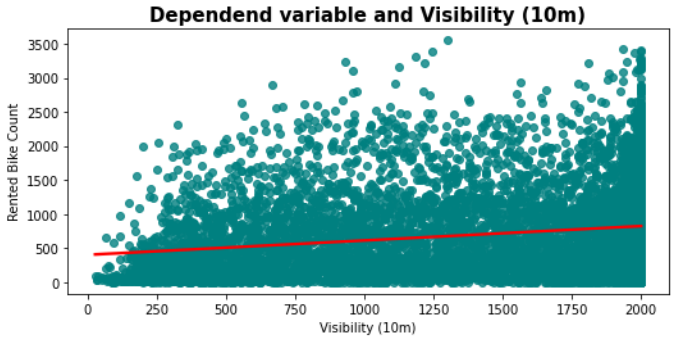

In this reg-plot, the rented bikes count and Humidity are correlated linearly. Because the demand for bikes decrease as humidity rises and a negative line of best fit forms, we may would that these two factors are linearly associated.

## Wind Speed (m/s)



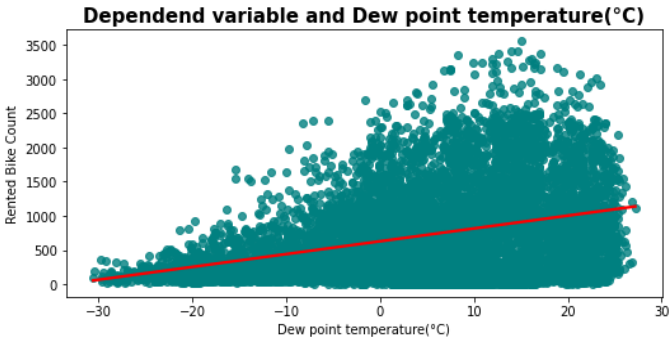

In this reg-plot, the rented bikes count and wind speed are correlated linearly. Because the demand for bikes increases as wind speed goes up and a positive line of best fit forms, we would say that these two factors are linearly associated.

## Visibility (10mm)



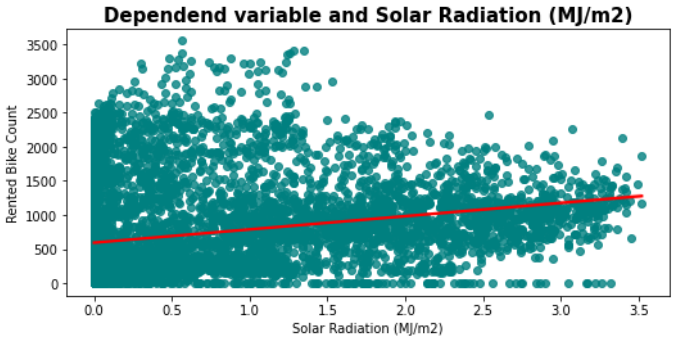

In this reg-plot, the number of rented bikes and visibility are correlated linearly. Because the demand for bikes rent is increases as visibility rise and a positive line of best fit forms, we would say that these two factors are linearly associated.

## Dew point Temperature (°C)



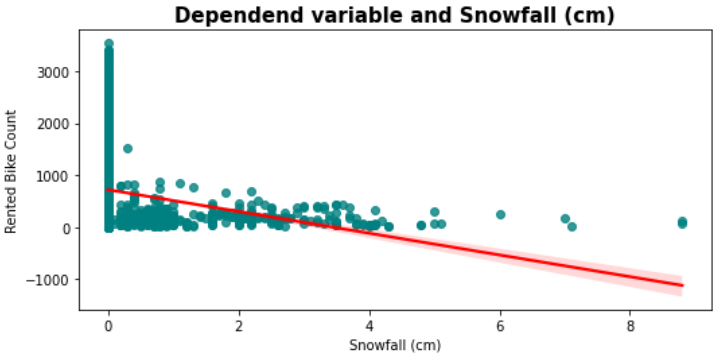

Here reg-plot, shows linear relationship between rented bike count and dew point temperature. We can say that, this is linearly correlated with each other because as dew point temperature increases the demand of bike rent also increases and it find a best fit positive line.

## Solar Radiation (MJ/m2)



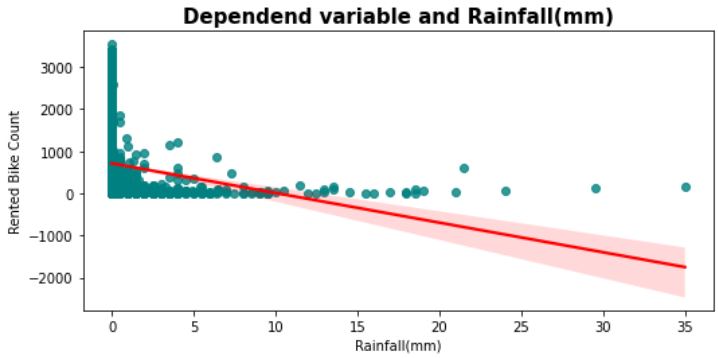

In reg-plot, rented bike count and solar radiation these two feature are not linearly correlated with each other. As we can see in graph as solar radiation is keep on increasing but demand of bike rent keeps decreasing. Although it find positive best fit line.

## Snowfall (cm)




In reg-plot, there is no linear relationship between rented bike count and snowfall, because it has huge data on 0 point rather than other points. Also it is generating negative best fit line.

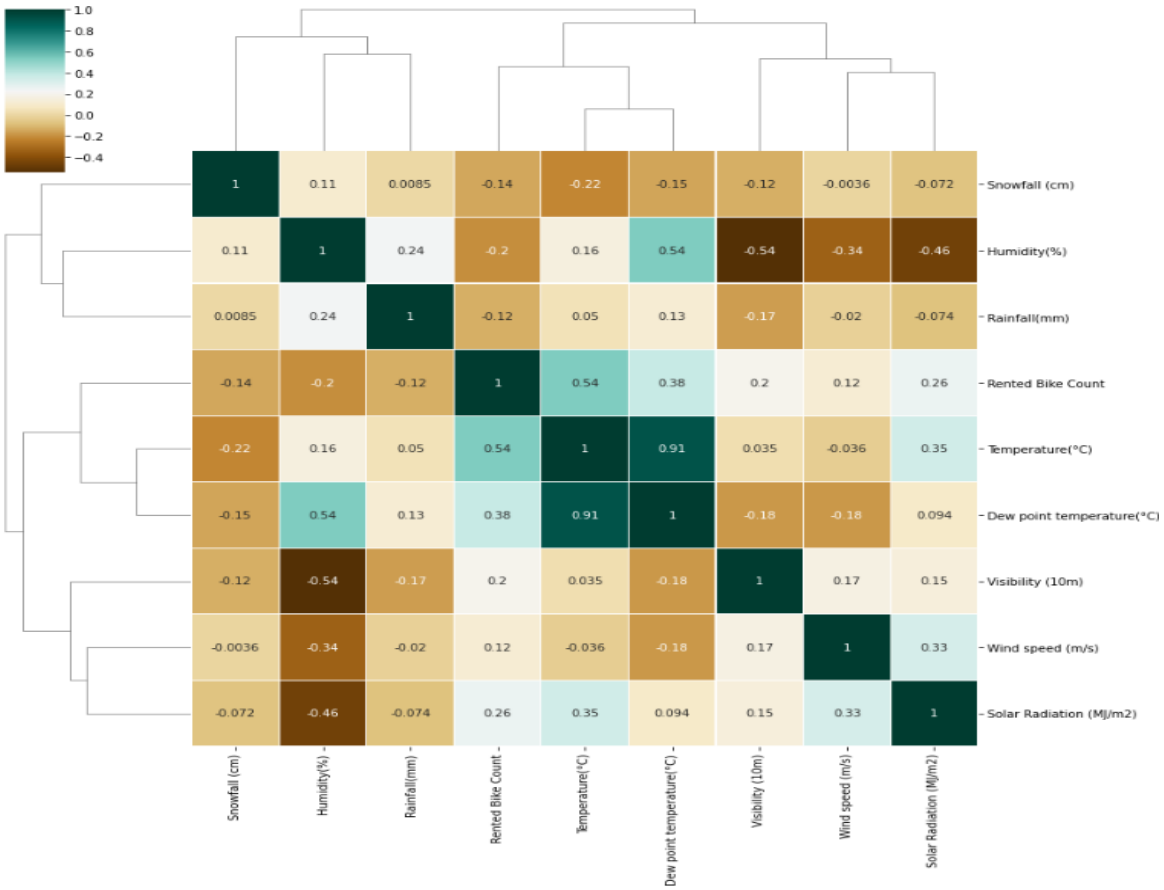
## Rainfall (mm)




In regplot, there is no linear relationship between rented bike count and rainfall, because it has huge data on 0 point rather than other points. Also it is generating negative best fit line.

# Multicollinearity

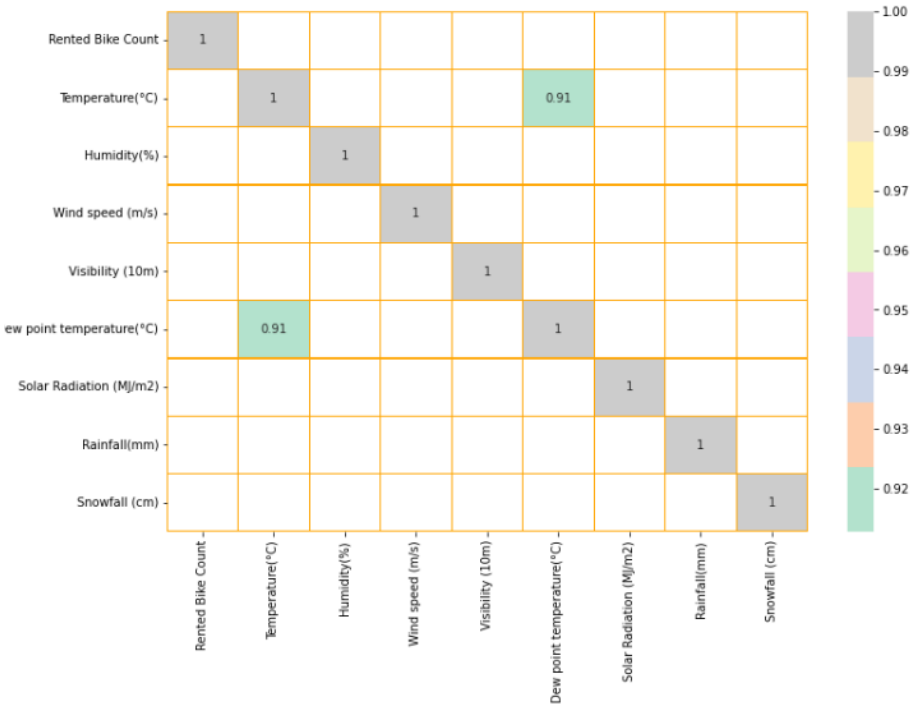
Multicollinearity happens when independent variables in the regression model are highly correlated to each other.




```
Rented Bike Count            1.000000
Temperature(°C)              0.538558
Humidity(%)                 -0.199780
Wind speed (m/s)             0.121108
Visibility (10m)             0.199280
Dew point temperature(°C)    0.379788
Solar Radiation (MJ/m2)      0.261837
Rainfall(mm)                -0.123074
Snowfall (cm)               -0.141804
Name: Rented Bike Count, dtype: float64
```

From the above observation of correlation table we found out that humidity, dew point temperature and temperature column correlated with each other. But we cannot drop all these column without cross verify.

We create one more heatmap to see only highly correlated columns for that we apply threshold value. If correlation of columns is more then set threshold then it is consider as highly correlated column.




From this heatmap we find out dew point temperature and temperature is highly correlated column.

## Checking correlation using VIF method

```
def calc_vif(X):

    # Calculating VIF
    vif = pd.DataFrame()
    vif["variables"] = X.columns
    vif["VIF"] = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]

    return(vif)
```

```
calc_vif(df[[i for i in df.describe().columns if i not in ['Rented Bike Count','Dew point temperature(°C)']]])
```

| | variables | VIF |
|---|---|---|
| 0 | Temperature(°C) | 3.166007 |
| 1 | Humidity(%) | 4.758651 |
| 2 | Wind speed (m/s) | 4.079926 |
| 3 | Visibility (10m) | 4.409448 |
| 4 | Solar Radiation (MJ/m2) | 2.246238 |
| 5 | Rainfall(mm) | 1.078501 |
| 6 | Snowfall (cm) | 1.118901 |

After calculating VIF we came to know that these columns are affecting VIF score too, so we chose dew point temperature because it is less correlated with our dependent feature so we drop dew point temperature column from dataset permanently.

# Converting Continues Variable to Categorical Variable for simplify prediction process

Here we convert some dataset independent feature into categorical variable to make easier prediction process. We apply lambda function on solar radiation, snowfall, rainfall, and visibility columns for easy conversion.

```
df['Visibility']= df['Visibility (10m)'].apply(lambda x: 1 if x>=2000 else 0)
df['Rainfall']= df['Rainfall(mm)'].apply(lambda x:1 if x>=0.148687 else 0)
df['Snowfall']= df['Snowfall (cm)'].apply(lambda x:1 if x>=0.075068 else 0)
df['Solar_Radiation']= df['Solar Radiation (MJ/m2)'].apply(lambda x:1 if x>=0.56911 else 0)
```

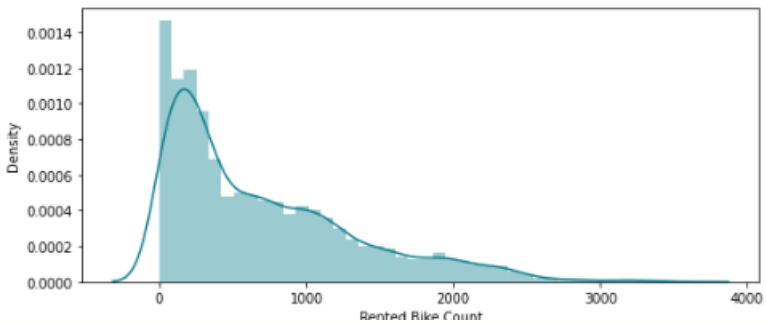
# Handling Categorical variable

To handle categorical variable we use simple encoding technique it is ‘Dummy Encoding’. Dummy encoding is one that takes only the value 0 or 1 to indicate the absence or presence of some categorical effect that may be expected to shift the outcome.it is easiest encoding method.

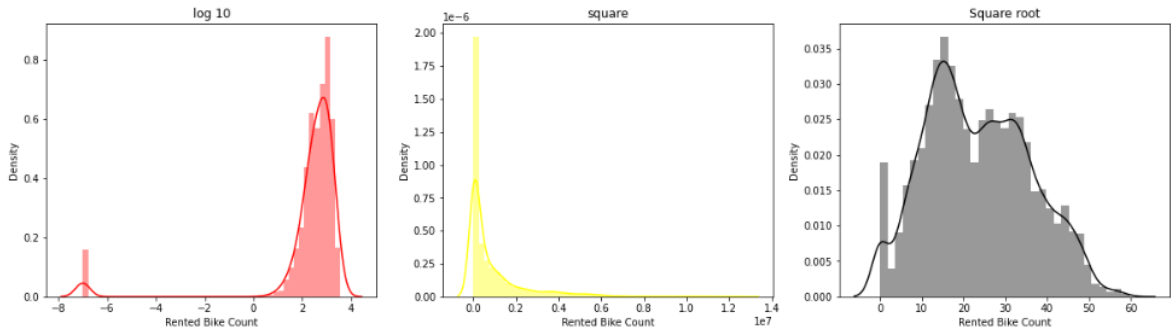
# Normalizing right skewed from dependent feature

Here we apply more than one type of transformation techniques to normalize dependent feature. Then select best technique which is perfectly normalized column.

## Before Normalization



## After Normalization




We apply three type to transformation techniques those are deal with skewed data's.

- First we apply log transformation, among the various forms of transformations used to change skewed data to roughly adhere to normality, the log transformation is likely the most common. But in our case it wasn't work as we can see in pink colour graph data seem much skewed.

- Second we apply Square transformation, here also same scenarios occur.

- Third we apply square root transformation this technique is perfectly remove right skewedness from our column, so we will go with is technique on our dependent variable.

## Model Performed

We check model performance of different types of Supervised Machine Leaning algorithms one by one.

1. Linear Regression With Regularization (Lasso & Ridge)
2. Polynomial Regression
3. Random Forest
4. Decision Tree

We define a function to train and evaluating model with difference parameters.

- MSE
- MAE
- RSME
- R2
- Adjusted R2

Before training model we have to normalize our data by using MinMaxScaler.it help use to scale our feature between -1 to 1.

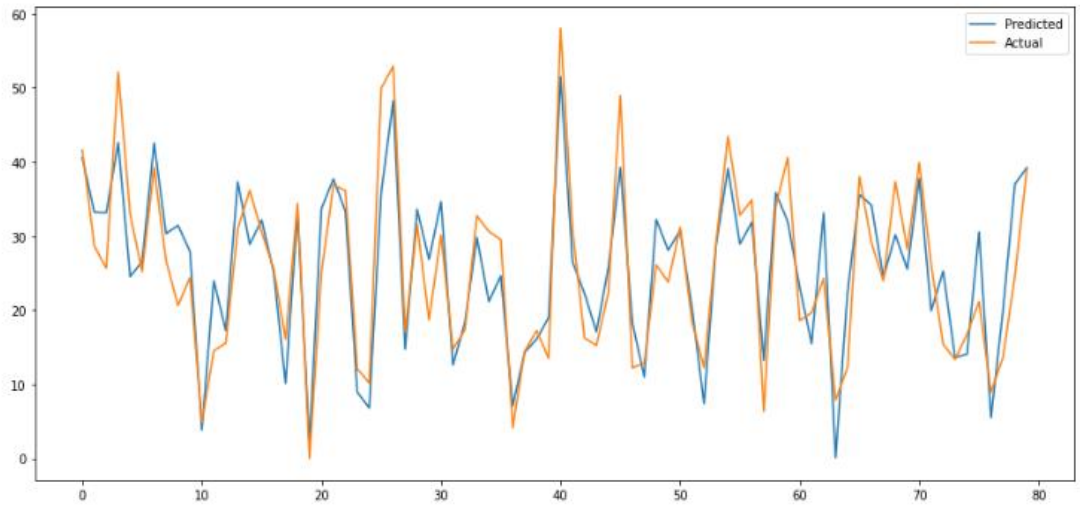
# Linear Regression with regularization

Linear regression is basic and easiest Supervised Machine Leaning algorithm. It is statistical method that is used to make predicative analyses.

The main goal of this algorithm is to find best fit line that means error between predicted value and actual value should be minimum. Best fit line indicate model have very less error.

This algorithm perfectly show relationship between dependent and independent variable. So we apply this algorithm for checking model performance.

```
Training score  = 0.7902269075496007

Best Parameters & Best Score

None

Evalution Matrix

MAE : 210.6768562260802
MSE : 97363.04473354678
RMSE : 312.0305189136902
R2 : 0.7620654672586054
Adjusted R2 :  0.756731110616108


coefficient

[ 24.85566777 -10.07871822  -0.42937367  -0.21141619 -13.0402467
   0.23423846   2.41242433  -2.14182676  -5.15584563  -7.16015048
  -9.90992002  -9.59912374  -4.68272993   1.31395712   6.49097381
   0.65671163  -3.82049494  -3.95131531  -2.67027725  -2.78849184
  -2.68270208  -1.75420059   0.04864875   4.38976513  10.20085171
   7.59426362   6.59896267   6.72747659   5.06167821   1.67165619
  -2.97955077  -2.96579509  -8.30128532   2.75593002  29.0201709
  -1.05684208  -0.07706871  -0.70419649  -0.29984146  -1.06685328
   3.40122979  -0.68740104   1.44871636  -0.08195272   1.2638585
  -0.08982535  -1.90566877   0.81268445]

Intercept

-9.709626280378156

 ploting the graph of Actual and predicted only with 80 observation
```




This lineplot shows how well this Algorithm fit to our model with minimum residual.

# Regularization

Regularization resolves the over-fitting problem, which affects the accuracy level of the model. Regularization is executed by the addition of the "penalty" term to the best-fit equation produced by the trained data.

# Regularization (Lasso- L1)

The full form of LASSO is the Least Absolute Shrinkage and Selection Operation. As the name suggests, LASSO uses the "shrinkage" technique in which coefficients are determined, which get shrunk towards the central point as the mean.

- Performs L1 regularization, i.e. adds penalty equivalent to absolute value of the magnitude of coefficients
- Minimization objective = LS Obj + α * (sum of absolute value of coefficients)

```
Training score  = 0.7902086120456122
```

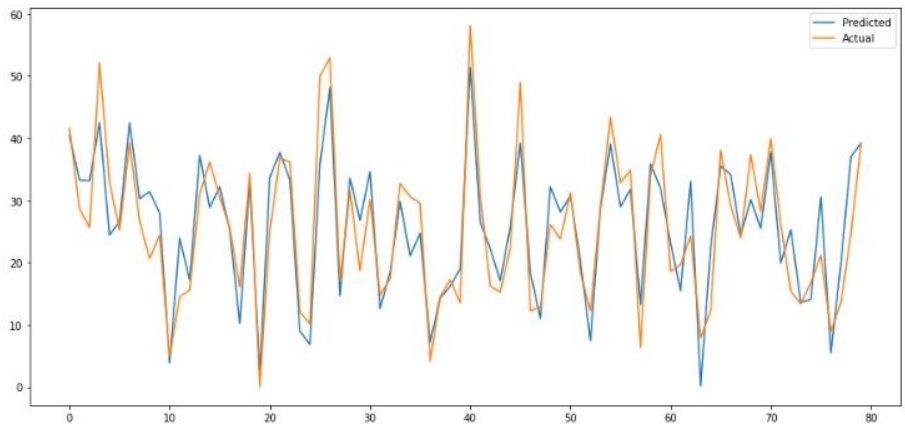
```
Best Parameters & Best Score
```

```
The best parameters found out to be :{'alpha': 0.0014}
where model best score is:  0.7863509105820757
```

```
Evalution Matrix
```

```
MAE : 210.98378286630376
MSE : 97743.31929081825
RMSE : 312.6392798271168
R2 : 0.761136157279183
Adjusted R2 :  0.7557809660364931
```

```
ploting the graph of Actual and predicted only with 80 observation
```




This graph shows us fitness of this regularization technique to our model with minimum residual.

# Regularization (Ridge – L2)

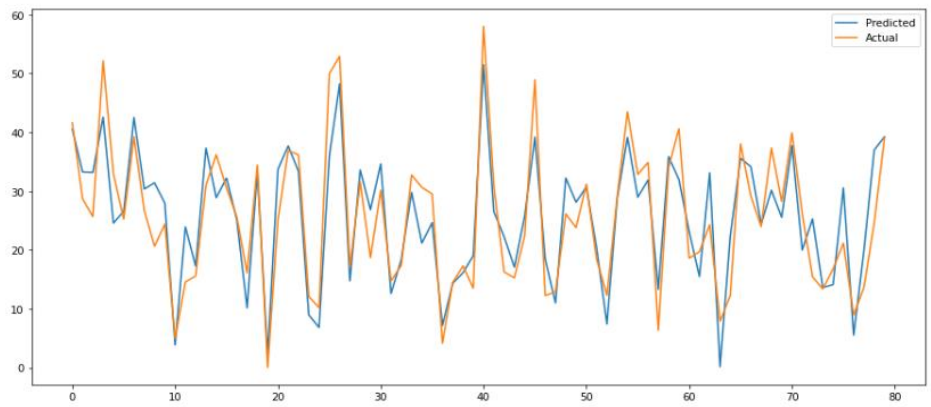
Ridge regression is a technique used to analyse multi-linear regression (multicollinear), also known as L2 regularization. It is applied when predicted values are greater than the observed values.

- Performs L2 regularization, i.e. adds penalty equivalent to square of the magnitude of coefficients
- Minimization objective = LS Obj + α * (sum of square of coefficients).

```
Training score  = 0.7902221237981176


Best Parameters & Best Score


The best parameters found out to be :{'alpha': 0.5}
where model best score is:  0.7862861209197668
```

```
Evalution Matrix


MAE : 210.78508169126954
MSE : 97501.6328762928
RMSE : 312.2525146036342
R2 : 0.7617267873716111
Adjusted R2 :  0.7563848377190363
```

```
ploting the graph of Actual and predicted only with 80 observation
```




This graph shows us fitness of this regularization technique to our model with minimum residual.

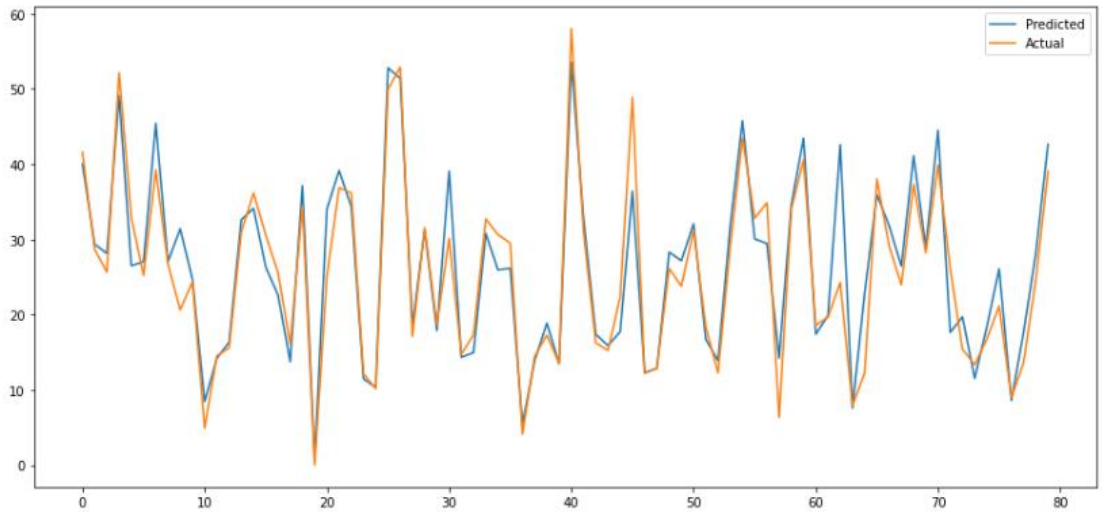
# Polynomial Regression

Polynomial Regression is very popular Machine Learning Model. This is also linear regression but it work with non-linear regression types of dataset.

If your data points clearly will not fit a linear regression (a straight line through all data points), it might be ideal for polynomial regression.

Polynomial regression, like linear regression, uses the relationship between the variables x and y to find the best way to draw a line through the data points. In this algorithm we get curve best fit line.

```
Training score  = 0.9212305400305827

Best Parameters & Best Score

None

Evalution Matrix

MAE : 135.61254379577196
MSE : 50477.57812303929
RMSE : 224.67215698221105
R2 : 0.8766435561101101
Adjusted R2 :  0.7198887389263808


coefficient

[-1.34669375e+09  8.80085315e+01  3.55280841e+01 ...  3.87892811e+10
  3.34273529e+00 -9.55912362e+10]

Intercept

-69304155610.83733
```




This graph shows fitness of this algorithm to our model with minimum residual.

# Decision Tree

A Decision Tree is a Flow Chart, and can help you make decisions based on previous experience. To make a decision tree, all data has to be numerical. We have to convert the non-numerical columns into numerical values. Multi-collinearity does not affect tree base models.

```
Training score  = 0.8807688497254179
```

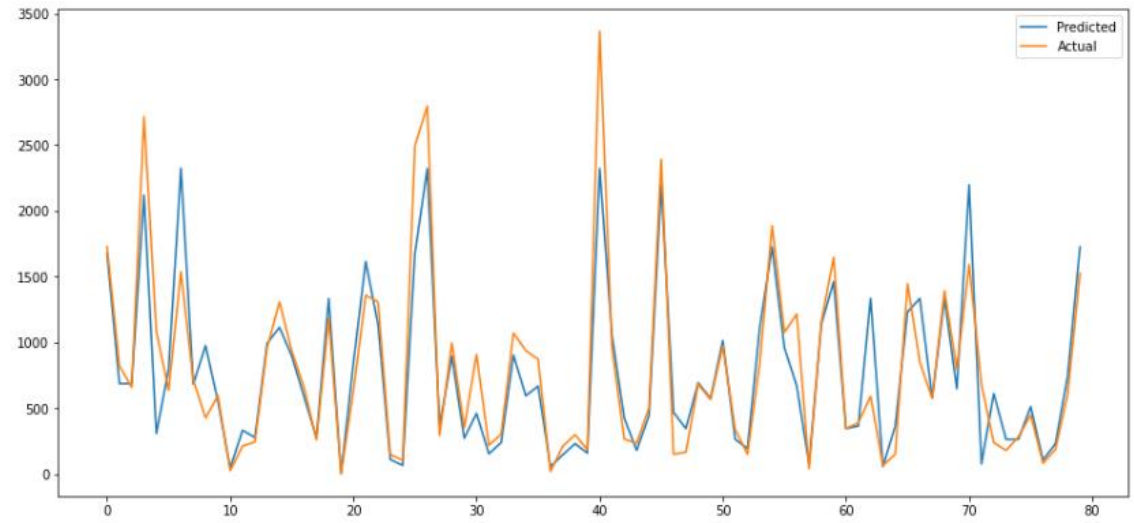
```
Best Parameters & Best Score
```

```
The best parameters found out to be :{'criterion': 'mse', 'max_depth': 25, 'max_features': 'auto', 'min_samples_leaf': 5, 'min_samples_split': 35}
where model best score is:  0.825471774962945
```

```
Evalution Matrix
```

```
MAE : 166.4147006664265
MSE : 72640.25604207265
RMSE : 269.5185634461431
R2 : 0.8224826942616074
Adjusted R2 :  0.8185028574211389
```

```
ploting the graph of Actual and predicted only with 80 observation
```




This graph shows us fitness of this algorithm to our model with minimum residual.

# Random Forest Regression

Random forest is popular Machine learning algorithm that belong to supervised learning techniques.it is based on essential learning technique.it is a method that combine several based model in order to produce an optimal predictive model.

As name suggest Random forest, instead of relying on single decision tree's prediction. It take prediction from each tree and based on higher majority vote of predictions.it predict that correct output.

## Why to use Random Forest

Random forest is a flexible, easy-to-use machine learning algorithm that produces, even without hyper-parameter tuning, a great result most of the time. It is also one of the most-used algorithms, due to its simplicity and diversity (it can be used for both classification and regression tasks).

```
Training score  = 0.9177937685104807
```

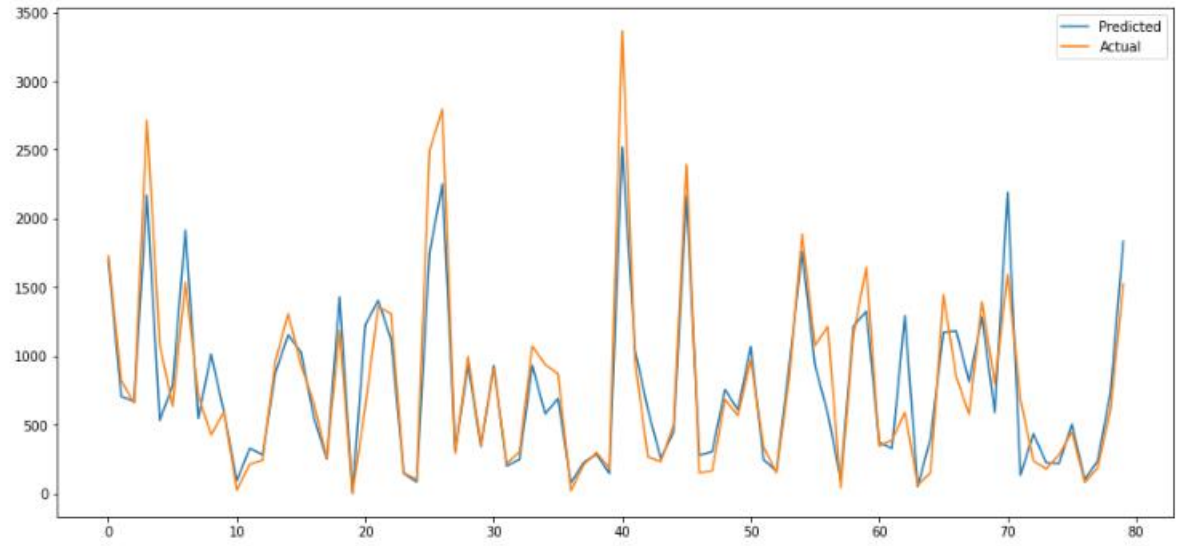
```
Best Parameters & Best Score
```

```
The best parameters found out to be :{'max_depth': 25, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_split': 20, 'n_estimators': 100}
where model best score is:  0.8619391165071102
```

```
Evalution Matrix
```

```
MAE : 144.97433432853114
MSE : 56589.28788304904
RMSE : 237.88503080910542
R2 : 0.8617078398947986
Adjusted R2 :  0.8586074084678721
```

ploting the graph of Actual and predicted only with 80 observation




This graph shows us fitness of this Algorithm to our model with minimum residual.

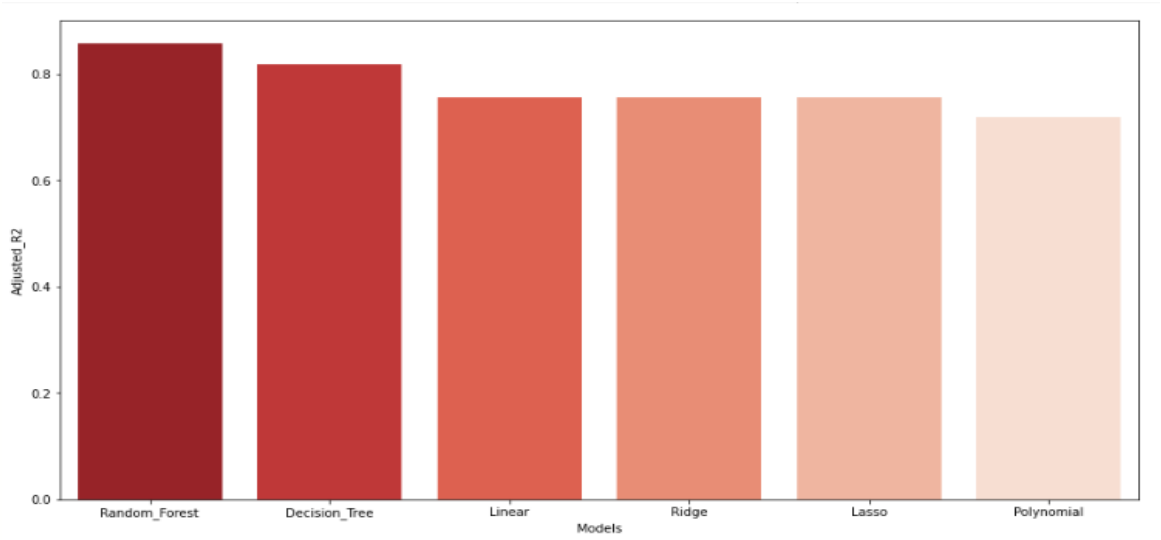
# Comparison of all Evaluation matrix with respect to models

Aim of creating this data-frame is to give single view of each algorithms performance on our model. Store every evaluation matrix into a dictionary and models into list then make DataFrame which look like this.

| | Models | Mean_Absolute_error | Mean_square_error | Root_Mean_square_error | Training_score | R2 | Adjusted_R2 |
|---|---|---|---|---|---|---|---|
| 0 | Random_Forest | 144.974334 | 56589.287883 | 237.885031 | 0.917794 | 0.861708 | 0.858607 |
| 1 | Decision_Tree | 166.414701 | 72640.256042 | 269.518563 | 0.880769 | 0.822483 | 0.818503 |
| 2 | Linear | 210.676856 | 97363.044734 | 312.030519 | 0.790227 | 0.762065 | 0.756731 |
| 3 | Ridge | 210.785082 | 97501.632876 | 312.252515 | 0.790222 | 0.761727 | 0.756385 |
| 4 | Lasso | 210.983783 | 97743.319291 | 312.639280 | 0.790209 | 0.761136 | 0.755781 |
| 5 | Polynomial | 135.612544 | 50477.578123 | 224.672157 | 0.921231 | 0.876644 | 0.719889 |

Here we get tabular presentation of each algorithms and its scores. We can easily make decision which algorithm work well on our model or give best fit result.

Then we also make bar graph of this dataframe to make accurate decision without any flaw.




Here we present model performance after applying algorithm in descending order. Each algorithm presenting by different colour. As we can see Random Forest algorithm perform very well on our bike_sharing_demand model. As compare to Decision Tree, Linear Regression and Lasso, Ridge and Polynomial regression is least performing algorithm than other.

# Conclusion

- We observed that Initially Bike sharing is not in demand but as time goes and many people got to know about it, they start using bicycle in daily purpose.
- At eight in the morning and six in the evening, account for the highest rental bike counts. The graph shows some upward tendency from 5 AM to 8 AM; it reaches its peak at 8 AM and then undergoes a downward trend. The demand then progressively increases until 6 o'clock, when it peaks, and then gradually decreases until midnight.
- We find pattern in peoples that when the weather is mild to sunny and even a little windy, we've seen that people prefer to hire bikes.
- Also demand of bicycle is more in weekdays than weekend probably most of the people going to office through bike.
- Bicycle rental rates have been found to be highest throughout the summer and autumn seasons and lowest during the winter.
- We discovered that the number of bike rentals is lowest on days with poor visibility and high humidity.
- After evaluating model with lots of algorithms we found Random Forest work well on model as compare to Decision tree, Linear Regression, Ridge, Lasso, and Polynomial Regression is least performing algorithm than other.