BIKE RENTAL COUNT PREDICTION

Project Report

Abstract

Project predicts bike rental count on daily basis based on season and environmental settings.

Gaurav Parmar

TABLE OF CONTENTS

1.INTRODUCTION

1.1 PROBLEM STATEMENT:

We need to predict bike rental count on daily basis based on the season and environmental settings.

1.2 DATA OVERVIEW:

The details of data attributes in the dataset are as follows -

instant: Record index
dteday: Date
season: Season (1: spring, 2: summer, 3: fall, 4: winter)
yr: Year (0: 2011, 1:2012)
mnth: Month (1 to 12)
hr: Hour (0 to 23)
holiday: weather day is holiday or not (extracted from Holiday Schedule)
weekday: Day of the week
workingday: If day is neither weekend nor holiday is 1, otherwise is 0.
weathersit: (extracted from Freemeteo)
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
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)
atemp: Normalized feeling temperature in Celsius. The values are derived via
(t-t_min)/(t_max-t_min), t_min=-16, t_max=+50 (only in hourly scale)
hum: Normalized humidity. The values are divided to 100 (max)
windspeed: Normalized wind speed. The values are divided to 67 (max)
casual: count of casual users
registered: count of registered users
cnt: count of total rental bikes including both casual and registered

We have 16 variables and 731 observations. In that 13 variables are independent and 3 dependent variables.

Let’s have a look at the data:

```
df.head()
```

	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

```
df.describe()
```

	instant	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	
count	731.000000	731.000000	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	0.627894	0.190486	
std	211.165812	1.110807	0.500342	3.451913	0.167155	2.004787	0.465233	0.544894	0.183051	0.162961	0.142429	0.077498	
min	1.000000	1.000000	0.000000	1.000000	0.000000	0.000000	0.000000	1.000000	0.059130	0.079070	0.000000	0.022392	
25%	183.500000	2.000000	0.000000	4.000000	0.000000	1.000000	0.000000	1.000000	0.337083	0.337842	0.520000	0.134950	
50%	366.000000	3.000000	1.000000	7.000000	0.000000	3.000000	1.000000	1.000000	0.498333	0.486733	0.626667	0.180975	
75%	548.500000	3.000000	1.000000	10.000000	0.000000	5.000000	1.000000	2.000000	0.655417	0.608602	0.730209	0.233214	1
max	731.000000	4.000000	1.000000	12.000000	1.000000	6.000000	1.000000	3.000000	0.861667	0.840896	0.972500	0.507463	3

Here casual, registered and count are our dependent variables

COUNT = CASUAL+REGISTERED

Remaining all are independent variables.

2. METHODOLOGY

2.1 DATA PRE-PROCESSING:

Data preprocessing is a data mining technique which transforms raw data into an understandable format. Data goes through series of steps during preprocessing. They are data cleaning, data visualization, data transformation, data reduction.

2.1.1 DATA EXPLORATION:

We need to check dimensions of the data, data types of the data, summary of the data. So that we can get good understandings about the data and also identify the target variable.

```
df.describe()
```

	instant	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	
count	731.000000	731.000000	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	0.627894	0.190486	
std	211.165812	1.110807	0.500342	3.451913	0.167155	2.004787	0.465233	0.544894	0.183051	0.162961	0.142429	0.077498	
min	1.000000	1.000000	0.000000	1.000000	0.000000	0.000000	0.000000	1.000000	0.059130	0.079070	0.000000	0.022392	
25%	183.500000	2.000000	0.000000	4.000000	0.000000	1.000000	0.000000	1.000000	0.337083	0.337842	0.520000	0.134950	
50%	366.000000	3.000000	1.000000	7.000000	0.000000	3.000000	1.000000	1.000000	0.498333	0.486733	0.626667	0.180975	
75%	548.500000	3.000000	1.000000	10.000000	0.000000	5.000000	1.000000	2.000000	0.655417	0.608602	0.730209	0.233214	1
max	731.000000	4.000000	1.000000	12.000000	1.000000	6.000000	1.000000	3.000000	0.861667	0.840896	0.972500	0.507463	3

```
df.columns
```

```
Index(['season', 'yr', 'mnth', 'holiday', 'weekday', 'workingday',
       'weathersit', 'temp', 'atemp', 'hum', 'windspeed', 'cnt'],
      dtype='object')
```

2.1.2 MISSING VALUE ANALYSIS:

Missing values are the data which is not present in the particular variable or observations. It may happen due to human error, or it may mark as an optional during the survey. If the data set contains missing values which is above 30%, either we need to drop the column or that particular observation. In our dataset we don't have any missing values but in real world problems there is always some missing values. We need to impute those missing values either it is classification or regression problems.

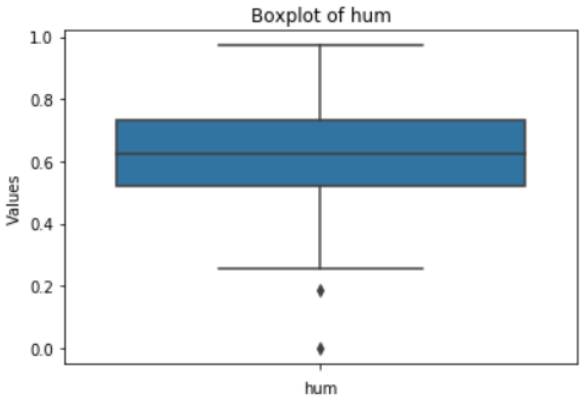
```
df.isnull().sum()
```

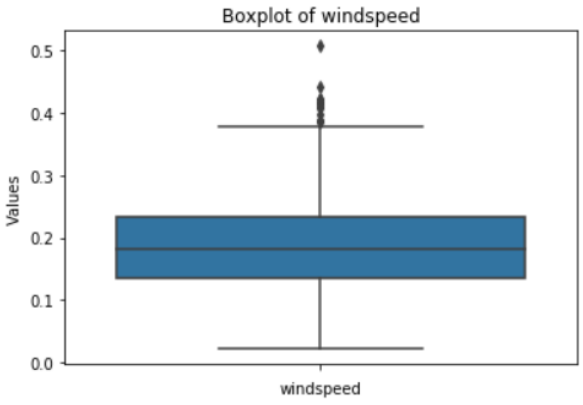
```
season        0
yr            0
mnth          0
holiday       0
weekday       0
workingday    0
weathersit    0
temp          0
atemp         0
hum           0
windspeed     0
cnt           0
dtype: int64
```

2.1.3 OUTLIER ANALYSIS:

Basically, outliers are the values which are lying far away from the remaining variables which may lead biased towards the higher value which results in the performance of our model. So, we need to treat the outliers.

Here outliers are detected using boxplot. We have inliers in humidity and outliers in windspeed other than that we don't have any outliers. So, in our case we imputed the outliers with median value of the column. So that we save the information and increase the performance of our model.



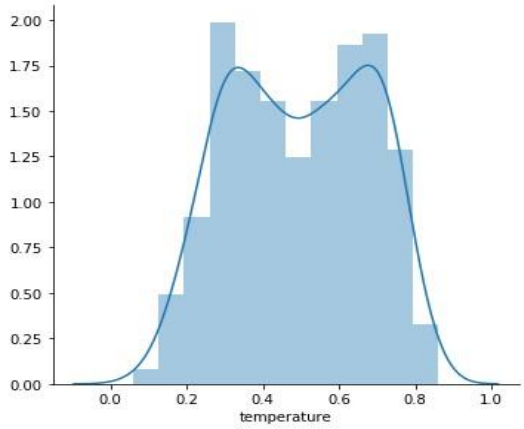


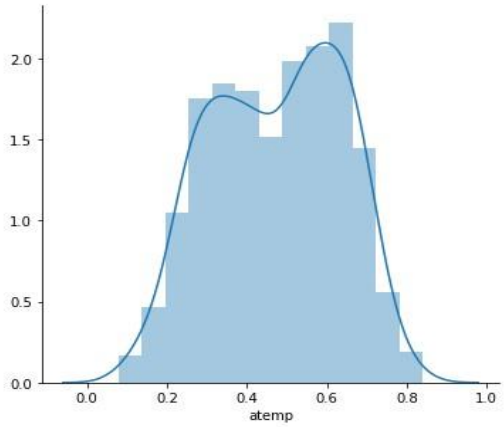
2.1.4 DATA VISUALIZATION:

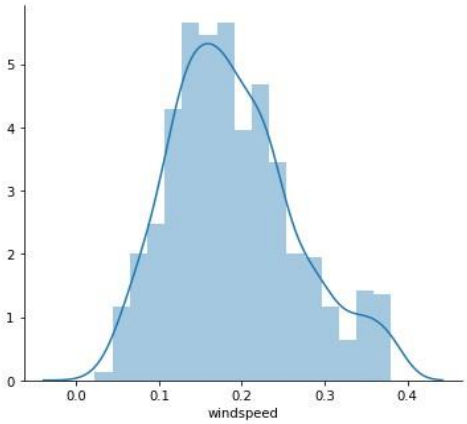
Data visualization is the graphical representation of information and data. By using visual elements like charts, graphs, and maps, data visualization tools provide an accessible way to see and understand trends, outliers, and patterns in data.

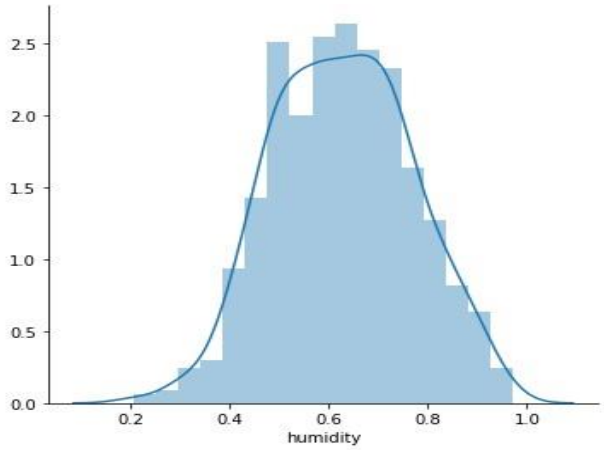
2.1.4.a: DISTRIBUTION OF NUMERIC VARIABLE:

Distribution plot is used basically for univariant set of observations and visualizes it through a histogram.





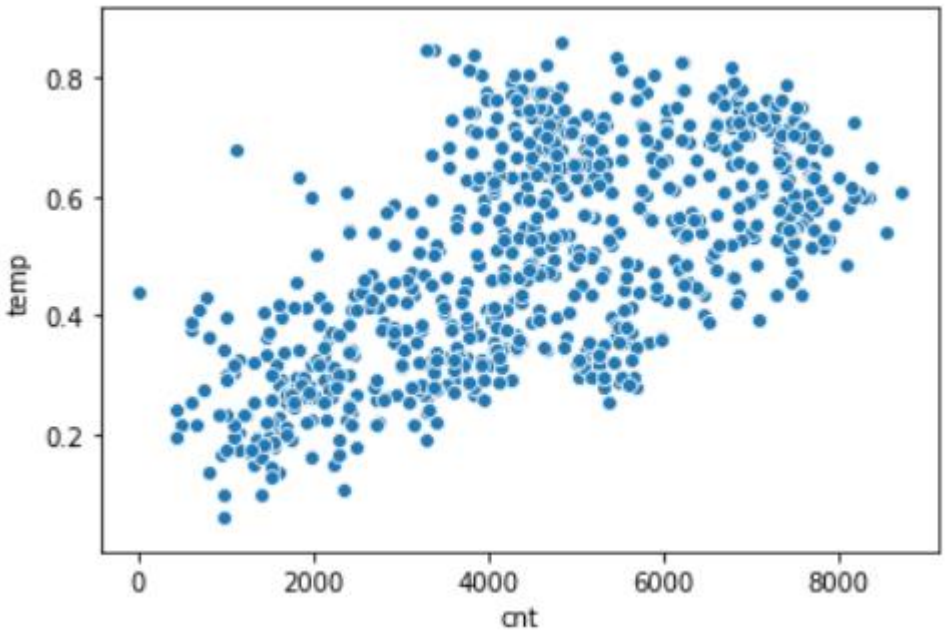




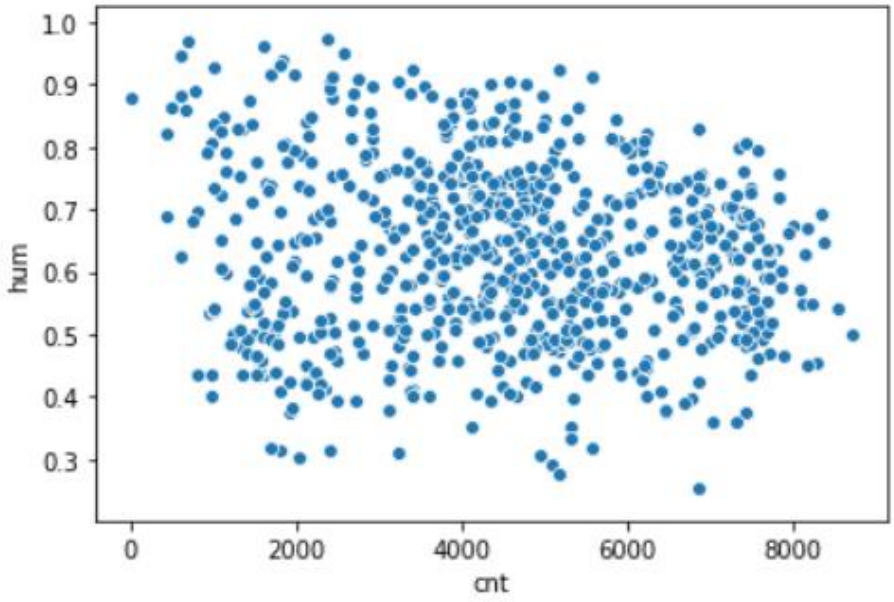
2.1.4.b: DISTRIBUTION OF CONTINUOUS VARIABLES WITH RESPECT TO TARGET VARIABLE:

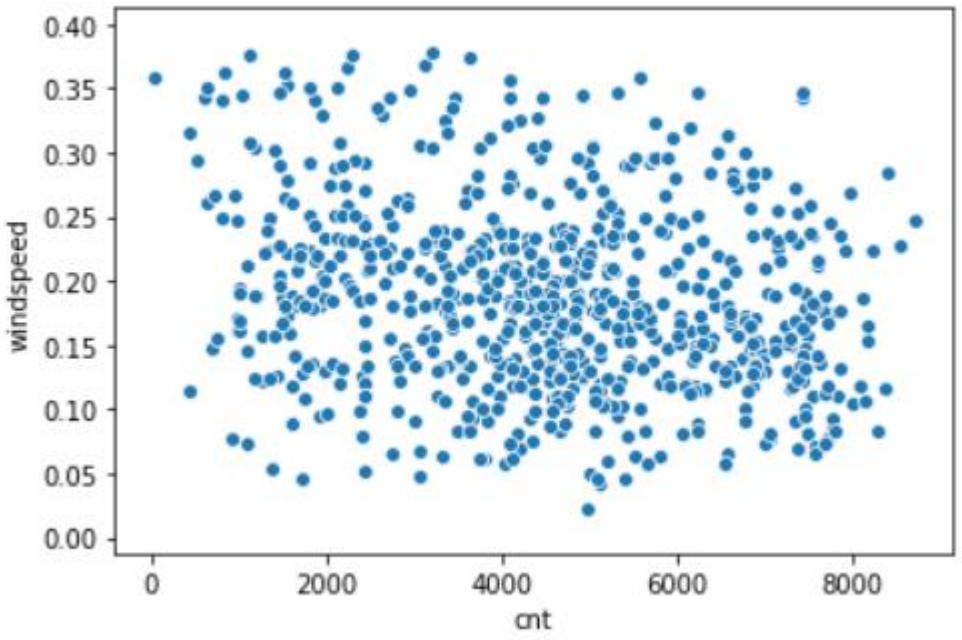
Here we used scatterplot for our visualization. Scatter plots are used to plot data points on a horizontal and a vertical axis in the attempt to show how much one variable is affected by another. Each row in the data table is represented by a marker whose position depends on its values in the columns set on the X and Y axes.

From the below plot, we can say that as temperature increases our bike rental count also increases.



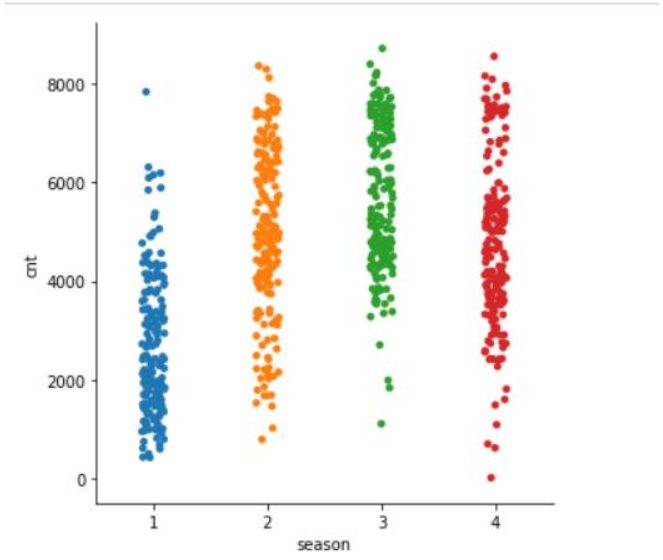
Second plot is humidity with respect to the count. Below plot shows that there is no significant relation between humidity and count.





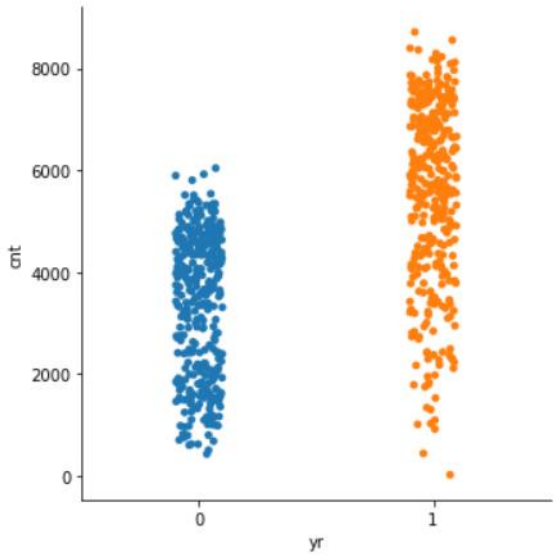
From the above plot we say that bike Rental count is not affected by windspeed.

2.1.4c: IMPACT OF THE CATEGORICAL VARIABLE WITH RESPECT TO TARGET VARIABLE:



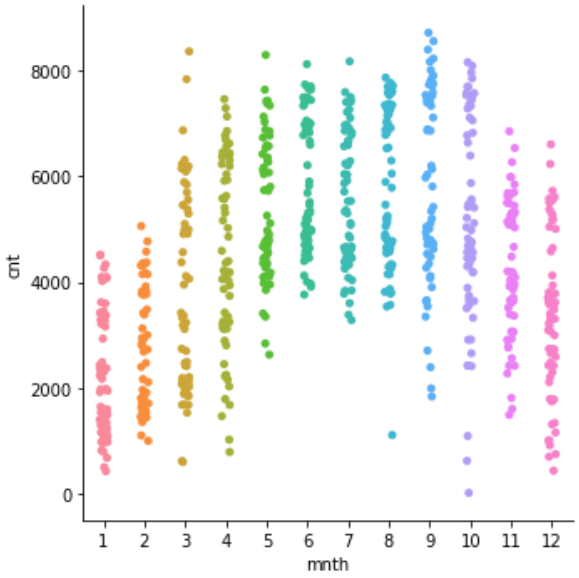
COUNT VS SEASON

Bike rental is higher in season 3 which is fall and low in season 1 which is spring.



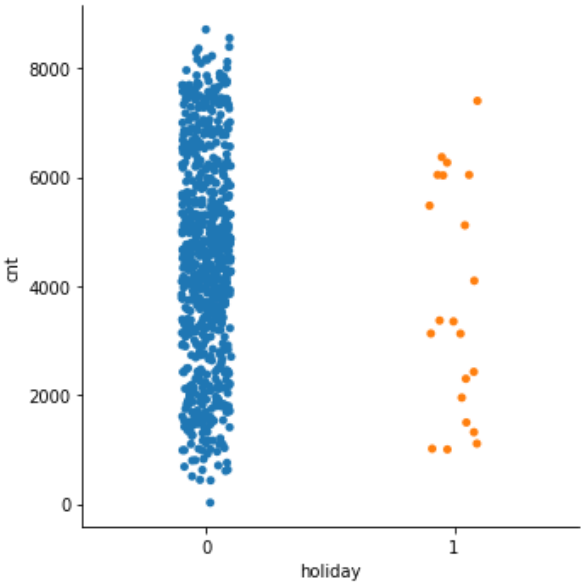
COUNT VS YEAR

Bike rental is higher in the year 1 which is 2012.



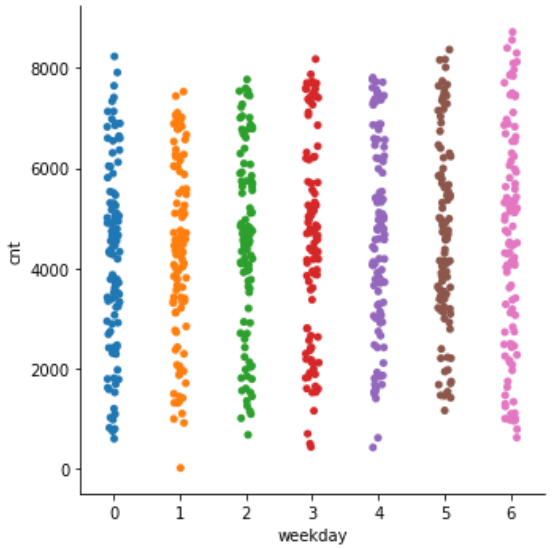
COUNT VS MONTH

Bike rental is higher in the month of 9 which is September and low in 1 which is January.



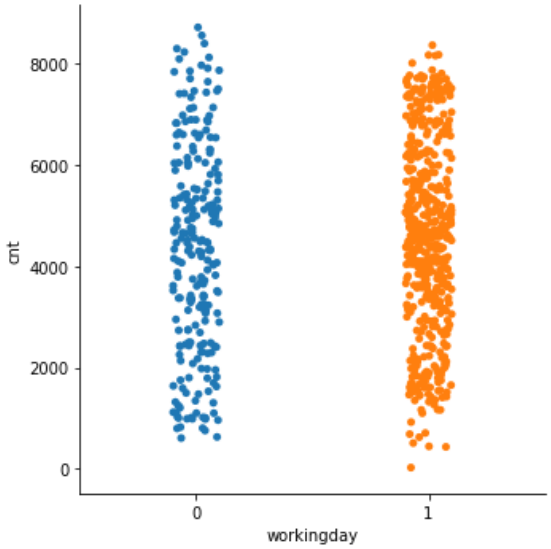
COUNT VS HOLIDAY

Bike rental count is higher in 0 which is holiday and low in workingday.



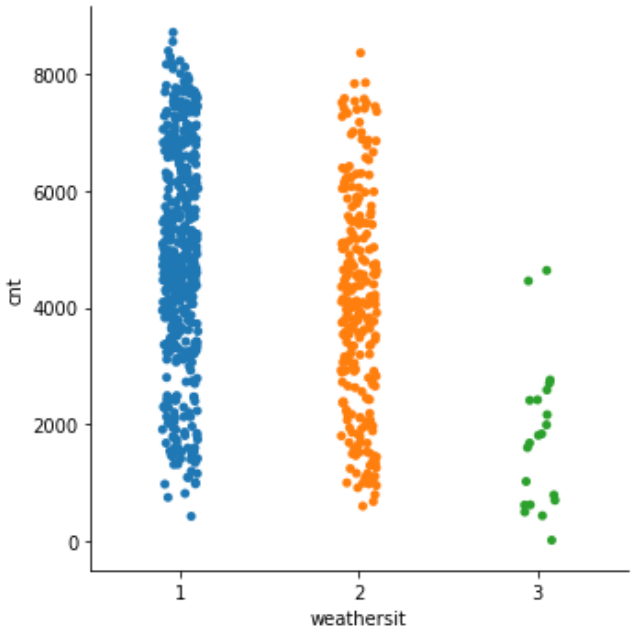
COUNT VS WEEKDAY

Bike rental count is high in 6 which is Saturday and low in 1 which is Monday.



COUNT VS WORKING DAY

Bike rental count is low in 1 which is working day and high in 0 which is holiday.



COUNT VS WEATHER

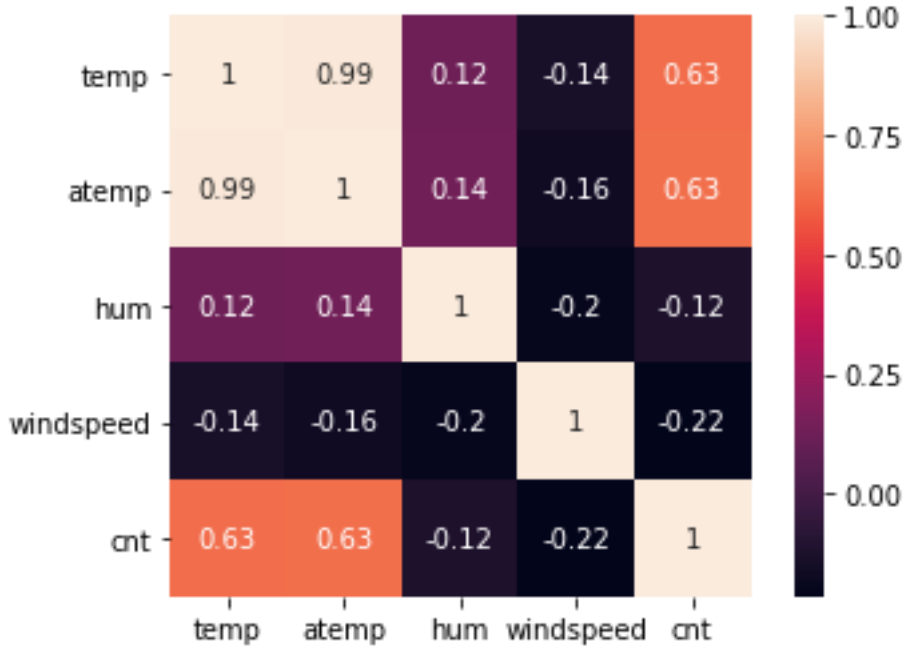
Bike rental count is high in 1 which is in 1 which clear, few clouds, partly cloudy and there is no bikes rental in 4.

2.1.5 FEATURE SELECTION:

We can use correlation analysis for numerical variables and Analysis of Variance for categorical variables. It shows correlation between the two variables. So that if two variables carrying same information can be removed.

2.1.5.a: CORRELATION MATRIX AND PLOT:

	temp	atemp	hum	windspeed	cnt
temp	1.0000000	0.9917016	0.1237322	-0.1392442	0.6274940
atemp	0.9917016	1.0000000	0.1373204	-0.1644920	0.6310657
hum	0.1237322	0.1373204	1.0000000	-0.2007293	-0.1214257
windspeed	-0.1392442	-0.1644920	-0.2007293	1.0000000	-0.2155710
cnt	0.6274940	0.6310657	-0.1214257	-0.2155710	1.0000000



From the above plot, we say that temperature and atemp variables are carrying same information. So, we need to remove atemp variable.

2.1.5.b ANALYSIS OF VARIANCE:

```
                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
```

From the above diagram, holiday, weekday, and working day have p-value which is higher than 0.05. Hence, we need to drop these variables.

2.1.5.c DIMENSION REDUCTION:

After the feature selection, we have only these 8 variables. They are mentioned in the below diagram.

```
df.drop(['atemp'], axis=1, inplace=True)
```

```
df.drop(['holiday','weekday','workingday'], axis=1, inplace=True)
```

```
df.shape
```

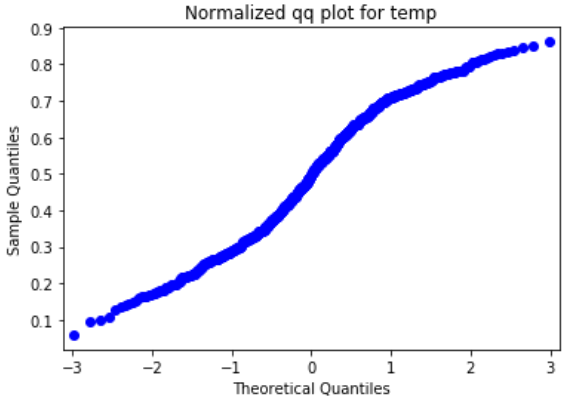
```
(731, 8)
```

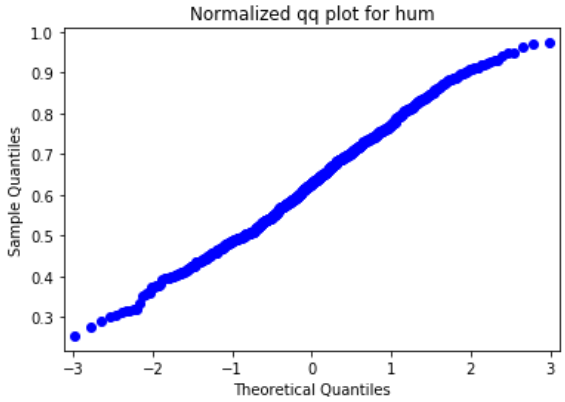
```
df.columns
```

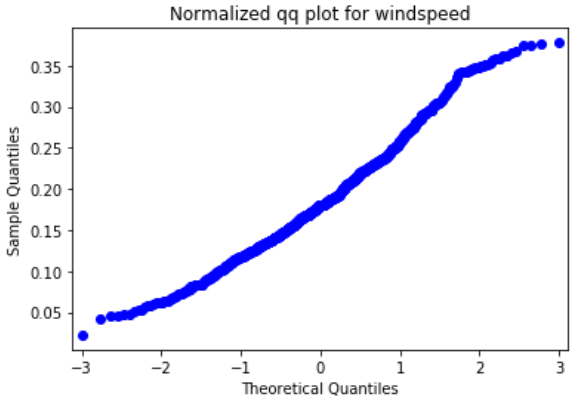
```
Index(['season', 'yr', 'mnth', 'weathersit', 'temp', 'hum', 'windspeed',
       'cnt'],
      dtype='object')
```

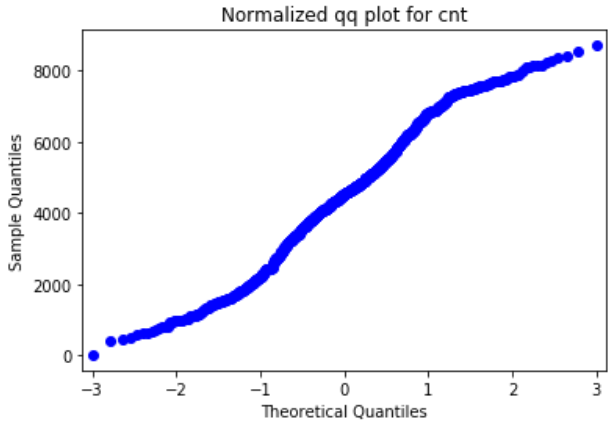
2.1.6 FEATURE SCALING:

In our dataset, all our continuous variables are already normalized. So, we don't need to need any scaling methods to scale the data. Though we can use qqplot, summary, distribution of the data to see the normality.









Summary of the data after feature selection and dimension reduction.

```
Bike_Data.describe()
```

	season	year	month	weather	temperature	humidity	windspeed	count
count	731.000000	731.000000	731.000000	731.000000	731.000000	731.000000	731.000000	731.000000
mean	2.496580	0.500684	6.519836	1.395349	0.495385	0.627894	0.190486	4504.348837
std	1.110807	0.500342	3.451913	0.544894	0.183051	0.142429	0.077498	1937.211452
min	1.000000	0.000000	1.000000	1.000000	0.059130	0.000000	0.022392	22.000000
25%	2.000000	0.000000	4.000000	1.000000	0.337083	0.520000	0.134950	3152.000000
50%	3.000000	1.000000	7.000000	1.000000	0.498333	0.626667	0.180975	4548.000000
75%	3.000000	1.000000	10.000000	2.000000	0.655417	0.730209	0.233214	5956.000000
max	4.000000	1.000000	12.000000	3.000000	0.861667	0.972500	0.507463	8714.000000

2.2.4 MODEL DEVELOPMENT:

Next, we need to split the data into train and test data and build a model using train data to predict the output using test data. Different models to be built and the model which gives more accurate values must be selected.

2.2.1 LINEAR REGRESSION:

Linear regression is a linear approach to modeling the relationship between a scalar response (or dependent variable) and one or more explanatory variables (or independent variables). The case of one explanatory variable is called simple linear regression.

Multiple linear regression is the most common form of linear regression analysis. Multiple regression is an extension of simple linear regression. It is used as a predictive analysis, when we want to predict the value of a variable based on the value of two or more other variables. The variable we want to predict is called the dependent variable (or sometimes, the outcome, target or criterion variable).

2.2.4 DECISION TREE:

Decision tree builds regression or classification models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes. The topmost decision node in a tree which corresponds to the best predictor called root node. Decision trees can handle both categorical and numerical data.

2.2.4 RANDOM FOREST:

A Random Forest is an ensemble technique capable of performing both regression and classification tasks with the use of multiple decision trees, which involves training each decision tree on a different data sample where sampling is done with replacement. The basic idea behind this is to combine multiple decision trees in determining the final output rather than relying on individual decision trees. 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.

2.2.4 GRADIENT BOOSTING:

Gradient boosting is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees. It builds the model in a stage-wise fashion like other boosting methods do, and it generalizes them by allowing optimization of an arbitrary differentiable loss function.

2.3 HYPERPARAMETER TUNING

Model hyperparameters are set by the data scientist ahead of training and control implementation aspects of the model. The weights learned during training of a linear regression model are parameters while the number of trees in a random forest is a model hyperparameter because this is set by the data scientist. Hyperparameters can be thought of as model settings. These settings need to be tuned for each problem because the best model hyperparameters for one particular dataset will not be the best across all datasets. The process of hyperparameter tuning (also called hyperparameter optimization) means finding the combination of hyperparameter values for a machine learning model that performs the best - as measured on a validation dataset - for a problem.
Here we have used two hyper parameters tuning techniques

2.3.1 TUNING PARAMETERS:

We will explore two different methods for optimizing hyperparameters:

- Grid Search
- Random Search

2.3.1.A. 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 optimized parameters without any aliasing.

2.3.1.B. 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. 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 will be selected.

3. MODEL EVALUATION

3.1 EVALUATION METRICS:

In regression problems, we have three important metrics. They are

- MAPE (Mean Absolute Percentage Error)
- R-SQUARED
- RMSE (Root Mean Square Error)

3.1.1 MAPE (Mean Absolute Percentage Error)

MAPE is a measure of prediction accuracy of a forecasting method. It measures accuracy in terms of percentage. Lower value of MAPE indicates better fit.

3.1.2 R-SQUARED

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. Higher values of R-square indicate better fit.

3.1.3 RMSE (Root Mean Square Error)

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. 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.

Below table shows the model results before applying hyper tuning:

Model Name	RMSE	R-Squared
Linear Regression	857.9	0.82
Decision Tree	814.12	0.83
Random Forest	686.79	0.88
Gradient Boosting	591.74	0.91

Below table shows results post using hyper parameter tuning techniques:

Model Name	Parameter	RMSE	R-Squared
Random Search CV	Random Forest	673.13	0.88
	Gradient Boosting	617.27	0.90
Grid Search CV	Random Forest	697.87	0.87
	Gradient Boosting	602.25	0.90

3.2 MODEL SELECTION:

On the basis RMSE and R Squared results a good model should have least RMSE and max R Squared value. So, from above tables we can see:

- From the observation of all RMSE Value and R-Squared Value we have concluded that, Both the models- Gradient Boosting and Random Forest perform comparatively well while comparing their RMSE and R-Squared value.
- After this, I chose Random Forest CV and Grid Search CV to apply cross validation technique and see changes brought about by that.
- After applying tunings Gradient Boosting model shows best results compared to Random forest.
- So finally, we can say that Gradient boosting model is the best method to make prediction for this project with highest explained variance of the target variables and lowest error chances.