In [1]:
```
%%javascript
IPython.OutputArea.prototype._should_scroll = function(lines) {
    return false;
}
```

PROJECT :

BIKE SHARING DEMANDS

To build a **multiple linear regression model** for the prediction of **demand for shared bikes**.

PROBLEM STATEMENT :

A **bike-sharing system** is a service in which bikes are made available for shared use to individuals on a **short term basis** for a price or free. Many bike share systems allow people to borrow a bike from a **"dock"** which is usually computer-controlled wherein the user enters the payment information, and the system unlocks it. This bike can then be returned to another dock belonging to the same system.

A US bike-sharing provider **BoomBikes** has recently suffered considerable dips in their revenues due to the ongoing Corona pandemic. The company is finding it very difficult to sustain in the current market scenario. So, it has decided to come up with a **mindful business plan** to be able to accelerate its revenue as soon as the ongoing lockdown comes to an end, and the economy restores to a healthy state.

In such an attempt, BoomBikes aspires to **understand the demand for shared bikes** among the people after this ongoing quarantine situation ends across the nation due to Covid-19. They have planned this to prepare themselves to cater to the people's needs and stand out from other service providers and make huge profits.

For this purpose, the company wants to understand the **factors affecting the demand for these shared bikes** in the American market and they should know:

> - **Which variables are significant in predicting the demand for shared bikes?**
> - **How well those variables describe the bike demands?**

Based on various meteorological surveys and people's styles, the service provider firm has gathered a large dataset on daily bike demands across the American market based on some factors.

BUSINESS GOAL :

We should model the **demand for shared bikes with the available independent variables**. It will be used by the management to understand how exactly the demands vary with different features. They can accordingly manipulate the business strategy to **meet the demand** levels and meet the customer's expectations. Further, the model will be a good way for management to understand the demand dynamics of a new market.

Major parts of the project :

> 1. Data understanding
> 2. Data cleaning
>
> > a. Fixing Rows and Columns
> > b. Check for missing values
> > c. Standardizing values
> > d. Check for outliers
> > e. Derived variables
>
> 3. Exploratory Data Analysis (EDA)
> 4. Model Building
>
> > a. Data preparation
> > b. Splitting the Data into Training and Testing Sets
> > c. Rescaling the Features using MinMax Scaler
> > d. Building Multiple linear regression model using RFE (Recursive Feature Elimination)
> > e. Residual Analysis of the train dataset
> > f. Making Predictions Using the Final Model

> 5. Model Evaluation
> 6. Interpretation of results

Import required libraries

In [2]:
```
# Importing required libraries for the project

import pandas as pd
import numpy as np
import datetime as dt
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
import statsmodels.api as sm
import sklearn
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error
from sklearn.feature_selection import RFE
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import MinMaxScaler
from statsmodels.stats.outliers_influence import variance_inflation_factor
from sklearn.preprocessing import OneHotEncoder
```

1. DATA UNDERSTANDING

Read the "Bike Sharing" dataset

In [3]:
```
# Reading the "Bike Sharing" dataset

BikeSharing_df = pd.read_csv("day.csv")
```

In [4]:
```
# First 5 records of the dataset

BikeSharing_df.head()
```

Out[4]:

	instant	dteday	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	casual	registered	cnt
0	1	01-01-2018	1	0	1	0	6	0	2	14.110847	18.18125	80.5833	10.749882	331	654	985
1	2	02-01-2018	1	0	1	0	0	0	2	14.902598	17.68695	69.6087	16.652113	131	670	801
2	3	03-01-2018	1	0	1	0	1	1	1	8.050924	9.47025	43.7273	16.636703	120	1229	1349
3	4	04-01-2018	1	0	1	0	2	1	1	8.200000	10.60610	59.0435	10.739832	108	1454	1562
4	5	05-01-2018	1	0	1	0	3	1	1	9.305237	11.46350	43.6957	12.522300	82	1518	1600

Shape of the dataset(rows,columns)

In [5]:

```
BikeSharing_df.shape
```

Out[5]:

```
(730, 16)
```

Column names and their intended meanings

- **instant**: record index
- **dteday** : date
- **season** : season (1:Spring, 2:Summer, 3:Fall, 4:Winter)
- **yr** : year (0: 2018, 1:2019)
- **mnth** : month (1 to 12)
- **holiday** : whether day is a holiday or not (weekend holiday excluded)
- **weekday** : day of the week (0 - Sunday, 1 - Monday, 2 - Tuesday, 3 - Wednesday, 4 - Thursday, 5 - Friday, 6 - Saturday)
- **workingday** : if day is neither weekend nor holiday is 1, otherwise is 0.
- **weathersit** :
  - **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** : normal temperature in Celsius
- **atemp**: Ambient temperature is the air temperature of any object or environment where equipment is stored (in Celsius)
- **hum**: humidity
- **windspeed**: wind speed
- **casual**: count of casual users
- **registered**: count of registered users
- **cnt**: count of total rental bikes including both casual and registered

Basic information of the dataset

In [6]:
```
BikeSharing_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 730 entries, 0 to 729
Data columns (total 16 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   instant     730 non-null    int64  
 1   dteday      730 non-null    object 
 2   season      730 non-null    int64  
 3   yr          730 non-null    int64  
 4   mnth        730 non-null    int64  
 5   holiday     730 non-null    int64  
 6   weekday     730 non-null    int64  
 7   workingday  730 non-null    int64  
 8   weathersit  730 non-null    int64  
 9   temp        730 non-null    float64
 10  atemp       730 non-null    float64
 11  hum         730 non-null    float64
 12  windspeed   730 non-null    float64
 13  casual      730 non-null    int64  
 14  registered  730 non-null    int64  
 15  cnt         730 non-null    int64  
dtypes: float64(4), int64(11), object(1)
memory usage: 91.4+ KB
```

Inferences :

- There are **no missing values** in the dataset.
- Except the variable - **'dteday'**, all other variables are **numeric**.

Description of the dataset

In [7]:

```
BikeSharing_df.describe()
```

Out[7]:

	instant	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	
count	730.000000	730.000000	730.000000	730.000000	730.000000	730.000000	730.000000	730.000000	730.000000	730.000000	730.000000	730.000000	
mean	365.500000	2.498630	0.500000	6.526027	0.028767	2.997260	0.683562	1.394521	20.319259	23.726322	62.765175	12.763620	
std	210.877136	1.110184	0.500343	3.450215	0.167266	2.006161	0.465405	0.544807	7.506729	8.150308	14.237589	5.195841	
min	1.000000	1.000000	0.000000	1.000000	0.000000	0.000000	0.000000	1.000000	2.424346	3.953480	0.000000	1.500244	
25%	183.250000	2.000000	0.000000	4.000000	0.000000	1.000000	0.000000	1.000000	13.811885	16.889713	52.000000	9.041650	
50%	365.500000	3.000000	0.500000	7.000000	0.000000	3.000000	1.000000	1.000000	20.465826	24.368225	62.625000	12.125325	
75%	547.750000	3.000000	1.000000	10.000000	0.000000	5.000000	1.000000	2.000000	26.880615	30.445775	72.989575	15.625589	1
max	730.000000	4.000000	1.000000	12.000000	1.000000	6.000000	1.000000	3.000000	35.328347	42.044800	97.250000	34.000021	3

2. DATA CLEANING

a. Fixing Rows and Columns

In [8]:

```
# The variable - 'instant' is just a row index and is of no significance for the analysis
# So the column 'instant' is dropped

BikeSharing_df = BikeSharing_df.drop("instant", axis = 1)
```

- The column **'casual'** describes the count of casual users.
- The column **'registered'** describes the count of registered users.
- **'cnt'** is the target variable which is the count of total rental bikes including both casual and registered.

> Obviously **'casual'** and **'registered'** variables lead to **data leakage** in the model.
> This may distort the analysis of other original predictors of the target variable **'cnt'**.
> Hence **drop** the variables **'casual'** and **'registered'**.

In [9]:

```
# Drop the variables 'casual' and 'registered'

BikeSharing_df = BikeSharing_df.drop(['casual','registered'], axis = 1)
```

In [10]:

```
# Shape of the dataframe after dropping insignificant columns

BikeSharing_df.shape
```

Out[10]:

```
(730, 13)
```

The values in the column **'weathersit'** implies the following labels :

> **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

Therefore map the values with meaningful labels associated with them.

In [11]:

```
# Mapping the associated labels for the values of 'weathersit' column

BikeSharing_df['weathersit'] = BikeSharing_df['weathersit'].map({1 : "Clear", 2 : "Misty", 3 : "Light Snow or Rain",
                                                                  4 : "Heavy Snow or Rain"})
```

The values in the column **'season'** implies the following labels :

> **1** - Spring
> **2** - Summer
> **3** - Fall
> **4** - Winter

Therefore map the values with meaningful labels associated with them.

In [12]:
```
# Mapping the associated labels for the values of 'season' column

BikeSharing_df['season'] = BikeSharing_df['season'].map({1 : "Spring", 2 : "Summer", 3 : "Fall", 4 : "Winter"})
```

Map the days of the week in the **'weekday'** column :

> **0** - Sunday
> **1** - Monday
> **2** - Tuesday
> **3** - Wednesday
> **4** - Thursday
> **5** - Friday
> **6** - Saturday

In [13]:
```
# Mapping the days of the week in the column 'weekday'

BikeSharing_df['weekday'] = BikeSharing_df['weekday'].map({0 : "Sunday", 1 : "Monday", 2 : "Tuesday", 3 : "Wednesday",
                                                            4 : "Thursday", 5 : "Friday", 6 : "Saturday"})
```

Map the corresponding months to the numeric values of **'mnth'** column :

> **1** - January
> **2** - February
> **3** - March
> **4** - April
> **5** - May
> **6** - June
> **7** - July
> **8** - August
> **9** - September
> **10** - October

> **11** - November
> **12** - December

In [14]:
```
# Mapping the corresponding months in 'mnth' column

BikeSharing_df['mnth'] = BikeSharing_df['mnth'].map({1 : "January", 2 : "February", 3 : "March", 4 : "April", 5 : "May",
                                                     6 : "June", 7 : "July", 8 : "August", 9 : "September",
                                                     10 : "October", 11 : "November", 12 : "December"})
```

In [15]:
```
# First 5 records of the dataframe after mapping associated labels

BikeSharing_df.head()
```

Out[15]:

	dteday	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	cnt
0	01-01-2018	Spring	0	January	0	Saturday	0	Misty	14.110847	18.18125	80.5833	10.749882	985
1	02-01-2018	Spring	0	January	0	Sunday	0	Misty	14.902598	17.68695	69.6087	16.652113	801
2	03-01-2018	Spring	0	January	0	Monday	1	Clear	8.050924	9.47025	43.7273	16.636703	1349
3	04-01-2018	Spring	0	January	0	Tuesday	1	Clear	8.200000	10.60610	59.0435	10.739832	1562
4	05-01-2018	Spring	0	January	0	Wednesday	1	Clear	9.305237	11.46350	43.6957	12.522300	1600

Therefore, some of the **NUMERICAL** variables are converted to meaningful **CATEGORICAL** variables.

b. Recheck for missing values in the dataset

In [16]:
```
BikeSharing_df.isnull().sum().sum()
```

Out[16]:
```
0
```

c. Standardizing values

In [17]:
```
# Convert the type of 'dteday' to 'datetime' object
```

```
BikeSharing_df['dteday'] = pd.to_datetime(BikeSharing_df['dteday'])
```

In [18]:
```
# Pick the columns of 'float' type to round off

roundOff_columns = ['temp','atemp','hum','windspeed']
```

In [19]:
```
# Round off float values to 2 decimal places

BikeSharing_df[roundOff_columns] = BikeSharing_df[roundOff_columns].apply(lambda x : round(x,2))
```

In [20]:
```
# First 5 records of the dataframe after standardizing values

BikeSharing_df.head()
```

Out[20]:

	dteday	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	cnt
0	2018-01-01	Spring	0	January	0	Saturday	0	Misty	14.11	18.18	80.58	10.75	985
1	2018-02-01	Spring	0	January	0	Sunday	0	Misty	14.90	17.69	69.61	16.65	801
2	2018-03-01	Spring	0	January	0	Monday	1	Clear	8.05	9.47	43.73	16.64	1349
3	2018-04-01	Spring	0	January	0	Tuesday	1	Clear	8.20	10.61	59.04	10.74	1562
4	2018-05-01	Spring	0	January	0	Wednesday	1	Clear	9.31	11.46	43.70	12.52	1600

d. Check for outliers

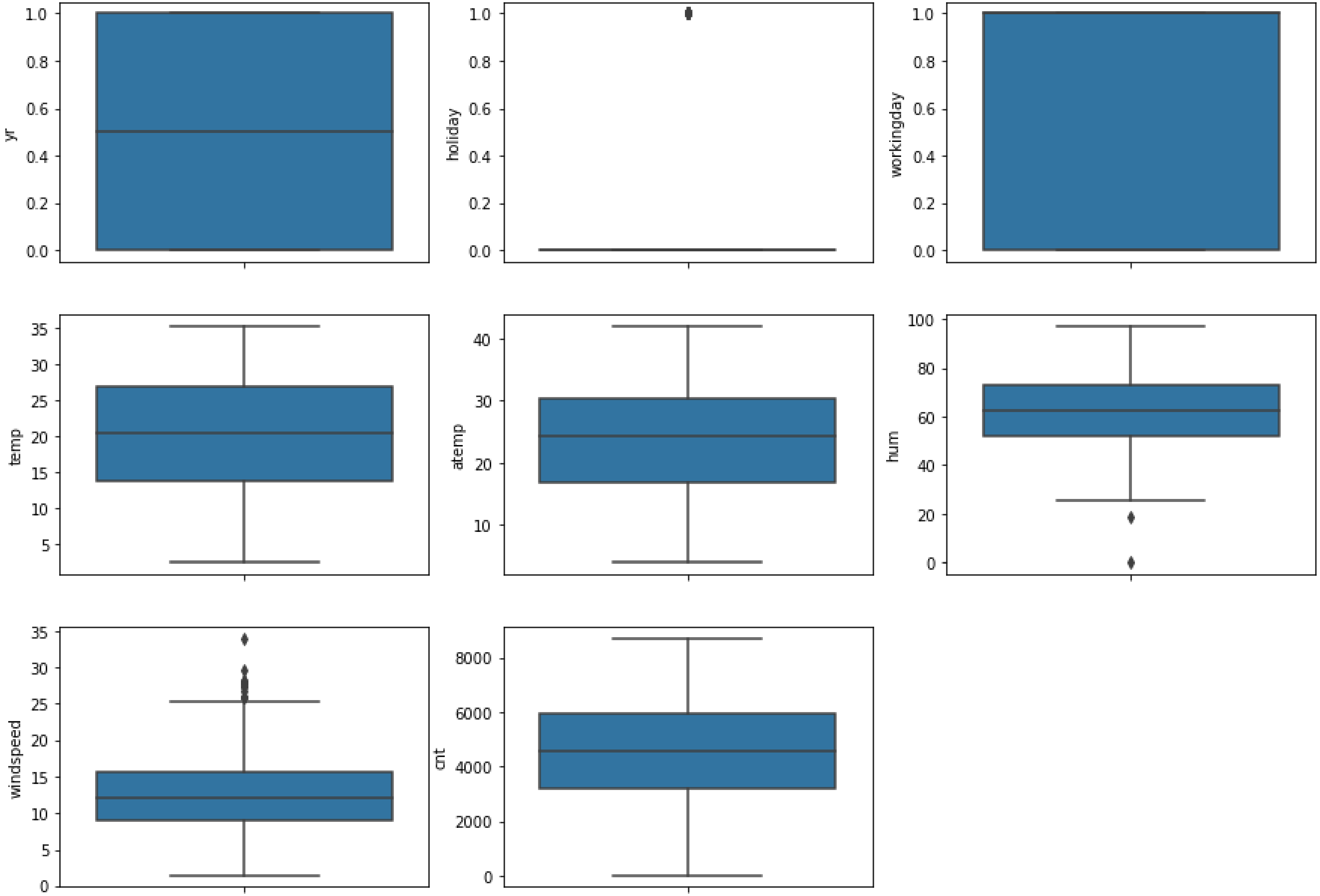
In [21]:
```
# Numeric columns are stored in the variable 'numeric_columns'

numeric_columns = BikeSharing_df[['yr','holiday','workingday','temp','atemp','hum','windspeed','cnt']]
```

In [22]:
```
# Boxplots are plotted for numeric columns to check for outliers

plt.figure(figsize=(15,30))
for i in range(len(numeric_columns.columns)):
    plt.subplot(8, 3, i + 1)
```

```
    sns.boxplot(y = numeric_columns.columns[i], data = BikeSharing_df)
plt.show()
```



Inference : The columns 'windspeed' and 'hum' seem to have outliers

In [23]:
```
# Description of 'windspeed' variable to check for outliers

BikeSharing_df['windspeed'].describe()
```

Out[23]:
```
count    730.000000
mean      12.763699
std        5.195640
min        1.500000
25%        9.040000
50%       12.130000
75%       15.627500
max       34.000000
Name: windspeed, dtype: float64
```

Inference :

- At **75th percentile**, the value is **15.6275** but suddenly the value peaks to **34.0000** at **100th percentile**.
- This is an indication of an **outlier** and hence it has to be removed.

In [24]:
```
# Check at which percentile the value distorts more, so that v can remove the outliers beyond that percentile.

print("At 80%, windspeed is ",BikeSharing_df['windspeed'].quantile(0.80))
print("At 90%, windspeed is ",BikeSharing_df['windspeed'].quantile(0.90))
print("At 95%, windspeed is ",BikeSharing_df['windspeed'].quantile(0.95))
print("At 99%, windspeed is ",BikeSharing_df['windspeed'].quantile(0.99))
print("At 100%,windspeed is ",BikeSharing_df['windspeed'].quantile(1.00))
```

```
At 80%, windspeed is  16.642
At 90%, windspeed is  19.83
At 95%, windspeed is  23.0
At 99%, windspeed is  27.382300000000004
At 100%,windspeed is  34.0
```

In [25]:
```
# The value distorts more at 100%. So let us drop the records where 'windspeed' > 28.

BikeSharing_df = BikeSharing_df[~ ( BikeSharing_df['windspeed'] > 28 )]
```

In [26]:
```
# Description of 'hum' variable to check for outliers

BikeSharing_df['hum'].describe()
```

Out[26]:
```
count    726.000000
mean      62.864697
std       14.154823
min        0.000000
25%       52.052500
50%       62.710000
75%       73.030000
max       97.250000
Name: hum, dtype: float64
```

Inference :

- At **0th percentile**, the value is **0.00000** but suddenly the value increases to **52.0525** (which is more than 50% of the total interval) at **25th percentile**.
- This is an indication of an **outlier** and hence it has to be removed.

In [27]:
```
# Check at which percentile the value distorts more, so that v can remove the outliers less than that percentile.

print("At 10%,humidity is ",BikeSharing_df['hum'].quantile(0.10))
print("At 5%, humidity is ",BikeSharing_df['hum'].quantile(0.05))
print("At 3%, humidity is ",BikeSharing_df['hum'].quantile(0.03))
print("At 2%, humidity is ",BikeSharing_df['hum'].quantile(0.02))
print("At 1% ,humidity is ",BikeSharing_df['hum'].quantile(0.01))
print("At 0% ,humidity is ",BikeSharing_df['hum'].quantile(0.00))
```

```
At 10%,humidity is  45.25
At 5%, humidity is  40.8725
At 3%, humidity is  38.8125
At 2%, humidity is  36.019999999999996
At 1% ,humidity is  31.4225
At 0% ,humidity is  0.0
```

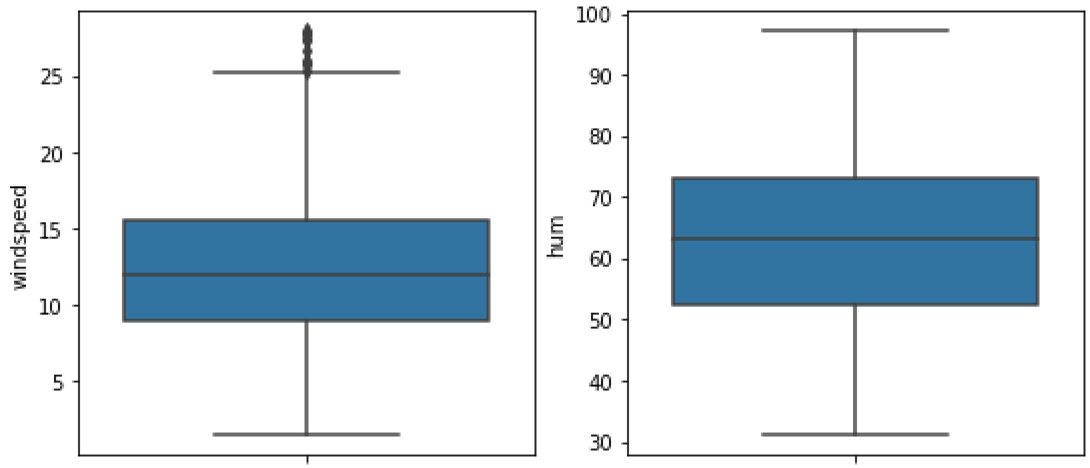
In [28]:
```
# The value distorts more between 0% and 1%. So let us drop the records where 'hum' < 31.

BikeSharing_df = BikeSharing_df[~ (BikeSharing_df['hum'] < 31)]
```

In [29]:
```
# After removing outliers, recheck the boxplots of 'windspeed' and 'hum'

plt.figure(figsize=(9,9))
plt.subplot(2, 2, 1)
```

```
sns.boxplot(y = 'windspeed', data = BikeSharing_df)
plt.subplot(2, 2, 2)
sns.boxplot(y = 'hum', data = BikeSharing_df)
plt.show()
```



Inference - From the boxplots of 'windspeed' and 'hum', it is clear that the outliers are removed

In [30]:

```
# Shape of the dataframe after removing outliers

BikeSharing_df.shape
```

Out[30]:

```
(720, 13)
```

In [31]:

```
# First 5 records of the dataframe after removing outliers

BikeSharing_df.head()
```

Out[31]:

	dteday	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	cnt
0	2018-01-01	Spring	0	January	0	Saturday	0	Misty	14.11	18.18	80.58	10.75	985
1	2018-02-01	Spring	0	January	0	Sunday	0	Misty	14.90	17.69	69.61	16.65	801
2	2018-03-01	Spring	0	January	0	Monday	1	Clear	8.05	9.47	43.73	16.64	1349

	dteday	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	cnt
3	2018-04-01	Spring	0	January	0	Tuesday	1	Clear	8.20	10.61	59.04	10.74	1562
4	2018-05-01	Spring	0	January	0	Wednesday	1	Clear	9.31	11.46	43.70	12.52	1600

e. Derived Variables

In [32]:

```
# Extract 'day' from 'dteday'

BikeSharing_df['day'] = BikeSharing_df['dteday'].dt.day
```

In [33]:

```
# Since we have day, month and year of the record in separate columns, drop the variable 'dteday' which is redundant

BikeSharing_df = BikeSharing_df.drop("dteday", axis = 1)
```

- **Categorize** the days of a month as **'Beginning of Month', 'Mid Month' and 'End of Month'** each of which holds number of days with the interval : **[0,10],[10,20],[20-31]** respectively
- **Binning** is used to bucket the days of the month

In [34]:

```
BikeSharing_df['day'] = pd.cut( x = BikeSharing_df['day'], bins = [0,10,20,31] )
BikeSharing_df['day'] = BikeSharing_df['day'].cat.codes
BikeSharing_df['day'] = BikeSharing_df['day'].map({0 : "Beginning of Month", 1 : "Mid Month", 2 : "End of Month"})
```

In [35]:

```
# First 5 records of the dataframe after binning the 'day' variable

BikeSharing_df.head()
```

Out[35]:

	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	cnt	day
0	Spring	0	January	0	Saturday	0	Misty	14.11	18.18	80.58	10.75	985	Beginning of Month
1	Spring	0	January	0	Sunday	0	Misty	14.90	17.69	69.61	16.65	801	Beginning of Month
2	Spring	0	January	0	Monday	1	Clear	8.05	9.47	43.73	16.64	1349	Beginning of Month

	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	cnt	day
3	Spring	0	January	0	Tuesday	1	Clear	8.20	10.61	59.04	10.74	1562	Beginning of Month
4	Spring	0	January	0	Wednesday	1	Clear	9.31	11.46	43.70	12.52	1600	Beginning of Month

3. Exploratory Data Analysis (EDA)

Continuous variables of the dataframe :

- yr
- holiday
- workingday
- temp
- atemp
- hum
- windspeed
- **cnt (target variable)**

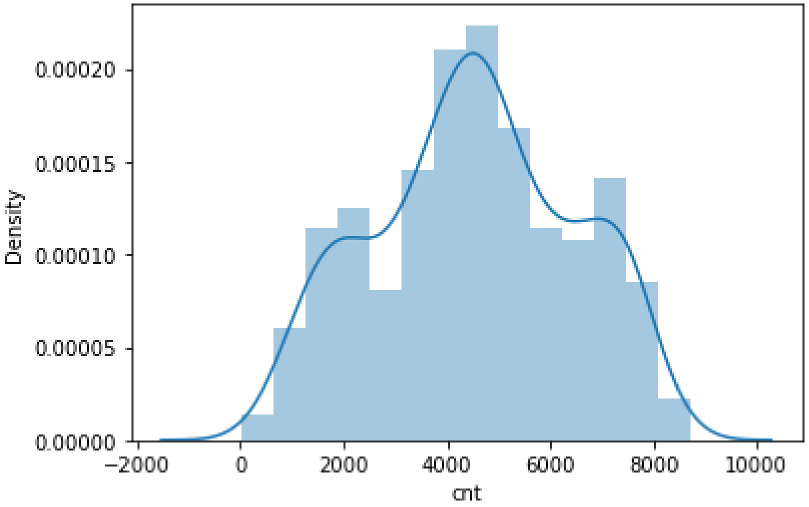
Categorical variables of the dataframe :

- season
- day
- mnth
- weathersit
- weekday

In [36]:

```
# Distribution plot for the target variable 'cnt'

sns.distplot(BikeSharing_df['cnt'])
plt.show()
```

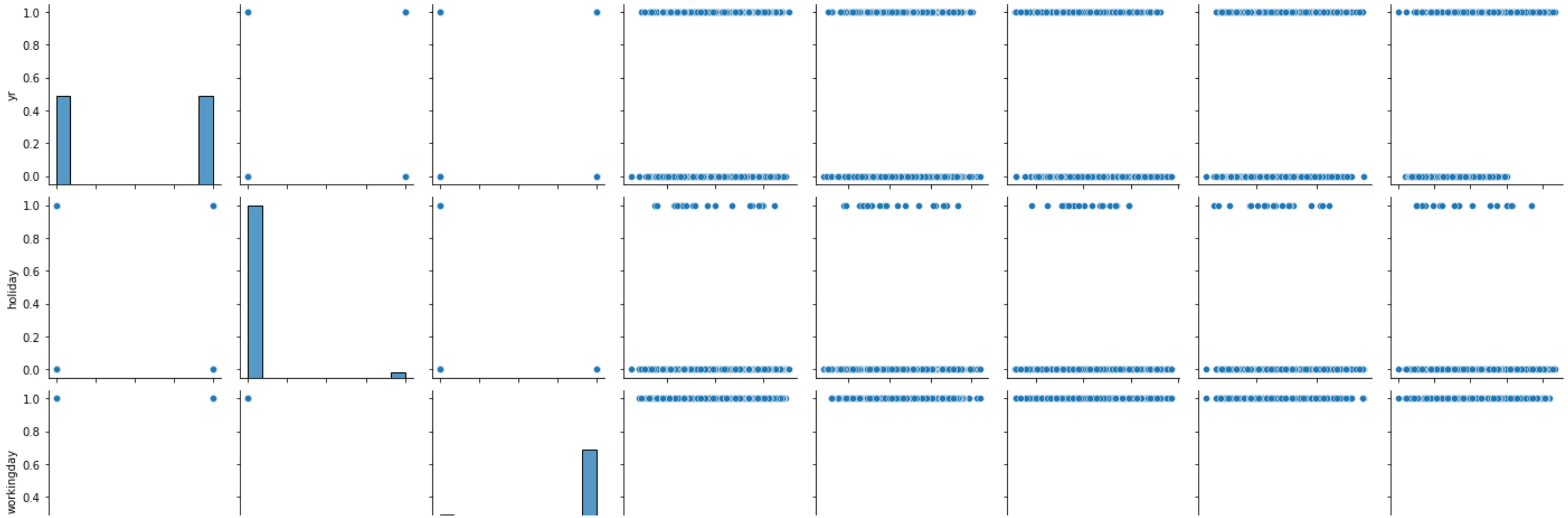


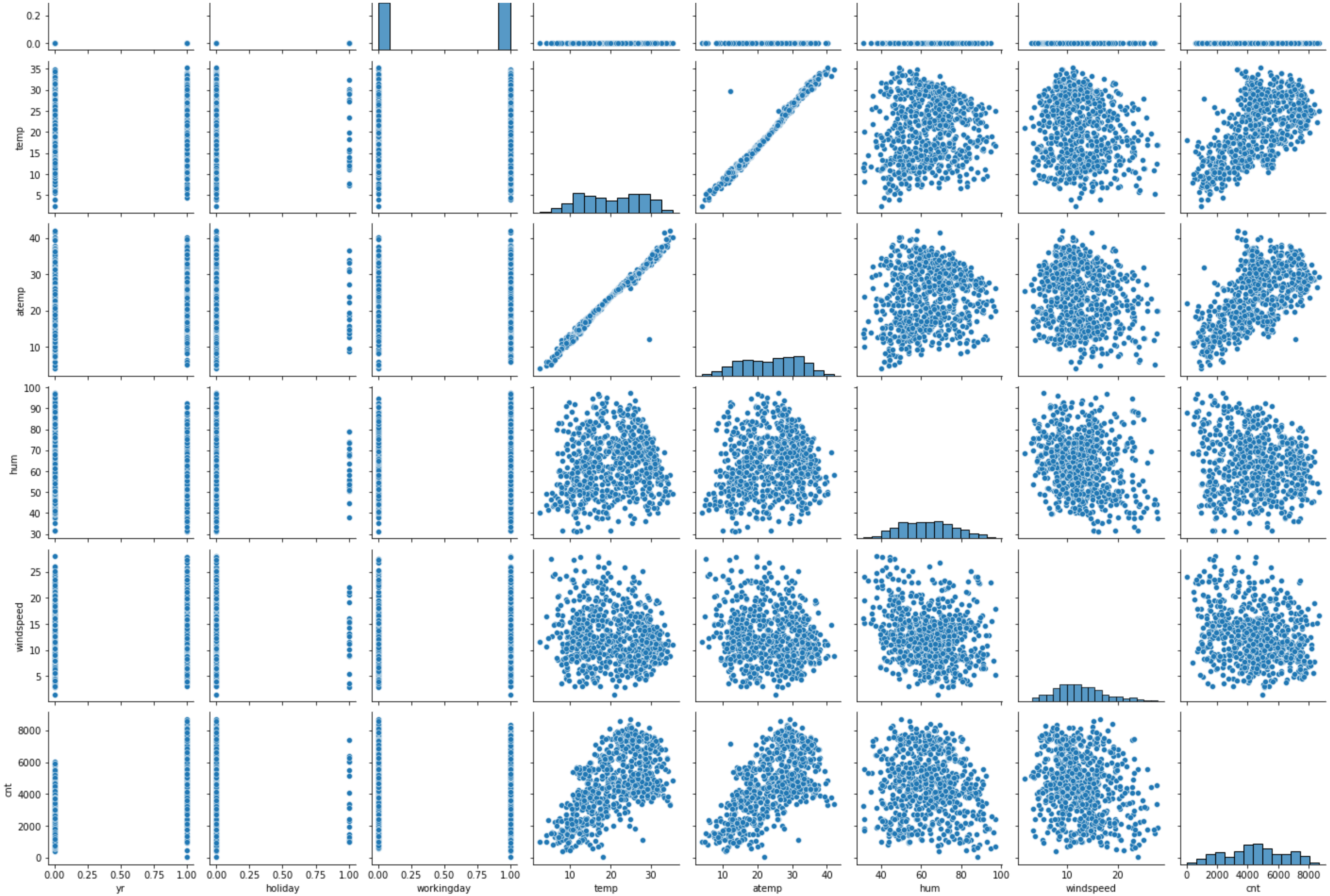
Inference - Average demand for shared bikes for the year 2018 - 2019 seems to be around 4000 - 5000

In [37]:

```
# Pairplots are used to learn the relationship between numerical variables of BikeSharing_df

sns.pairplot(BikeSharing_df)
plt.show()
```





Inferences :

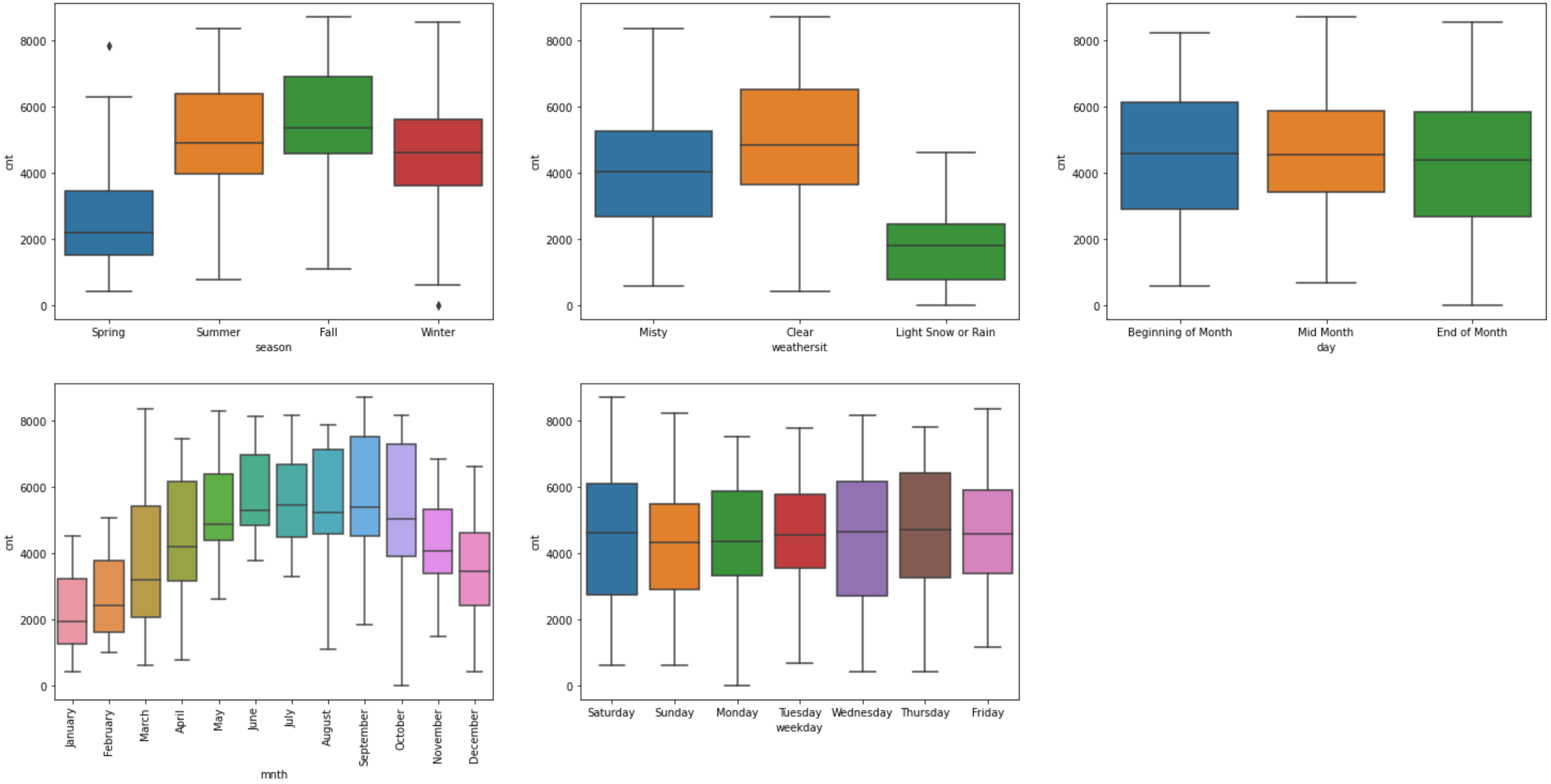
- The variables **'temp'** and **'atemp'** are **correlated**.

- **'temp'** and **'atemp'** show similar kind of relationship with all other variables.
- **'temp' and 'cnt'** variables are related linearly and also **'atemp' and 'cnt'** have linear relationship.
- **yr - 2019** shows relatively higher demand for shared bikes compared to the **yr - 2018**.
- From the histogram of **'holiday'**, it is evident that demand for shared bikes seems to be more for **non - holiday** days.
- Similarly, histogram of **'workingday'** shows that the demand falls for **non - working days**.

In [38]:

```
# Boxplots are used here to learn the relationship of categorical variables with the target variable 'cnt'

plt.figure(figsize = (25,12))
plt.subplot(2,3,1)
sns.boxplot(x = 'season', y = 'cnt', data = BikeSharing_df)
plt.subplot(2,3,2)
sns.boxplot(x = 'weathersit', y = 'cnt', data = BikeSharing_df)
plt.subplot(2,3,3)
sns.boxplot(x = 'day', y = 'cnt', data = BikeSharing_df)
plt.subplot(2,3,4)
plt.xticks(rotation = 90)
sns.boxplot(x = 'mnth', y = 'cnt', data = BikeSharing_df)
plt.subplot(2,3,5)
sns.boxplot(x = 'weekday', y = 'cnt', data = BikeSharing_df)
plt.show()
```

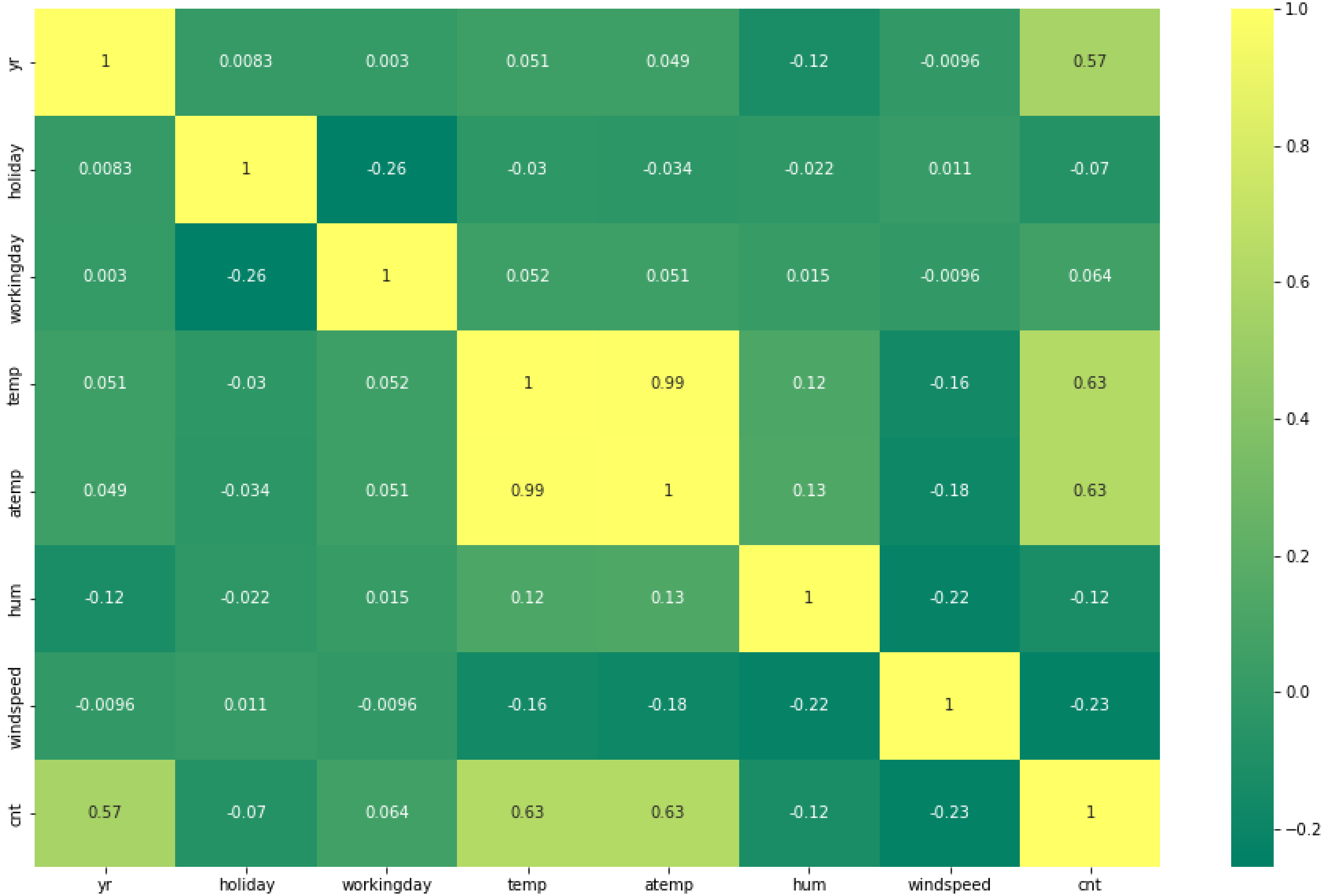


Inferences :

- Comparatively **'Fall'** season has more demand for shared bikes and **'Spring'** season has lesser demand.
- If the weather is **'Clear'**, then the demand for shared bikes is **high**.
- **'Beginning of Month'** shows a **little higher** demand for rental bikes compared to other days of the month.
- During **Snow or Rain**, most probably people **dont prefer** shared bikes.
- The demand for shared bikes is lesser for **Sundays** compared to other weekdays.
- Demand for shared bikes is typically **high** in the **mid of the year** and is relatively **low** towards the **beginning** and **end** of the year.
- **Thursdays** seem to have comparatively higher demand for rental bikes.
- In the month of **'September'**, the demand seems to **high** but for the month of **'January'**, the demand **drops**.

In [39]:
```
# Analysing the correlation of numeric variables in BikeSharing_df

plt.figure(figsize = (16,10))
sns.heatmap(BikeSharing_df.corr(), annot = True, cmap = "summer")
plt.show()
```

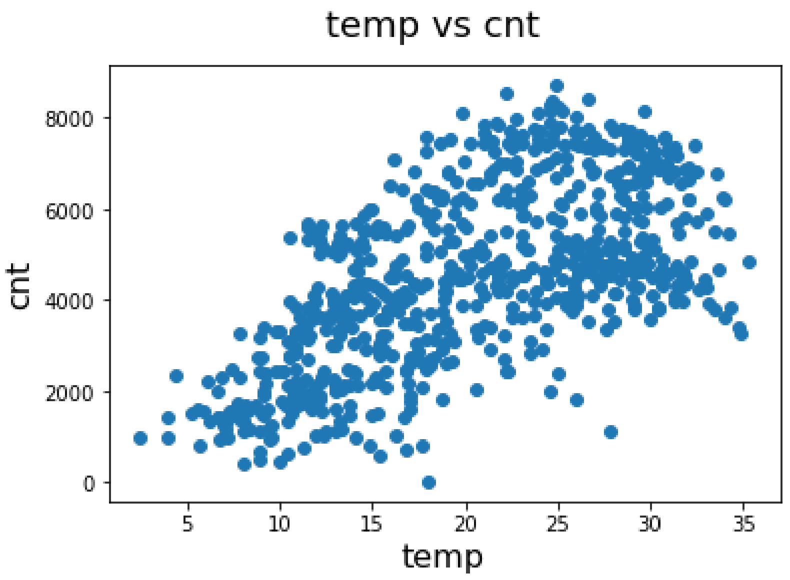


Inferences :

- The target variable **'cnt'** is **positively** correlated with **'yr', 'workingday', 'temp'** and **'atemp'**.
- The target variable **'cnt'** is **negatively** correlated with **'holiday', 'hum'** and **'windspeed'**.
- **'temp'** and **'atemp'** variables are **strongly** correlated.

In [40]:
```
#Scatter plot for 'temp' vs 'cnt'

fig = plt.figure()
plt.scatter(BikeSharing_df['temp'],BikeSharing_df['cnt'])
fig.suptitle('temp vs cnt', fontsize = 18)
plt.xlabel('temp', fontsize = 16)
plt.ylabel('cnt', fontsize = 16)
plt.show()
```

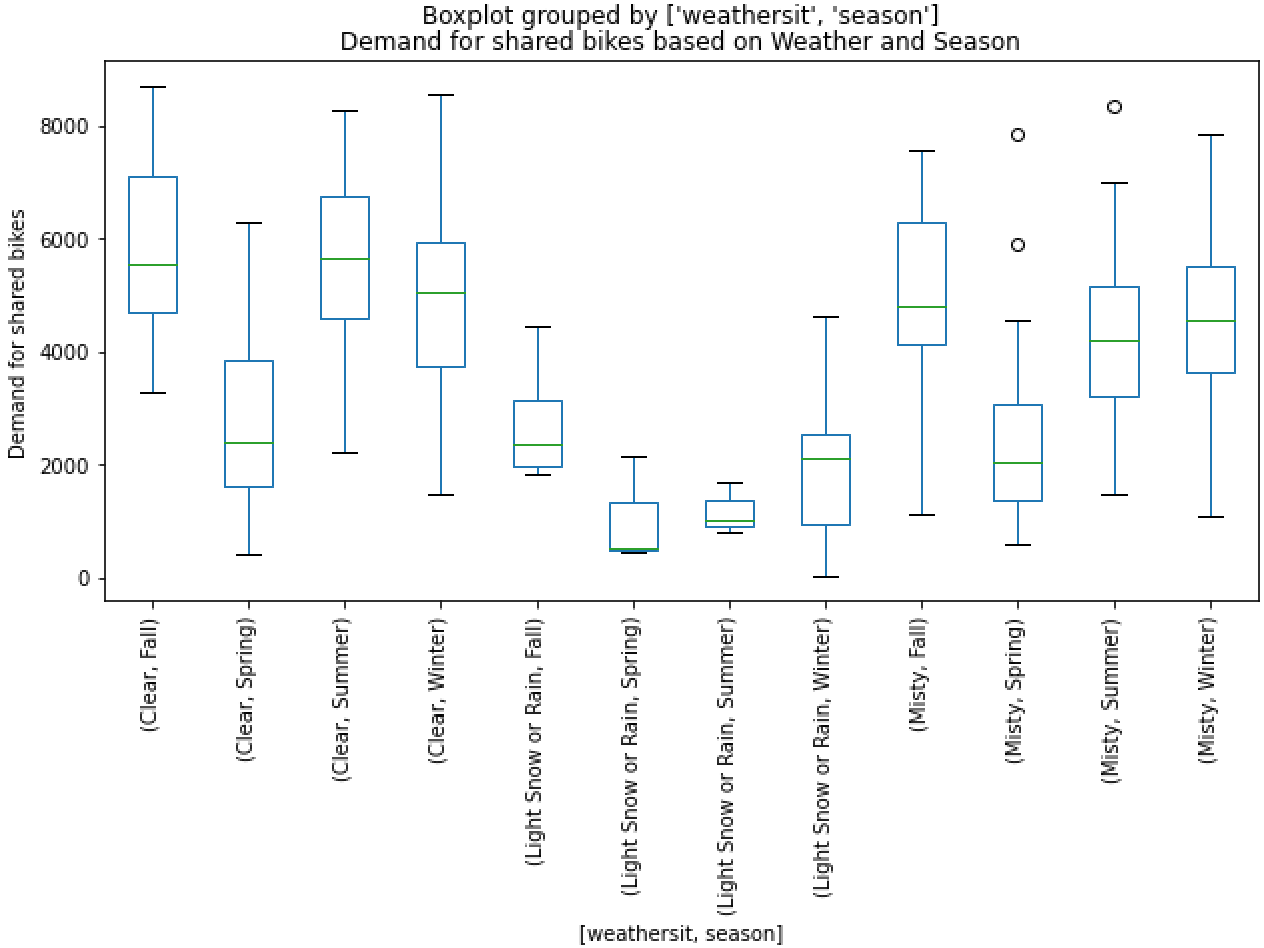


Inferences :

- **'temp'** is **positively** correlated with the target variable **'cnt'**.
- As **'temp'** increases, demand for shared bikes also **increases** but when the **'temp'** exceeds 30, the demand **drops**.

In [41]:
```
# Analysing how demand varies based on Weather and Season together using boxplots
```

```
BikeSharing_df.boxplot(by=['weathersit','season'], column='cnt', figsize=(10,5), grid = False)
plt.xticks(rotation=90)
plt.title("Demand for shared bikes based on Weather and Season")
plt.ylabel("Demand for shared bikes")
plt.show()
```

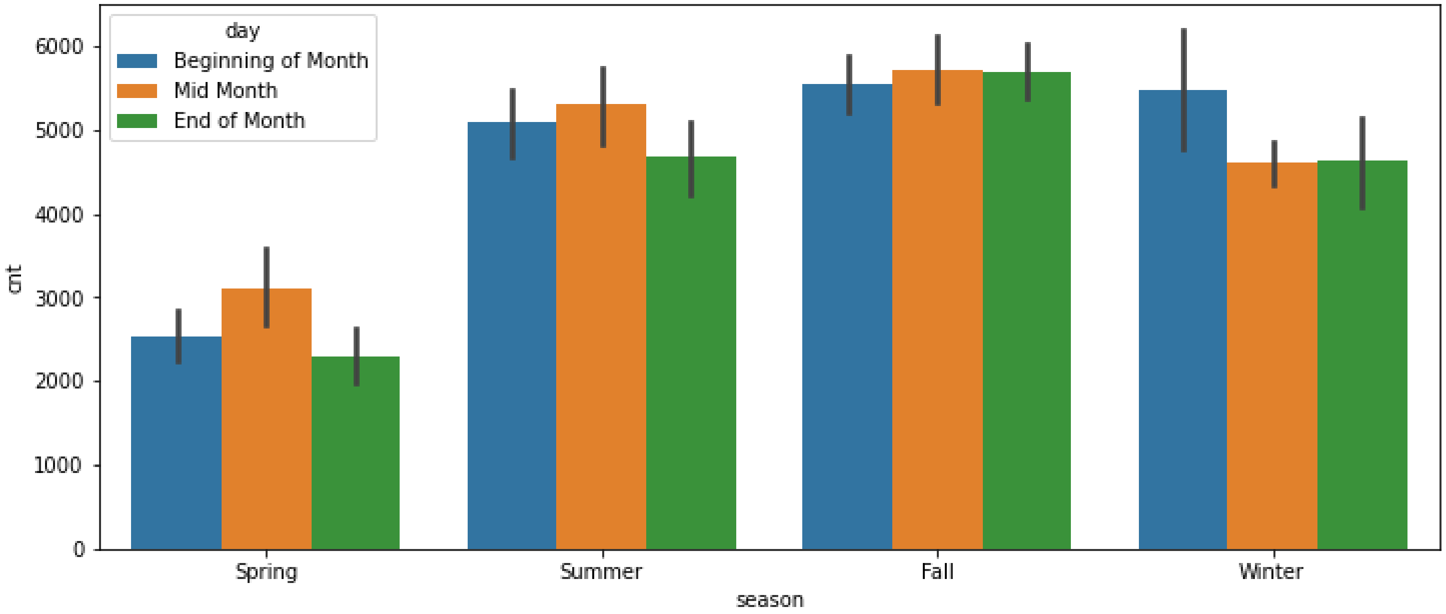


Inferences:

- **Irrespective of the season**, if there is **rain or snow**, the demand for rental bikes **drops**.
- Higher demands of shared bikes seem to occur during **'Fall'** season when the weather is **Clear**.

In [42]:
```
# Barplot to analyse how season and days of a month impact the demand for shared bikes

plt.figure(figsize = (12, 5))
sns.barplot(x = 'season', y = 'cnt', hue = 'day', data = BikeSharing_df)
plt.show()
```

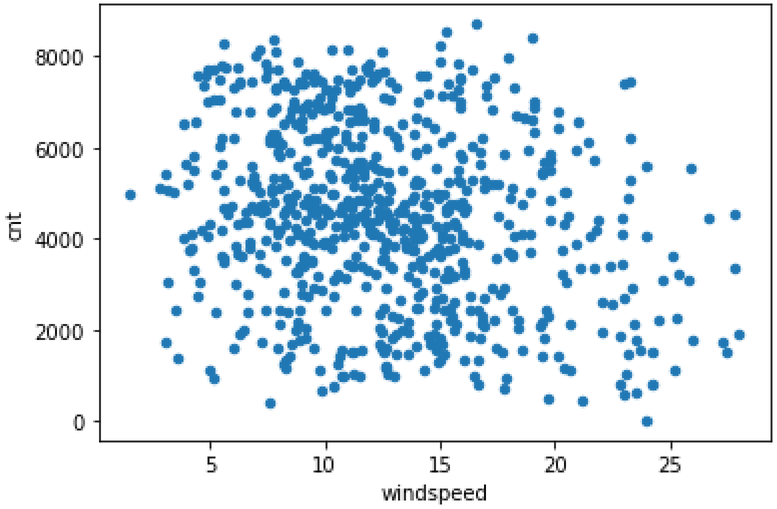


Inferences :

- The demand is relatively low during **'End of Month'**.
- During **'Spring'** season, the demand for rental bikes is low across the month, compared to other seasons.
- For **'Winter'** season, the demand is high in the **'Beginning of Month'** rather than other days of the month. This may be due to the transition in the season from **'Fall'** to **'Winter'**.

In [43]:
```
# Scatter plot of 'windspeed' and 'cnt'

BikeSharing_df.plot(kind="scatter", x="windspeed", y="cnt" )
plt.show()
```

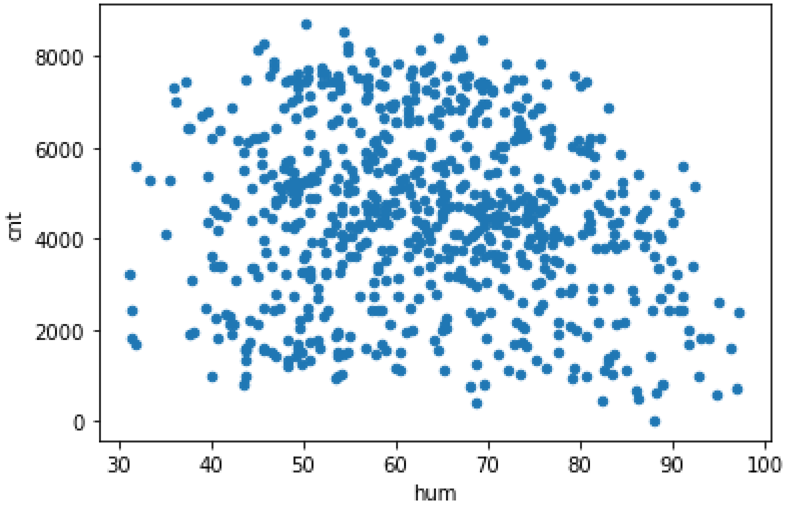


Inferences :

- From the scatter plot, it is evident that the demand for shared bikes is **more** when the **'windspeed' < 15**.
- As the **'windspeed'** increases, the demand drops.

In [44]:
```
# Scatter plot of 'hum' and 'cnt'

BikeSharing_df.plot(kind="scatter", x="hum", y="cnt" )
plt.show()
```



Inferences :

- The bike-sharing demand is **high** for the **humidity** range of **[40 - 80]**.
- Beyond this interval of humidity range, the demand drops.

Insights obtained as a result of EDA :

- The variables **'temp'** and **'atemp'** are **strongly** correlated and hence they show similar kind of relationship with all other variables.
- **'temp'** is **positively** correlated with the target variable **'cnt'**. As **'temp'** increases, demand for shared bikes also **increases** but when the **'temp'** exceeds 30, the demand **drops**.
- **yr - 2019** shows relatively good demand for shared bikes compared to the **yr - 2018**.
- From the histogram of **'holiday'**, it is evident that demand for shared bikes seems to be more for **non - holiday** days.
- Similarly, histogram of **'workingday'** shows that the demand falls for **non - working days**.
- Comparatively **'Fall'** season has more demand for shared bikes and **'Spring'** season has lesser demand.
- If the weather is **'Clear'**, then the demand for shared bikes is **high**.
- During **Snow or Rain**, most probably people **dont prefer** shared bikes.
- The demand for shared bikes is lesser for **Sundays** compared to other weekdays.
- In the month of **'September'**, the demand seems to **high** but for the month of **'January'**, the demand **drops**.
- Demand for rental bikes seems to be high on **Thursdays** rather than other weekdays.
- Demand for shared bikes is typically **high** in the **mid of the year** and is relatively **low** towards the **beginning** and **end** of the year.

- Heatmap shows that **'cnt'** variable is **positively** correlated with **'yr', 'workingday', 'temp' and 'atemp'** and **negatively** correlated with **'holiday', 'hum' and 'windspeed'**.
- **Irrespective of the season**, if there is **rain or snow**, the demand for rental bikes **drops**.
- Higher demands of shared bikes seem to occur during **'Fall'** season when the weather is **Clear**.
- The demand is relatively low during **'End of Month'**.
- During **'Spring'** season, the demand for rental bikes is low across the month, compared to other seasons.
- For **'Winter'** season, the demand is high in the **'Beginning of Month'** rather than other days of the month. This may be due to the transition in the season from **'Fall'** to **'Winter'**.
- The bike-sharing demand is **high** for the **humidity** range of **[40 - 80]**.
- Demand for shared bikes is **more** when the **'windspeed' < 15**.

4. MODEL BUILDING

From the **visualization** of data, it is evident that some of the features have **linear relationship** with the **target variable 'cnt'**.
Hence, a **Multiple Linear Regression model** can be built to **predict the demand** for shared bikes.

Assumptions of a linear regression model

Before building the model, let us assume that

1. **Linear relationship** exists between X and Y
2. Error terms are **normally** distributed
3. Error terms are **independent** to each other
4. Error terms have **constant variance (homoscedasticity)**

a. Data Preparation

In [45]:

```
# First 5 records of the dataframe

BikeSharing_df.head()
```

Out[45]:

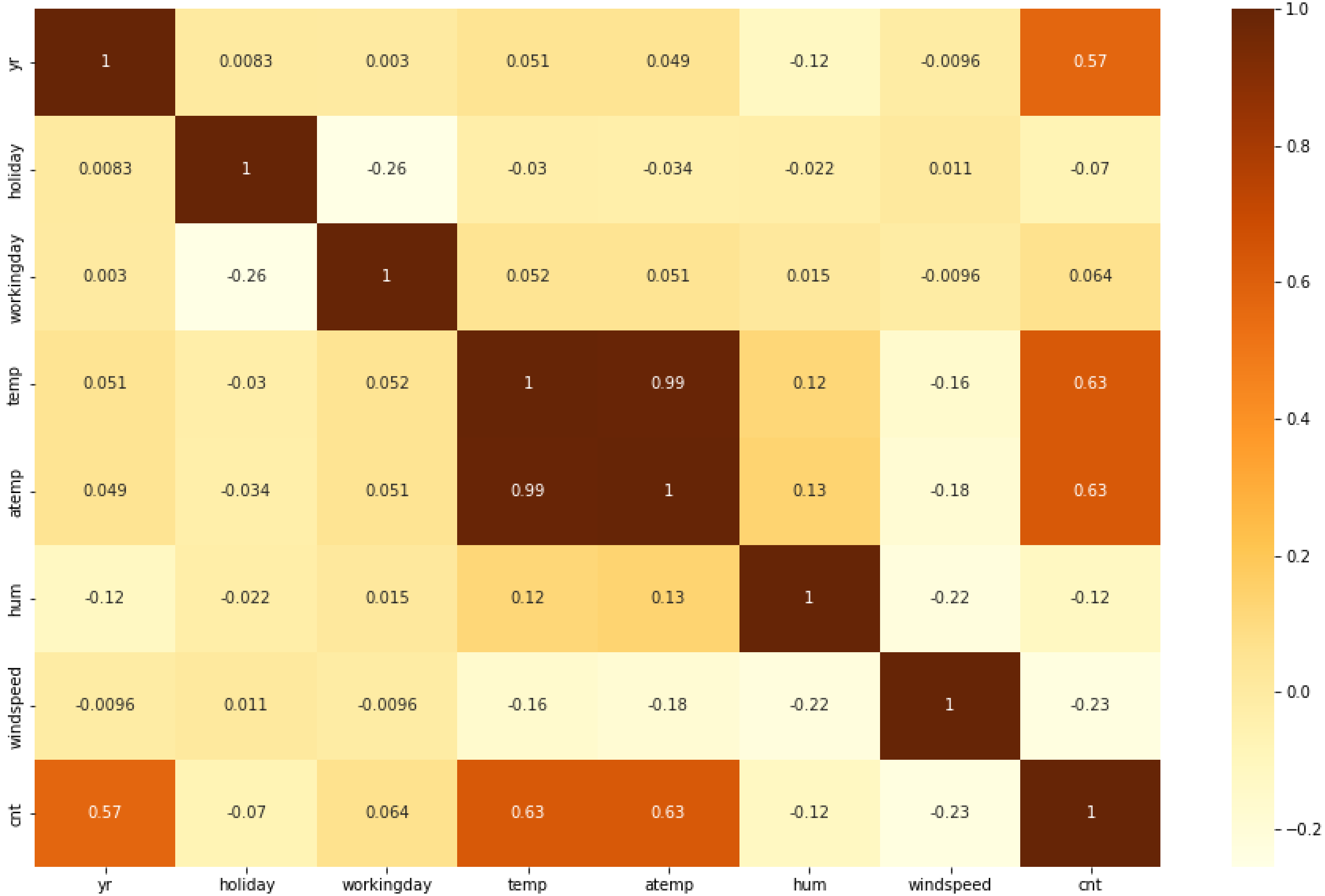
	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	cnt	day
0	Spring	0	January	0	Saturday	0	Misty	14.11	18.18	80.58	10.75	985	Beginning of Month

	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	cnt	day
1	Spring	0	January	0	Sunday	0	Misty	14.90	17.69	69.61	16.65	801	Beginning of Month
2	Spring	0	January	0	Monday	1	Clear	8.05	9.47	43.73	16.64	1349	Beginning of Month
3	Spring	0	January	0	Tuesday	1	Clear	8.20	10.61	59.04	10.74	1562	Beginning of Month
4	Spring	0	January	0	Wednesday	1	Clear	9.31	11.46	43.70	12.52	1600	Beginning of Month

In [46]:

```
# Analysing the correlation of numeric variables in BikeSharing_df

plt.figure(figsize = (16,10))
sns.heatmap(BikeSharing_df.corr(), annot = True, cmap = "YlOrBr")
plt.show()
```



Inference :

From the heatmap, it is clear that the variables **'temp'** and **'atemp'** are strongly **correlated**. Thus, to resolve **multicollinearity** issue in model building , drop one of the redundant features **'atemp'** from the dataset.

In [47]:

```
# Drop the feature 'atemp' to avoid multicollinearity
```

```
BikeSharing_df = BikeSharing_df.drop("atemp", axis = 1)
```

In [48]:
```
# Shape of the dataframe after removing redundant features

BikeSharing_df.shape
```

Out[48]: (720, 12)

In [49]:
```
# First 5 records of the dataframe

BikeSharing_df.head()
```

Out[49]:

	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	hum	windspeed	cnt	day
0	Spring	0	January	0	Saturday	0	Misty	14.11	80.58	10.75	985	Beginning of Month
1	Spring	0	January	0	Sunday	0	Misty	14.90	69.61	16.65	801	Beginning of Month
2	Spring	0	January	0	Monday	1	Clear	8.05	43.73	16.64	1349	Beginning of Month
3	Spring	0	January	0	Tuesday	1	Clear	8.20	59.04	10.74	1562	Beginning of Month
4	Spring	0	January	0	Wednesday	1	Clear	9.31	43.70	12.52	1600	Beginning of Month

> **DEPENDENT VARIABLE (TARGET VARIABLE) : cnt**
>
> **INDEPENDENT VARIABLES (PREDICTORS) :**
>
> - day
> - yr
> - mnth
> - season
> - weathersit
> - holiday
> - weekday
> - workingday

> - temp
> - hum
> - windspeed

One hot encoding for categorical features

- **One hot encoding** can be defined as the essential process of **converting the categorical data variables** to be provided to machine and deep learning algorithms which in turn **improve predictions** as well as **classification accuracy** of a model.

- This encoding ensures that machine learning **does not assume that higher numbers are more important**.

- **One hot encoding** is similar to **dummy encoding** but for quick **data cleaning and EDA**, it is preferred to use **pandas 'get_dummies'** and to transform a categorical column to multiple binary columns for **machine learning**, it's better to use **OneHotEncoder()**.

- One hot encoding does the same things as get dummies but in addition, it **saves** the exploded categories into it's **object**. Saving exploded categories is extremely useful when the **same data pre-processing** has to be applied on the test set.

In [50]:
```
# Adding categorical variables of BikeSharing_df to 'BikeSharing_Categorical'

BikeSharing_Categorical = ['season','weathersit','day','weekday','mnth']
```

Creating an object for OneHotEncoder() and dropping the first dummy variable created so that multicollinearity issue doesnot impact the interpretation of the model.

In [51]:
```
ohe_obj  =  OneHotEncoder(sparse = False, drop = 'first')
```

In [52]:
```
# Create dummies for categorical variables of BikeSharing_df using 'OneHotEncoder' and store it in 'OHE_Array'

OHE_Array = ohe_obj.fit_transform(BikeSharing_df[BikeSharing_Categorical])
```

In [53]:
```
# Convert the array to Dataframe 'OHE_DF' with index of BikeSharing_df

OHE_DF = pd.DataFrame(OHE_Array, index = BikeSharing_df.index)
```

In [54]:
```
# Rename the columns of dummy variables with meaningful feature names

OHE_DF.columns = ohe_obj.get_feature_names()

OHE_DF = OHE_DF.rename(columns = lambda x : x[3:])
```

In [55]:
```
# First 5 records of OHE_DF

OHE_DF.head()
```

Out[55]:

	Spring	Summer	Winter	Light Snow or Rain	Misty	End of Month	Mid Month	Monday	Saturday	Sunday	...	December	February	January	July	June	March	May	Nov
0	1.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	0.0	...	0.0	0.0	1.0	0.0	0.0	0.0	0.0	
1	1.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	...	0.0	0.0	1.0	0.0	0.0	0.0	0.0	
2	1.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	...	0.0	0.0	1.0	0.0	0.0	0.0	0.0	
3	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	1.0	0.0	0.0	0.0	0.0	
4	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	1.0	0.0	0.0	0.0	0.0	

5 rows × 24 columns

In [56]:
```
# Drop the categorical columns from BikeSharing_df and store the rest of the numerical columns in 'BikeSharing_Numerical'

BikeSharing_Numerical = BikeSharing_df.drop(columns = BikeSharing_Categorical)
```

In [57]:
```
# Concatenate the numerical variables with OHE_DF dataframe

BikeSharing = pd.concat([OHE_DF, BikeSharing_Numerical], axis=1)
```

In [58]:

```
# Shape of the dataframe after concatenating one hot encoded variables

BikeSharing.shape
```

Out[58]: (720, 31)

In [59]:

```
# First 5 records of the dataframe after concatenating one hot encoded variables

BikeSharing.head()
```

Out[59]:

	Spring	Summer	Winter	Light Snow or Rain	Misty	End of Month	Mid Month	Monday	Saturday	Sunday	...	November	October	September	yr	holiday	workingday	te
0	1.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	0.0	...	0.0	0.0	0.0	0	0	0	1
1	1.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	...	0.0	0.0	0.0	0	0	0	1
2	1.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	...	0.0	0.0	0.0	0	0	1	
3	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0	0	1	
4	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0	0	1	

5 rows × 31 columns

Now all the columns have numerical values so that we can easily use it for the machine learning process.

b. Splitting the Data into Training and Testing Sets

In [60]:

```
# We specify random_state so that the train and test data set always have the same set of split each time

df_train, df_test = train_test_split(BikeSharing, train_size = 0.7, test_size = 0.3, random_state = 100)
```

c. Rescaling the Features using MinMax Scaler

In [61]:

```
# Create an instance of scaler

scaler = MinMaxScaler()
```

In [62]:
```
# Apply scaler() to the columns 'temp','hum','windspeed','cnt'

scale_vars = ['temp','hum','windspeed','cnt']

df_train[scale_vars] = scaler.fit_transform(df_train[scale_vars])
```

In [63]:
```
# First 5 records of the training dataset after scaling
# All values of the training set ranges from 0 to 1

df_train.head()
```

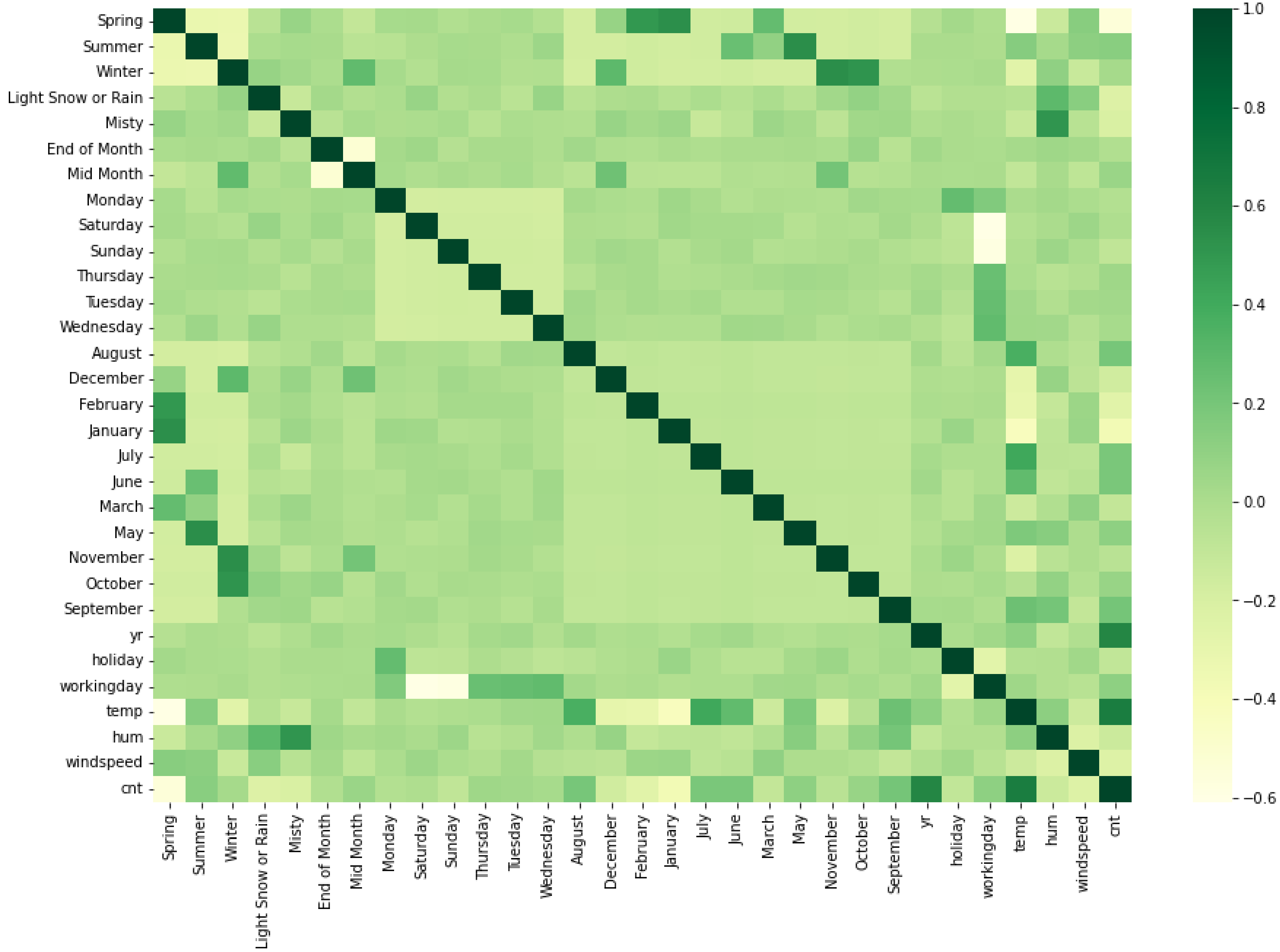
Out[63]:

	Spring	Summer	Winter	Light Snow or Rain	Misty	End of Month	Mid Month	Monday	Saturday	Sunday	...	November	October	September	yr	holiday	workingday
423	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	1	0	1
728	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	...	0.0	0.0	0.0	1	0	0
482	0.0	1.0	0.0	0.0	1.0	1.0	0.0	0.0	1.0	0.0	...	0.0	0.0	0.0	1	0	0
114	0.0	1.0	0.0	0.0	0.0	1.0	0.0	1.0	0.0	0.0	...	0.0	0.0	0.0	0	0	1
582	0.0	0.0	0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	...	0.0	0.0	0.0	1	0	1

5 rows × 31 columns

In [64]:
```
# Analysing the correlation of variables in df_train

plt.figure(figsize = (15,10))
sns.heatmap(df_train.corr(), cmap = "YlGn")
plt.show()
```



Inference - Some of the predictors seem to be dependent on each other slightly, which can be resolved by calculating VIF values and dropping the redundant features during model building.

Defining 'X' (INDEPENDENT VARIABLES) and 'Y' (DEPENDENT VARIABLE) for model building

In [65]:
```
y_train = df_train.pop("cnt")
X_train = df_train
```

In [66]:
```
# Features selected for RFE (Recursive Feature Elimination)

X_train.columns
```

Out[66]:
```
Index(['Spring', 'Summer', 'Winter', 'Light Snow or Rain', 'Misty',
       'End of Month', 'Mid Month', 'Monday', 'Saturday', 'Sunday', 'Thursday',
       'Tuesday', 'Wednesday', 'August', 'December', 'February', 'January',
       'July', 'June', 'March', 'May', 'November', 'October', 'September',
       'yr', 'holiday', 'workingday', 'temp', 'hum', 'windspeed'],
      dtype='object')
```

d. Building Multiple linear regression model using RFE (Recursive Feature Elimination)

MODEL 1

We use **LinearRegression** function from **SciKit Learn** for its compatibility with RFE

In [67]:
```
# Creating an instance of LinearRegression and fitting a line

lr = LinearRegression()
lr.fit(X_train, y_train)
```

Out[67]:
```
LinearRegression()
```

In [68]:
```
# Running RFE with output number of variables equal to 20

rfe = RFE(lr, 20)
rfe = rfe.fit(X_train, y_train)
```

In [69]:
```
# printing the list of predictors with its rfe ranking and support values
```

```
list(zip(X_train.columns,rfe.support_,rfe.ranking_))
```

Out[69]:
```
[('Spring', True, 1),
 ('Summer', True, 1),
 ('Winter', True, 1),
 ('Light Snow or Rain', True, 1),
 ('Misty', True, 1),
 ('End of Month', False, 10),
 ('Mid Month', True, 1),
 ('Monday', False, 2),
 ('Saturday', True, 1),
 ('Sunday', True, 1),
 ('Thursday', False, 8),
 ('Tuesday', False, 4),
 ('Wednesday', False, 7),
 ('August', False, 11),
 ('December', True, 1),
 ('February', True, 1),
 ('January', True, 1),
 ('July', True, 1),
 ('June', False, 5),
 ('March', False, 9),
 ('May', False, 3),
 ('November', True, 1),
 ('October', False, 6),
 ('September', True, 1),
 ('yr', True, 1),
 ('holiday', True, 1),
 ('workingday', True, 1),
 ('temp', True, 1),
 ('hum', True, 1),
 ('windspeed', True, 1)]
```

In [70]:
```
# Storing the features selected by RFE in 'col'

col = X_train.columns[rfe.support_]
col
```

Out[70]:
```
Index(['Spring', 'Summer', 'Winter', 'Light Snow or Rain', 'Misty',
       'Mid Month', 'Saturday', 'Sunday', 'December', 'February', 'January',
       'July', 'November', 'September', 'yr', 'holiday', 'workingday', 'temp',
       'hum', 'windspeed'],
      dtype='object')
```

In [71]:
```
# Features eliminated by RFE

X_train.columns[~rfe.support_]
```

Out[71]:
```
Index(['End of Month', 'Monday', 'Thursday', 'Tuesday', 'Wednesday', 'August',
       'June', 'March', 'May', 'October'],
      dtype='object')
```

In [72]:
```
# Function for building linear model using 'statsmodels'

def stats_model_building(columns):
    X_train_set = X_train[columns]              # Creating 'X_train_set' dataframe with selective predictors
    X_train_set = sm.add_constant(X_train_set) # Adding a constant variable
    lm = sm.OLS(y_train,X_train_set).fit()     # Fitting the line using 'statsmodels' obj
    print(lm.summary())                        # prints the summary statistics of the model
    return lm                                  # returns lm
```

In [73]:
```
# Function for calculating VIF (variance_inflation_factor)
# VIF is used for detecting multicollinearity among predictors

def calculate_VIF(X_train_set):
    vif = pd.DataFrame()
    vif['Features'] = X_train_set.columns
    vif['VIF'] = [variance_inflation_factor(X_train_set.values, i) for i in range(X_train_set.shape[1])]
    vif['VIF'] = round(vif['VIF'], 2)
    vif = vif.sort_values(by = "VIF", ascending = False)
    return vif
```

Re-building the RFE model using 'statsmodels' for detailed statistics

In [74]:
```
# Building a multiple linear regression model using 'statsmodels' with the features selected by RFE

lm_1 = stats_model_building(col)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.850
Model:                            OLS   Adj. R-squared:                  0.844
```

```
Method:                 Least Squares   F-statistic:                     144.3
Date:                Wed, 13 Apr 2022   Prob (F-statistic):          3.33e-185
Time:                        15:53:37   Log-Likelihood:                 506.28
No. Observations:                 503   AIC:                            -972.6
Df Residuals:                     483   BIC:                            -888.2
Df Model:                          19
Covariance Type:            nonrobust
========================================================================================
                          coef    std err          t      P>|t|      [0.025      0.975]
----------------------------------------------------------------------------------------
const                   0.2293      0.028      8.155      0.000       0.174       0.285
Spring                 -0.0612      0.023     -2.682      0.008      -0.106      -0.016
Summer                  0.0354      0.016      2.246      0.025       0.004       0.066
Winter                  0.0922      0.018      5.028      0.000       0.056       0.128
Light Snow or Rain     -0.2278      0.030     -7.572      0.000      -0.287      -0.169
Misty                  -0.0481      0.011     -4.430      0.000      -0.069      -0.027
Mid Month               0.0372      0.009      4.020      0.000       0.019       0.055
Saturday                0.0988      0.012      8.391      0.000       0.076       0.122
Sunday                  0.0473      0.012      3.900      0.000       0.023       0.071
December               -0.0720      0.019     -3.835      0.000      -0.109      -0.035
February               -0.0439      0.022     -1.956      0.051      -0.088       0.000
January                -0.0694      0.023     -3.063      0.002      -0.114      -0.025
July                   -0.0271      0.018     -1.482      0.139      -0.063       0.009
November               -0.0821      0.019     -4.234      0.000      -0.120      -0.044
September               0.0648      0.017      3.855      0.000       0.032       0.098
yr                      0.2288      0.008     27.791      0.000       0.213       0.245
holiday                -0.0070      0.020     -0.347      0.729      -0.046       0.032
workingday              0.0901      0.010      9.447      0.000       0.071       0.109
temp                    0.4320      0.039     11.169      0.000       0.356       0.508
hum                    -0.1465      0.028     -5.275      0.000      -0.201      -0.092
windspeed              -0.1522      0.023     -6.587      0.000      -0.198      -0.107
==============================================================================
Omnibus:                       78.016   Durbin-Watson:                   1.974
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              187.991
Skew:                          -0.801   Prob(JB):                     1.51e-41
Kurtosis:                       5.531   Cond. No.                     2.09e+16
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 3.49e-30. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
```

Inferences from Summary statistics of Model 1 :

- **R-Squared** value for **Model 1** is **0.850**
- **Adjusted R-Squared** value for **Model 1** is **0.844**
- **Prob (F-statistic): 3.33e-185** is very low which implies that the **overall significance** is statistically good
- Features with **Positive** coefficients : **const, Summer, Winter, Mid Month, Saturday, Sunday, September, yr, workingday, temp**
- Features with **Negative** coefficients : **Spring, Light Snow or Rain, Misty, December, February, January, July, November, holiday, hum, windspeed**
- Features with **p-value > 0.05** (Significance level) : holiday **(0.729)**, July **(0.139)**, February **(0.051)**

In [75]:

```
# Variable Inflation Factor is used to detect multicollinearity among predictor variables
# Calculate VIF for the predictor variables of Model 1

calculate_VIF(X_train[col])
```

Out[75]:

	Features	VIF
16	workingday	50.90
6	Saturday	11.31
7	Sunday	10.92
0	Spring	5.84
17	temp	4.89
2	Winter	3.92
15	holiday	3.48
1	Summer	2.82
10	January	2.43
18	hum	2.12
9	February	2.06
12	November	1.93
8	December	1.81
4	Misty	1.70
11	July	1.55

	Features	VIF
13	September	1.42
3	Light Snow or Rain	1.41
19	windspeed	1.19
5	Mid Month	1.15
14	yr	1.05

Inference - VIF values are too high (i.e. > 5) for the features 'workingday' - 50.90, 'Saturday' - 11.31, 'Sunday' - 10.92, 'Spring' - 5.84

- **p - value** of the feature **holiday = 0.729** which is greater than the significance level of **alpha = 0.05**
- The feature **'workingday'** has the highest **VIF** of 50.90
- When both the **'p - value' and 'VIF'** are higher than the tolerance level, improvement in **significance** is given priority
- So drop the feature **'holiday'** and re-build the **Model 2** using 'statsmodels'

In [76]:

```
# Drop the feature 'holiday' from the dataframe of selected predictor variables

X_train_model_2 = X_train[col].drop("holiday", axis = 1)
```

In [77]:

```
# Selected features for Model 2

X_train_model_2.columns
```

Out[77]:

```
Index(['Spring', 'Summer', 'Winter', 'Light Snow or Rain', 'Misty',
       'Mid Month', 'Saturday', 'Sunday', 'December', 'February', 'January',
       'July', 'November', 'September', 'yr', 'workingday', 'temp', 'hum',
       'windspeed'],
      dtype='object')
```

MODEL 2

In [78]:

```
# Re-build the model using 'statsmodels' after removing the feature 'holiday' from Model 1

lm_2 = stats_model_building(X_train_model_2.columns)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.850
Model:                            OLS   Adj. R-squared:                  0.844
Method:                 Least Squares   F-statistic:                     144.3
Date:                Wed, 13 Apr 2022   Prob (F-statistic):          3.33e-185
Time:                        15:53:37   Log-Likelihood:                 506.28
No. Observations:                 503   AIC:                            -972.6
Df Residuals:                     483   BIC:                            -888.2
Df Model:                          19
Covariance Type:            nonrobust
======================================================================================
                         coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------------
const                  0.2223      0.042      5.285      0.000       0.140       0.305
Spring                -0.0612      0.023     -2.682      0.008      -0.106      -0.016
Summer                 0.0354      0.016      2.246      0.025       0.004       0.066
Winter                 0.0922      0.018      5.028      0.000       0.056       0.128
Light Snow or Rain    -0.2278      0.030     -7.572      0.000      -0.287      -0.169
Misty                 -0.0481      0.011     -4.430      0.000      -0.069      -0.027
Mid Month              0.0372      0.009      4.020      0.000       0.019       0.055
Saturday               0.1058      0.025      4.213      0.000       0.056       0.155
Sunday                 0.0543      0.025      2.147      0.032       0.005       0.104
December              -0.0720      0.019     -3.835      0.000      -0.109      -0.035
February              -0.0439      0.022     -1.956      0.051      -0.088       0.000
January               -0.0694      0.023     -3.063      0.002      -0.114      -0.025
July                  -0.0271      0.018     -1.482      0.139      -0.063       0.009
November              -0.0821      0.019     -4.234      0.000      -0.120      -0.044
September              0.0648      0.017      3.855      0.000       0.032       0.098
yr                     0.2288      0.008     27.791      0.000       0.213       0.245
workingday             0.0971      0.023      4.170      0.000       0.051       0.143
temp                   0.4320      0.039     11.169      0.000       0.356       0.508
hum                   -0.1465      0.028     -5.275      0.000      -0.201      -0.092
windspeed             -0.1522      0.023     -6.587      0.000      -0.198      -0.107
==============================================================================
Omnibus:                       78.016   Durbin-Watson:                   1.974
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              187.991
Skew:                          -0.801   Prob(JB):                     1.51e-41
Kurtosis:                       5.531   Cond. No.                         26.2
==============================================================================
```

```
Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

Inferences from Summary statistics of Model 2 :

- **R-Squared** value for **Model 2** is **0.850**
- **Adjusted R-Squared** value for **Model 2** is **0.844**
- R-Squared and Adjusted R-Squared values **remain the same** even after removing the feature 'holiday' from Model 1
- **Prob (F-statistic): 3.33e-185** is very low which implies that the **overall significance** is statistically good
- Features with **Positive** coefficients : **const, Summer, Winter, Mid Month, Saturday, Sunday, September, yr, workingday, temp**
- Features with **Negative** coefficients : **Spring, Light Snow or Rain, Misty, December, February, January, July, November, hum, windspeed**
- No change in signs of coefficients compared to previous model
- Features with **p-value > 0.05** (Significance level) : July **(0.139)**, February **(0.051)**

In [79]:

```
# Calculate VIF for the predictor variables of Model 2

calculate_VIF(X_train_model_2)
```

Out[79]:

	Features	VIF
16	temp	17.68
15	workingday	15.59
17	hum	13.56
18	windspeed	6.06
0	Spring	5.77
6	Saturday	4.01
2	Winter	4.00
7	Sunday	3.92
1	Summer	2.83
4	Misty	2.69
10	January	2.38

	Features	VIF
9	February	2.12
14	yr	2.11
12	November	1.95
8	December	1.84
5	Mid Month	1.68
11	July	1.67
13	September	1.51
3	Light Snow or Rain	1.44

Inference - VIF values are reduced slightly compared to previous model. Features with VIF > 5 : ' temp' - 17.68, 'workingday' - 15.59, 'hum' - 13.56, 'windspeed' - 6.06, 'Spring' - 5.77

- **p - value** of the feature **July = 0.139** which is greater than the significance level of **alpha = 0.05**
- The feature **'temp'** has the highest **VIF** of 17.68
- When both the **'p - value' and 'VIF'** are higher than the tolerance level, improvement in **significance** is given priority
- So drop the feature **'July'** and re-build the **Model 3** using 'statsmodels'

In [80]:
```
# Drop the feature 'July' from the dataframe of selected predictor variables

X_train_model_3 = X_train_model_2.drop("July", axis = 1)
```

In [81]:
```
# Selected features for Model 3

X_train_model_3.columns
```

Out[81]:
```
Index(['Spring', 'Summer', 'Winter', 'Light Snow or Rain', 'Misty',
       'Mid Month', 'Saturday', 'Sunday', 'December', 'February', 'January',
       'November', 'September', 'yr', 'workingday', 'temp', 'hum',
       'windspeed'],
      dtype='object')
```

MODEL 3

In [82]:

```
# Re-build the model using 'statsmodels' after removing the feature 'July' from Model 2

lm_3 = stats_model_building(X_train_model_3.columns)
```

```
                            OLS Regression Results                            
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.850
Model:                            OLS   Adj. R-squared:                  0.844
Method:                 Least Squares   F-statistic:                     151.8
Date:                Wed, 13 Apr 2022   Prob (F-statistic):          7.92e-186
Time:                        15:53:37   Log-Likelihood:                 505.14
No. Observations:                 503   AIC:                            -972.3
Df Residuals:                     484   BIC:                            -892.1
Df Model:                          18                                         
Covariance Type:            nonrobust                                         
======================================================================================
                         coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------------
const                  0.2166      0.042      5.165      0.000       0.134       0.299
Spring                -0.0544      0.022     -2.431      0.015      -0.098      -0.010
Summer                 0.0438      0.015      2.982      0.003       0.015       0.073
Winter                 0.0987      0.018      5.542      0.000       0.064       0.134
Light Snow or Rain    -0.2303      0.030     -7.656      0.000      -0.289      -0.171
Misty                 -0.0483      0.011     -4.444      0.000      -0.070      -0.027
Mid Month              0.0374      0.009      4.035      0.000       0.019       0.056
Saturday               0.1059      0.025      4.212      0.000       0.056       0.155
Sunday                 0.0545      0.025      2.154      0.032       0.005       0.104
December              -0.0726      0.019     -3.862      0.000      -0.110      -0.036
February              -0.0447      0.022     -1.992      0.047      -0.089      -0.001
January               -0.0710      0.023     -3.130      0.002      -0.115      -0.026
November              -0.0820      0.019     -4.223      0.000      -0.120      -0.044
September              0.0727      0.016      4.549      0.000       0.041       0.104
yr                     0.2292      0.008     27.815      0.000       0.213       0.245
workingday             0.0979      0.023      4.200      0.000       0.052       0.144
temp                   0.4232      0.038     11.059      0.000       0.348       0.498
hum                   -0.1439      0.028     -5.185      0.000      -0.198      -0.089
windspeed             -0.1509      0.023     -6.528      0.000      -0.196      -0.105
==============================================================================
Omnibus:                       80.953   Durbin-Watson:                   1.991
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              193.037
```

```
Skew:                          -0.833   Prob(JB):                     1.21e-42
Kurtosis:                       5.537   Cond. No.                         26.1
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

Inferences from Summary statistics of Model 3 :

- **R-Squared** value for **Model 3** is **0.850**
- **Adjusted R-Squared** value for **Model 3** is **0.844**
- R-Squared and Adjusted R-Squared values **remain the same** even after removing the feature 'July' from Model 2
- **Prob (F-statistic): 7.92e-186** is very low which implies that the **overall significance** is statistically good
- Features with **Positive** coefficients : **const, Summer, Winter, Mid Month, Saturday, Sunday, September, yr, workingday, temp**
- Features with **Negative** coefficients : **Spring, Light Snow or Rain, Misty, December, February, January, November, hum, windspeed**
- No change in signs of coefficients compared to previous model
- No features with **p-value > 0.05** (Significance level)

In [83]:

```
# Calculate VIF for the predictor variables of Model 3

calculate_VIF(X_train_model_3)
```

Out[83]:

	Features	VIF
15	temp	16.16
14	workingday	15.57
16	hum	13.52
17	windspeed	6.06
0	Spring	5.59
6	Saturday	4.00
7	Sunday	3.91
2	Winter	3.79
4	Misty	2.69

	Features	VIF
1	Summer	2.45
10	January	2.36
13	yr	2.11
9	February	2.11
11	November	1.95
8	December	1.84
5	Mid Month	1.68
3	Light Snow or Rain	1.44
12	September	1.37

Inference - VIF values are reduced slightly compared to previous model. Features with VIF > 5 : ' temp' - 16.16, 'workingday' - 15.57, 'hum' - 13.52, 'windspeed' - 6.06, 'Spring' - 5.59

- No **p - values** of the features are greater than the significance level of **alpha = 0.05**
- The feature **'temp'** has the highest **VIF** of 16.16
- Thus, in order to improve the VIF values and **resolve the multicollinearity** issue, let us drop the feature **'temp'** and re-build the **Model 4** using 'statsmodels'

In [84]:
```
# Drop the feature 'temp' from the dataframe of selected predictor variables

X_train_model_4 = X_train_model_3.drop("temp", axis = 1)
```

In [85]:
```
# Selected features for Model 4

X_train_model_4.columns
```

Out[85]:
```
Index(['Spring', 'Summer', 'Winter', 'Light Snow or Rain', 'Misty',
       'Mid Month', 'Saturday', 'Sunday', 'December', 'February', 'January',
       'November', 'September', 'yr', 'workingday', 'hum', 'windspeed'],
      dtype='object')
```

MODEL 4

In [86]:

```
# Re-build the model using 'statsmodels' after removing the feature 'temp' from Model 3

lm_4 = stats_model_building(X_train_model_4.columns)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.812
Model:                            OLS   Adj. R-squared:                  0.805
Method:                 Least Squares   F-statistic:                     122.8
Date:                Wed, 13 Apr 2022   Prob (F-statistic):          2.34e-163
Time:                        15:53:37   Log-Likelihood:                 448.48
No. Observations:                 503   AIC:                            -861.0
Df Residuals:                     485   BIC:                            -785.0
Df Model:                          17
Covariance Type:            nonrobust
======================================================================================
                         coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------------
const                  0.5423      0.033     16.237      0.000       0.477       0.608
Spring                -0.2042      0.020    -10.254      0.000      -0.243      -0.165
Summer                -0.0464      0.014     -3.394      0.001      -0.073      -0.020
Winter                -0.0162      0.016     -1.001      0.318      -0.048       0.016
Light Snow or Rain    -0.2978      0.033     -9.046      0.000      -0.363      -0.233
Misty                 -0.0726      0.012     -6.101      0.000      -0.096      -0.049
Mid Month              0.0406      0.010      3.922      0.000       0.020       0.061
Saturday               0.0905      0.028      3.227      0.001       0.035       0.146
Sunday                 0.0404      0.028      1.430      0.153      -0.015       0.096
December              -0.1526      0.019     -7.862      0.000      -0.191      -0.114
February              -0.1096      0.024     -4.526      0.000      -0.157      -0.062
January               -0.1696      0.023     -7.279      0.000      -0.215      -0.124
November              -0.1516      0.021     -7.384      0.000      -0.192      -0.111
September              0.0468      0.018      2.650      0.008       0.012       0.082
yr                     0.2432      0.009     26.703      0.000       0.225       0.261
workingday             0.0900      0.026      3.456      0.001       0.039       0.141
hum                   -0.0656      0.030     -2.187      0.029      -0.125      -0.007
windspeed             -0.1624      0.026     -6.292      0.000      -0.213      -0.112
==============================================================================
Omnibus:                       65.441   Durbin-Watson:                   1.952
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              159.293
Skew:                          -0.675   Prob(JB):                     2.57e-35
```

```
Kurtosis:                       5.404   Cond. No.                         19.9
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

Inferences from Summary statistics of Model 4 :

- **R-Squared** value for **Model 4** is **0.812**
- **Adjusted R-Squared** value for **Model 4** is **0.805**
- R-Squared and Adjusted R-Squared values **decreased** after removing the feature 'temp' from Model 3
- **Prob (F-statistic): 2.34e-163** is very low which implies that the **overall significance** is statistically good
- Features with **Positive** coefficients : **const, Mid Month, Saturday, Sunday, September, yr, workingday**
- Features with **Negative** coefficients : **Summer, Winter, Spring, Light Snow or Rain, Misty, December, February, January, July, November, hum, windspeed**
- The features **'Summer' and 'Winter'** have changed their signs of coefficients from **positive to negative**
- Features with **p-value > 0.05** (Significance level) : Winter **(0.318)**, Sunday **(0.153)**

In [87]:

```
# Calculate VIF for the predictor variables of Model 4

calculate_VIF(X_train_model_4)
```

Out[87]:

	Features	VIF
15	hum	10.28
14	workingday	9.45
16	windspeed	5.56
0	Spring	4.60
2	Winter	3.21
6	Saturday	2.82
7	Sunday	2.81
4	Misty	2.48
10	January	2.22

	Features	VIF
1	Summer	2.17
9	February	2.06
13	yr	1.99
11	November	1.89
8	December	1.70
5	Mid Month	1.67
12	September	1.36
3	Light Snow or Rain	1.31

Inference - VIF values are reduced slightly compared to previous model. Features with VIF > 5 : 'hum' - 10.28, 'workingday' - 9.45, 'hum' - 13.52, 'windspeed' - 5.56

- Since the **coefficients of 'Summer' and 'Winter' have changed signs**, Model 4 is **not stable**.
- Therefore, instead of dropping the feature 'temp' from Model 3, **drop the feature 'workingday'** which has the next highest VIF value of 15.57 and re-build the **Model 4.1** using 'statsmodels'

In [88]:

```
# Drop the feature 'workingday' from the dataframe of selected predictor variables

X_train_model_4 = X_train_model_3.drop("workingday", axis = 1)
```

In [89]:

```
# Selected features for Model 4.1

X_train_model_4.columns
```

Out[89]:

```
Index(['Spring', 'Summer', 'Winter', 'Light Snow or Rain', 'Misty',
       'Mid Month', 'Saturday', 'Sunday', 'December', 'February', 'January',
       'November', 'September', 'yr', 'temp', 'hum', 'windspeed'],
      dtype='object')
```

MODEL 4.1

In [90]:
```
# Re-build the model using 'statsmodels' after removing the feature 'workingday' from Model 3

lm_4 = stats_model_building(X_train_model_4.columns)
```

```
                            OLS Regression Results                            
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.844
Model:                            OLS   Adj. R-squared:                  0.839
Method:                 Least Squares   F-statistic:                     154.4
Date:                Wed, 13 Apr 2022   Prob (F-statistic):          3.45e-183
Time:                        15:53:37   Log-Likelihood:                 496.14
No. Observations:                 503   AIC:                            -956.3
Df Residuals:                     485   BIC:                            -880.3
Df Model:                          17                                         
Covariance Type:            nonrobust                                         
======================================================================================
                         coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------------
const                  0.3165      0.035      9.009      0.000       0.247       0.386
Spring                -0.0561      0.023     -2.465      0.014      -0.101      -0.011
Summer                 0.0412      0.015      2.755      0.006       0.012       0.070
Winter                 0.0982      0.018      5.422      0.000       0.063       0.134
Light Snow or Rain    -0.2281      0.031     -7.458      0.000      -0.288      -0.168
Misty                 -0.0488      0.011     -4.410      0.000      -0.070      -0.027
Mid Month              0.0378      0.009      4.009      0.000       0.019       0.056
Saturday               0.0124      0.012      1.044      0.297      -0.011       0.036
Sunday                -0.0392      0.012     -3.225      0.001      -0.063      -0.015
December              -0.0752      0.019     -3.935      0.000      -0.113      -0.038
February              -0.0465      0.023     -2.038      0.042      -0.091      -0.002
January               -0.0774      0.023     -3.366      0.001      -0.123      -0.032
November              -0.0887      0.020     -4.507      0.000      -0.127      -0.050
September              0.0689      0.016      4.244      0.000       0.037       0.101
yr                     0.2296      0.008     27.396      0.000       0.213       0.246
temp                   0.4183      0.039     10.753      0.000       0.342       0.495
hum                   -0.1423      0.028     -5.040      0.000      -0.198      -0.087
windspeed             -0.1548      0.023     -6.593      0.000      -0.201      -0.109
==============================================================================
Omnibus:                       88.198   Durbin-Watson:                   2.024
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              199.273
Skew:                          -0.923   Prob(JB):                     5.35e-44
Kurtosis:                       5.470   Cond. No.                         22.1
==============================================================================
```

```
Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

Inferences from Summary statistics of Model 4.1 :

- **R-Squared** value for **Model 4.1** is **0.844**
- **Adjusted R-Squared** value for **Model 4.1** is **0.839**
- R-Squared and Adjusted R-Squared values **slightly decreased** after removing the feature 'workingday' from Model 3
- **Prob (F-statistic): 3.45e-183** is very low which implies that the **overall significance** is statistically good
- Features with **Positive** coefficients : **const, Summer, Winter, Mid Month, Saturday, September, yr, workingday, temp**
- Features with **Negative** coefficients : **Spring, Sunday, Light Snow or Rain, Misty, December, February, January, July, November, hum, windspeed**
- The feature **'Sunday'** has changed the sign from **positive to negative**
- Feature with **p-value > 0.05** (Significance level) : Saturday **(0.297)**

In [91]:

```
# Calculate VIF for the predictor variables of Model 4.1

calculate_VIF(X_train_model_4)
```

Out[91]:

	Features	VIF
15	hum	13.37
14	temp	9.81
16	windspeed	5.83
0	Spring	4.85
2	Winter	3.28
4	Misty	2.69
10	January	2.31
1	Summer	2.19
13	yr	2.11
9	February	2.08
11	November	1.93

	Features	VIF
8	December	1.79
5	Mid Month	1.68
3	Light Snow or Rain	1.44
12	September	1.36
6	Saturday	1.21
7	Sunday	1.20

Inference - VIF values increased slightly compared to previous model. Features with VIF > 5 : 'hum' - 13,37, 'temp' - 9.81, 'windspeed' - 5.83

- Since the **coefficient of 'Sunday' has changed signs**, Model 4.1 is **not stable**.
- Therefore, instead of dropping the feature 'workingday' from Model 3, **drop the feature 'hum'** which has the next highest VIF value of 13.52 and re-build the **Model 4.2** using 'statsmodels'

In [92]:

```
# Drop the feature 'hum' from the dataframe of selected predictor variables

X_train_model_4 = X_train_model_3.drop("hum", axis = 1)
```

In [93]:

```
# Selected features for Model 4.2

X_train_model_4.columns
```

Out[93]:

```
Index(['Spring', 'Summer', 'Winter', 'Light Snow or Rain', 'Misty',
       'Mid Month', 'Saturday', 'Sunday', 'December', 'February', 'January',
       'November', 'September', 'yr', 'workingday', 'temp', 'windspeed'],
      dtype='object')
```

MODEL 4.2

In [94]:

```
# Re-build the model using 'statsmodels' after removing the feature 'hum' from Model 3

lm_4 = stats_model_building(X_train_model_4.columns)
```

```
                            OLS Regression Results                            
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.841
Model:                            OLS   Adj. R-squared:                  0.836
Method:                 Least Squares   F-statistic:                     151.1
Date:                Wed, 13 Apr 2022   Prob (F-statistic):          2.81e-181
Time:                        15:53:37   Log-Likelihood:                 491.55
No. Observations:                 503   AIC:                            -947.1
Df Residuals:                     485   BIC:                            -871.1
Df Model:                          17                                         
Covariance Type:            nonrobust                                         
======================================================================================
                         coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------------
const                  0.1885      0.043      4.416      0.000       0.105       0.272
Spring                -0.0705      0.023     -3.098      0.002      -0.115      -0.026
Summer                 0.0304      0.015      2.047      0.041       0.001       0.060
Winter                 0.0828      0.018      4.597      0.000       0.047       0.118
Light Snow or Rain    -0.3058      0.027    -11.321      0.000      -0.359      -0.253
Misty                 -0.0827      0.009     -9.364      0.000      -0.100      -0.065
Mid Month              0.0385      0.010      4.049      0.000       0.020       0.057
Saturday               0.1040      0.026      4.033      0.000       0.053       0.155
Sunday                 0.0481      0.026      1.854      0.064      -0.003       0.099
December              -0.0847      0.019     -4.422      0.000      -0.122      -0.047
February              -0.0437      0.023     -1.900      0.058      -0.089       0.002
January               -0.0786      0.023     -3.387      0.001      -0.124      -0.033
November              -0.0862      0.020     -4.329      0.000      -0.125      -0.047
September              0.0563      0.016      3.503      0.001       0.025       0.088
yr                     0.2350      0.008     28.050      0.000       0.219       0.252
workingday             0.0962      0.024      4.021      0.000       0.049       0.143
temp                   0.3726      0.038      9.812      0.000       0.298       0.447
windspeed             -0.1196      0.023     -5.224      0.000      -0.165      -0.075
==============================================================================
Omnibus:                       79.006   Durbin-Watson:                   1.943
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              204.910
Skew:                          -0.785   Prob(JB):                     3.19e-45
Kurtosis:                       5.704   Cond. No.                         25.0
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

Inferences from Summary statistics of Model 4.2 :

- **R-Squared** value for **Model 4.2** is **0.841**
- **Adjusted R-Squared** value for **Model 4.2** is **0.836**
- R-Squared and Adjusted R-Squared values **slightly decreased** after removing the feature 'hum' from Model 3
- **Prob (F-statistic): 2.81e-181** is very low which implies that the **overall significance** is statistically good
- Features with **Positive** coefficients : **const, Summer, Winter, Mid Month, Saturday, Sunday, September, yr, workingday, temp**
- Features with **Negative** coefficients : **Spring, Light Snow or Rain, Misty, December, February, January, November, windspeed**
- No change in signs of coefficients compared to Model 3
- Features with **p-value > 0.05** (Significance level) : Sunday **(0.064)**, February **(0.058)**

In [95]:

```
# Calculate VIF for the predictor variables of Model 4.2

calculate_VIF(X_train_model_4)
```

Out[95]:

	Features	VIF
14	workingday	15.39
15	temp	12.28
16	windspeed	5.73
0	Spring	5.28
6	Saturday	3.96
7	Sunday	3.83
2	Winter	3.52
10	January	2.33
1	Summer	2.26
9	February	2.11
13	yr	2.07
11	November	1.94
8	December	1.79
5	Mid Month	1.68

	Features	VIF
4	Misty	1.67
12	September	1.30
3	Light Snow or Rain	1.10

Inference - VIF values increased slightly compared to previous model. Features with VIF > 5 : 'workingday' - 15.39, 'temp' - 12.28, 'windspeed' - 5.73, 'Spring' - 5.28

- **p - value** of the feature **Sunday = 0.064** which is greater than the significance level of **alpha = 0.05**
- The feature **'workingday'** has the highest **VIF** of 15.39
- When both the **'p - value' and 'VIF'** are higher than the tolerance level, improvement in **significance** is given priority
- So drop the feature **'Sunday'** and re-build the **Model 5** using 'statsmodels'

In [96]:
```
# Drop the feature 'Sunday' from the dataframe of selected predictor variables

X_train_model_5 = X_train_model_4.drop("Sunday", axis = 1)
```

In [97]:
```
# Selected features for Model 5

X_train_model_5.columns
```

Out[97]:
```
Index(['Spring', 'Summer', 'Winter', 'Light Snow or Rain', 'Misty',
       'Mid Month', 'Saturday', 'December', 'February', 'January', 'November',
       'September', 'yr', 'workingday', 'temp', 'windspeed'],
      dtype='object')
```

MODEL 5

In [98]:
```
# Re-build the model using 'statsmodels' after removing the feature 'Sunday' from Model 4.2

lm_5 = stats_model_building(X_train_model_5.columns)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.840
```

```
Model:                            OLS   Adj. R-squared:                  0.835
Method:                 Least Squares   F-statistic:                     159.5
Date:                Wed, 13 Apr 2022   Prob (F-statistic):          1.20e-181
Time:                        15:53:37   Log-Likelihood:                 489.77
No. Observations:                 503   AIC:                            -945.5
Df Residuals:                     486   BIC:                            -873.8
Df Model:                          16
Covariance Type:            nonrobust
=======================================================================================
                          coef    std err          t      P>|t|      [0.025      0.975]
---------------------------------------------------------------------------------------
const                   0.2312      0.036      6.416      0.000       0.160       0.302
Spring                 -0.0718      0.023     -3.150      0.002      -0.117      -0.027
Summer                  0.0293      0.015      1.967      0.050    3.24e-05       0.058
Winter                  0.0825      0.018      4.569      0.000       0.047       0.118
Light Snow or Rain     -0.3051      0.027    -11.270      0.000      -0.358      -0.252
Misty                  -0.0827      0.009     -9.332      0.000      -0.100      -0.065
Mid Month               0.0387      0.010      4.056      0.000       0.020       0.057
Saturday                0.0650      0.015      4.352      0.000       0.036       0.094
December               -0.0854      0.019     -4.451      0.000      -0.123      -0.048
February               -0.0440      0.023     -1.907      0.057      -0.089       0.001
January                -0.0815      0.023     -3.509      0.000      -0.127      -0.036
November               -0.0892      0.020     -4.483      0.000      -0.128      -0.050
September               0.0546      0.016      3.393      0.001       0.023       0.086
yr                      0.2348      0.008     27.958      0.000       0.218       0.251
workingday              0.0570      0.011      5.050      0.000       0.035       0.079
temp                    0.3698      0.038      9.722      0.000       0.295       0.445
windspeed              -0.1212      0.023     -5.285      0.000      -0.166      -0.076
==============================================================================
Omnibus:                       81.186   Durbin-Watson:                   1.961
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              203.163
Skew:                          -0.818   Prob(JB):                     7.65e-45
Kurtosis:                       5.648   Cond. No.                         22.9
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

Inferences from Summary statistics of Model 5 :

- **R-Squared** value for **Model 5** is **0.840**
- **Adjusted R-Squared** value for **Model 5** is **0.835**
- R-Squared and Adjusted R-Squared values **slightly decreased** after removing the feature 'Sunday' from Model 4.2

- **Prob (F-statistic): 1.20e-181** is very low which implies that the **overall significance** is statistically good
- Features with **Positive** coefficients : **const, Summer, Winter, Mid Month, Saturday, September, yr, workingday, temp**
- Features with **Negative** coefficients : **Spring, Light Snow or Rain, Misty, December, February, January, November, windspeed**
- No change in signs of coefficients compared to Model 4.2
- Feature with **p-value > 0.05** (Significance level) : February **(0.057)**

In [99]:

```
# Calculate VIF for the predictor variables of Model 5

calculate_VIF(X_train_model_5)
```

Out[99]:

	Features	VIF
14	temp	6.39
15	windspeed	5.57
13	workingday	4.78
0	Spring	4.65
2	Winter	3.03
9	January	2.28
8	February	2.07
12	yr	2.07
1	Summer	2.00
10	November	1.92
6	Saturday	1.79
7	December	1.73
5	Mid Month	1.68
4	Misty	1.66
11	September	1.29
3	Light Snow or Rain	1.10

Inference - VIF values decreased compared to previous model. Features with VIF > 5 : 'temp' - 6.39, 'windspeed' - 5.57

- **p - value** of the feature **February = 0.057** which is greater than the significance level of **alpha = 0.05**
- The feature **'temp'** has the highest **VIF** of 6.39
- When both the **'p - value' and 'VIF'** are higher than the tolerance level, improvement in **significance** is given priority
- So drop the feature **'February'** and re-build the **Model 6** using 'statsmodels'

In [100…

```
# Drop the feature 'February' from the dataframe of selected predictor variables

X_train_model_6 = X_train_model_5.drop("February", axis = 1)
```

In [101…

```
# Selected features for Model 6

X_train_model_6.columns
```

Out[101…

```
Index(['Spring', 'Summer', 'Winter', 'Light Snow or Rain', 'Misty',
       'Mid Month', 'Saturday', 'December', 'January', 'November', 'September',
       'yr', 'workingday', 'temp', 'windspeed'],
      dtype='object')
```

MODEL 6

In [102…

```
# Re-build the model using 'statsmodels' after removing the feature 'February' from Model 5

lm_6 = stats_model_building(X_train_model_6.columns)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.839
Model:                            OLS   Adj. R-squared:                  0.834
Method:                 Least Squares   F-statistic:                     169.0
Date:                Wed, 13 Apr 2022   Prob (F-statistic):          5.51e-182
Time:                        15:53:37   Log-Likelihood:                 487.90
No. Observations:                 503   AIC:                            -943.8
Df Residuals:                     487   BIC:                            -876.3
Df Model:                          15
Covariance Type:            nonrobust
================================================================================
                       coef    std err          t      P>|t|      [0.025      0.975]
```

```
------------------------------------------------------------------------------------------
const                  0.2136      0.035      6.116      0.000       0.145       0.282
Spring                -0.0854      0.022     -3.936      0.000      -0.128      -0.043
Summer                 0.0336      0.015      2.283      0.023       0.005       0.063
Winter                 0.0846      0.018      4.682      0.000       0.049       0.120
Light Snow or Rain    -0.3057      0.027    -11.260      0.000      -0.359      -0.252
Misty                 -0.0825      0.009     -9.285      0.000      -0.100      -0.065
Mid Month              0.0377      0.010      3.953      0.000       0.019       0.056
Saturday               0.0665      0.015      4.450      0.000       0.037       0.096
December              -0.0718      0.018     -4.020      0.000      -0.107      -0.037
January               -0.0560      0.019     -2.940      0.003      -0.093      -0.019
November              -0.0820      0.020     -4.187      0.000      -0.120      -0.043
September              0.0565      0.016      3.511      0.000       0.025       0.088
yr                     0.2340      0.008     27.820      0.000       0.217       0.251
workingday             0.0577      0.011      5.092      0.000       0.035       0.080
temp                   0.3896      0.037     10.616      0.000       0.317       0.462
windspeed             -0.1185      0.023     -5.163      0.000      -0.164      -0.073
==============================================================================
Omnibus:                       74.850   Durbin-Watson:                   1.963
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              183.204
Skew:                          -0.765   Prob(JB):                     1.65e-40
Kurtosis:                       5.530   Cond. No.                         22.1
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

Inferences from Summary statistics of Model 6 :

- **R-Squared** value for **Model 6** is **0.839**
- **Adjusted R-Squared** value for **Model 6** is **0.834**
- R-Squared and Adjusted R-Squared values **slightly decreased** after removing the feature 'February' from Model 5
- **Prob (F-statistic): 5.51e-182** is very low which implies that the **overall significance** is statistically good
- Features with **Positive** coefficients : **const, Summer, Winter, Mid Month, Saturday, September, yr, workingday, temp**
- Features with **Negative** coefficients : **Spring, Light Snow or Rain, Misty, December, January, November, windspeed**
- No change in signs of coefficients compared to Model 5
- No Features with **p-value > 0.05** (Significance level)

In [103…

```
# Calculate VIF for the predictor variables of Model 6
```

```
calculate_VIF(X_train_model_6)
```

Out[103…

	Features	VIF
13	temp	6.33
14	windspeed	5.56
12	workingday	4.78
0	Spring	2.98
2	Winter	2.98
11	yr	2.07
1	Summer	2.00
9	November	1.89
6	Saturday	1.79
5	Mid Month	1.67
4	Misty	1.66
8	January	1.63
7	December	1.56
10	September	1.29
3	Light Snow or Rain	1.10

Inference - VIF values slightly decreased compared to previous model. Features with VIF > 5 : 'temp' - 6.33, 'windspeed' - 5.56

- No **p - values** of the features are greater than the significance level of **alpha = 0.05**
- The feature **'temp'** has the highest **VIF** of 6.33
- Thus, in order to improve the VIF values and **resolve the multicollinearity** issue, let us drop the feature **'temp'** and re-build the **Model 7** using 'statsmodels'

In [104…

```
# Drop the feature 'temp' from the dataframe of selected predictor variables
```

```
X_train_model_7 = X_train_model_6.drop("temp", axis = 1)
```

In [105…

```
# Selected features for Model 7

X_train_model_7.columns
```

Out[105…

```
Index(['Spring', 'Summer', 'Winter', 'Light Snow or Rain', 'Misty',
       'Mid Month', 'Saturday', 'December', 'January', 'November', 'September',
       'yr', 'workingday', 'windspeed'],
      dtype='object')
```

MODEL 7

In [106…

```
# Re-build the model using 'statsmodels' after removing the feature 'temp' from Model 6

lm_7 = stats_model_building(X_train_model_7.columns)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.802
Model:                            OLS   Adj. R-squared:                  0.796
Method:                 Least Squares   F-statistic:                     140.8
Date:                Wed, 13 Apr 2022   Prob (F-statistic):          3.52e-161
Time:                        15:53:37   Log-Likelihood:                 435.54
No. Observations:                 503   AIC:                            -841.1
Df Residuals:                     488   BIC:                            -777.8
Df Model:                          14
Covariance Type:            nonrobust
======================================================================================
                         coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------------
const                  0.5389      0.019     29.035      0.000       0.502       0.575
Spring                -0.2570      0.016    -15.999      0.000      -0.289      -0.225
Summer                -0.0475      0.014     -3.402      0.001      -0.075      -0.020
Winter                -0.0252      0.016     -1.534      0.126      -0.057       0.007
Light Snow or Rain    -0.3349      0.030    -11.187      0.000      -0.394      -0.276
Misty                 -0.0881      0.010     -8.966      0.000      -0.107      -0.069
Mid Month              0.0390      0.011      3.688      0.000       0.018       0.060
Saturday               0.0629      0.017      3.794      0.000       0.030       0.095
December              -0.1283      0.019     -6.786      0.000      -0.165      -0.091
January               -0.1157      0.020     -5.735      0.000      -0.155      -0.076
```

```
November              -0.1416      0.021     -6.813      0.000      -0.182      -0.101
September              0.0419      0.018      2.358      0.019       0.007       0.077
yr                     0.2441      0.009     26.358      0.000       0.226       0.262
workingday             0.0604      0.013      4.816      0.000       0.036       0.085
windspeed             -0.1443      0.025     -5.702      0.000      -0.194      -0.095
==============================================================================
Omnibus:                       50.717   Durbin-Watson:                   1.946
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              127.085
Skew:                          -0.516   Prob(JB):                     2.53e-28
Kurtosis:                       5.236   Cond. No.                         10.7
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

Inferences from Summary statistics of Model 7 :

- **R-Squared** value for **Model 7** is **0.802**
- **Adjusted R-Squared** value for **Model 7** is **0.796**
- R-Squared and Adjusted R-Squared values **decreased** after removing the feature 'temp' from Model 6
- **Prob (F-statistic): 3.52e-161** is very low which implies that the **overall significance** is statistically good
- Features with **Positive** coefficients : **const, Mid Month, Saturday, September, yr, workingday, temp**
- Features with **Negative** coefficients : **Spring, Summer, Winter, Light Snow or Rain, Misty, December, January, November, windspeed**
- The features **'Summer' and 'Winter'** have changed their signs of coefficients from **positive to negative**
- Feature with **p-value > 0.05** (Significance level) : Winter **(0.126)**

In [107…

```
# Calculate VIF for the predictor variables of Model 7

calculate_VIF(X_train_model_7)
```

Out[107…

	Features	VIF
13	windspeed	4.44
12	workingday	3.49
2	Winter	2.96
0	Spring	2.70
1	Summer	2.00

	Features	VIF
11	yr	1.91
9	November	1.86
4	Misty	1.64
5	Mid Month	1.64
8	January	1.60
6	Saturday	1.57
7	December	1.54
10	September	1.25
3	Light Snow or Rain	1.09

Inference - No feature with VIF > 5. The feature 'windspeed' has the highest VIF of 4.44 in Model 7.

- Since the **coefficients of 'Summer' and 'Winter' have changed signs**, Model 7 is **not stable**.
- Therefore, instead of dropping the feature 'temp' from Model 6, **drop the feature 'windspeed'** which has the next highest VIF value of 5.56 and re-build the **Model 7.1** using 'statsmodels'

In [108…

```
# Drop the feature 'windspeed' from the dataframe of selected predictor variables

X_train_model_7 = X_train_model_6.drop("windspeed", axis = 1)
```

In [109…

```
# Selected features for Model 7.1

X_train_model_7.columns
```

Out[109…

```
Index(['Spring', 'Summer', 'Winter', 'Light Snow or Rain', 'Misty',
       'Mid Month', 'Saturday', 'December', 'January', 'November', 'September',
       'yr', 'workingday', 'temp'],
      dtype='object')
```

MODEL 7.1

In [110…

```
# Re-build the model using 'statsmodels' after removing the feature 'windspeed' from Model 6

lm_7 = stats_model_building(X_train_model_7.columns)
```

```
                            OLS Regression Results                            
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.830
Model:                            OLS   Adj. R-squared:                  0.825
Method:                 Least Squares   F-statistic:                     170.2
Date:                Wed, 13 Apr 2022   Prob (F-statistic):          1.67e-177
Time:                        15:53:37   Log-Likelihood:                 474.49
No. Observations:                 503   AIC:                            -919.0
Df Residuals:                     488   BIC:                            -855.7
Df Model:                          14                                         
Covariance Type:            nonrobust                                         
======================================================================================
                         coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------------
const                  0.1504      0.034      4.483      0.000       0.085       0.216
Spring                -0.0873      0.022     -3.920      0.000      -0.131      -0.044
Summer                 0.0291      0.015      1.930      0.054      -0.001       0.059
Winter                 0.0919      0.018      4.973      0.000       0.056       0.128
Light Snow or Rain    -0.3234      0.028    -11.706      0.000      -0.378      -0.269
Misty                 -0.0803      0.009     -8.826      0.000      -0.098      -0.062
Mid Month              0.0398      0.010      4.069      0.000       0.021       0.059
Saturday               0.0654      0.015      4.262      0.000       0.035       0.096
December              -0.0663      0.018     -3.626      0.000      -0.102      -0.030
January               -0.0514      0.020     -2.634      0.009      -0.090      -0.013
November              -0.0841      0.020     -4.188      0.000      -0.124      -0.045
September              0.0606      0.017      3.671      0.000       0.028       0.093
yr                     0.2336      0.009     27.074      0.000       0.217       0.251
workingday             0.0592      0.012      5.095      0.000       0.036       0.082
temp                   0.4096      0.037     10.941      0.000       0.336       0.483
==============================================================================
Omnibus:                       76.519   Durbin-Watson:                   1.998
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              211.539
Skew:                          -0.740   Prob(JB):                     1.16e-46
Kurtosis:                       5.811   Cond. No.                         21.0
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

Inferences from Summary statistics of Model 7.1 :

- **R-Squared** value for **Model 7.1** is **0.830**
- **Adjusted R-Squared** value for **Model 7.1** is **0.825**
- R-Squared and Adjusted R-Squared values **slightly decreased** after removing the feature 'windspeed' from Model 6
- **Prob (F-statistic): 1.67e-177** is very low which implies that the **overall significance** is statistically good
- Features with **Positive** coefficients : **const, Summer, Winter, Mid Month, Saturday, September, yr, workingday, temp**
- Features with **Negative** coefficients : **Spring, Light Snow or Rain, Misty, December, January, November**
- No change in signs of coefficients compared to Model 6
- Feature with **p-value > 0.05** (Significance level) : Summer **(0.054)**

In [111…

```
# Calculate VIF for the predictor variables of Model 7.1

calculate_VIF(X_train_model_7)
```

Out[111…

	Features	VIF
13	temp	5.06
12	workingday	4.76
2	Winter	2.88
0	Spring	2.42
11	yr	2.07
9	November	1.87
6	Saturday	1.78
1	Summer	1.76
5	Mid Month	1.67
4	Misty	1.66
8	January	1.62
7	December	1.56
10	September	1.29

	Features	VIF
3	Light Snow or Rain	1.07

Inference - VIF values are decreased compared to Model 6. Feature with VIF > 5 : 'temp' - 5.06

- **p - value** of the feature **Summer = 0.054** which is greater than the significance level of **alpha = 0.05**
- The feature **'temp'** has the highest **VIF** of 5.06
- When both the **'p - value' and 'VIF'** are higher than the tolerance level, improvement in **significance** is given priority
- So drop the feature **'Summer'** and re-build the **Model 8** using 'statsmodels'

In [112…

```
# Drop the feature 'Summer' from the dataframe of selected predictor variables

X_train_model_8 = X_train_model_7.drop("Summer", axis = 1)
```

In [113…

```
# Selected features for Model 8

X_train_model_8.columns
```

Out[113…

```
Index(['Spring', 'Winter', 'Light Snow or Rain', 'Misty', 'Mid Month',
       'Saturday', 'December', 'January', 'November', 'September', 'yr',
       'workingday', 'temp'],
      dtype='object')
```

MODEL 8

In [114…

```
# Re-build the model using 'statsmodels' after removing the feature 'Summer' from Model 7.1

lm_8 = stats_model_building(X_train_model_8.columns)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.829
Model:                            OLS   Adj. R-squared:                  0.824
Method:                 Least Squares   F-statistic:                     182.0
Date:                Wed, 13 Apr 2022   Prob (F-statistic):          7.70e-178
Time:                        15:53:38   Log-Likelihood:                 472.58
```

```
No. Observations:                 503   AIC:                            -917.2
Df Residuals:                     489   BIC:                            -858.1
Df Model:                          13
Covariance Type:            nonrobust
=======================================================================================
                          coef    std err          t      P>|t|      [0.025      0.975]
---------------------------------------------------------------------------------------
const                   0.1925      0.026      7.524      0.000       0.142       0.243
Spring                 -0.1155      0.017     -6.864      0.000      -0.149      -0.082
Winter                  0.0711      0.015      4.724      0.000       0.042       0.101
Light Snow or Rain     -0.3253      0.028    -11.749      0.000      -0.380      -0.271
Misty                  -0.0791      0.009     -8.689      0.000      -0.097      -0.061
Mid Month               0.0404      0.010      4.119      0.000       0.021       0.060
Saturday                0.0644      0.015      4.189      0.000       0.034       0.095
December               -0.0731      0.018     -4.059      0.000      -0.108      -0.038
January                -0.0575      0.019     -2.978      0.003      -0.095      -0.020
November               -0.0914      0.020     -4.619      0.000      -0.130      -0.052
September               0.0500      0.016      3.202      0.001       0.019       0.081
yr                      0.2343      0.009     27.106      0.000       0.217       0.251
workingday              0.0590      0.012      5.065      0.000       0.036       0.082
temp                    0.3714      0.032     11.657      0.000       0.309       0.434
==============================================================================
Omnibus:                       76.395   Durbin-Watson:                   1.993
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              217.730
Skew:                          -0.729   Prob(JB):                     5.25e-48
Kurtosis:                       5.874   Cond. No.                         15.9
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

Inferences from Summary statistics of Model 8 :

- **R-Squared** value for **Model 8** is **0.829**
- **Adjusted R-Squared** value for **Model 8** is **0.824**
- R-Squared and Adjusted R-Squared values **slightly decreased** after removing the feature 'Summer' from Model 7.1
- **Prob (F-statistic): 7.70e-178** is very low which implies that the **overall significance** is statistically good
- Features with **Positive** coefficients : **const, Winter, Mid Month, Saturday, September, yr, workingday, temp**
- Features with **Negative** coefficients : **Spring, Light Snow or Rain, Misty, December, January, November**
- No change in signs of coefficients compared to Model 7.1
- No Features with **p-value > 0.05** (Significance level)

In [115…

```
# Calculate VIF for the predictor variables of Model 8

calculate_VIF(X_train_model_8)
```

Out[115…

	Features	VIF
12	temp	4.90
11	workingday	4.54
1	Winter	2.63
0	Spring	2.19
10	yr	2.05
8	November	1.87
5	Saturday	1.73
4	Mid Month	1.66
7	January	1.62
3	Misty	1.60
6	December	1.56
9	September	1.19
2	Light Snow or Rain	1.06

Inference - No feature with VIF > 5. The feature 'temp' has the highest VIF of 4.90 in Model 8.

- No **p - values** of the features are greater than the significance level of **alpha = 0.05**. Infact, p- values of all the features are around **0.000**.
- For Model 8, **VIF** values of all the predictors are also **less than 5**.
- **R - Squared (0.829) and Adjusted R - Squared (0.824)** values are also **above 80%**, which is a good score for a linear regression model.
- Prob (F-statistic): **7.70e-178** is very low, which implies that the **overall significance** is good. It also means that the model fit is **statistically significant** and the explained variance isn't purely by chance.

In [116…

```
# Build the final linear regression model using Scikit Learn
```

```
lr = LinearRegression()
LinearRegressionModel = lr.fit(X_train_model_8, y_train)
print("Intercept value of the built model : \t",lr.intercept_,"\n")
j=0
columns = X_train_model_8.columns
for i in columns:
    print("Coefficient of the feature -",i," : ",lr.coef_[j])
    j+=1
```

```
Intercept value of the built model :     0.1925153799731144

Coefficient of the feature - Spring  :  -0.11549906427979702
Coefficient of the feature - Winter  :  0.07107811036970689
Coefficient of the feature - Light Snow or Rain  :  -0.32528257723638254
Coefficient of the feature - Misty  :  -0.07913116256968115
Coefficient of the feature - Mid Month  :  0.04039813583014747
Coefficient of the feature - Saturday  :  0.06440997256785737
Coefficient of the feature - December  :  -0.07308927186144788
Coefficient of the feature - January  :  -0.05750498734447426
Coefficient of the feature - November  :  -0.09136891118991015
Coefficient of the feature - September  :  0.049971247277551374
Coefficient of the feature - yr  :  0.2343096537075494
Coefficient of the feature - workingday  :  0.058977436299761675
Coefficient of the feature - temp  :  0.3714075618040843
```

e. Residual Analysis of the train dataset

To check whether the assumptions made before building the model hold TRUE, let us plot the histogram of error terms

In [117…

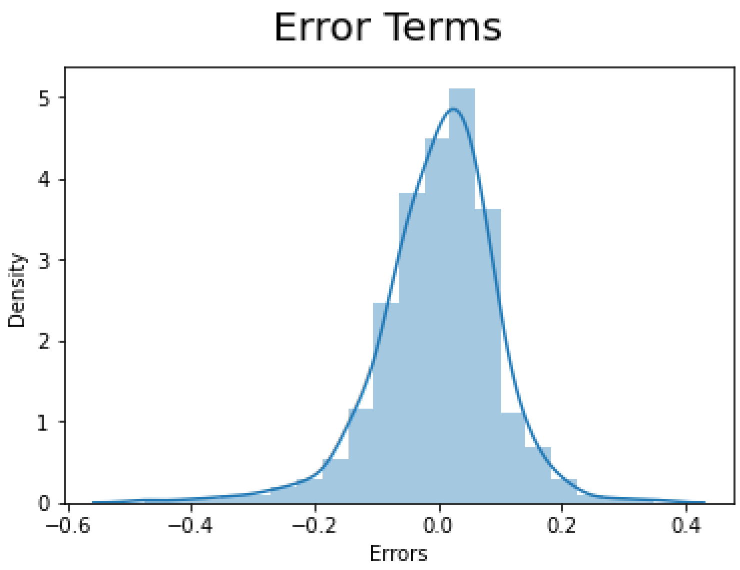
```
# Use the built linear regression model on the train set and predict y- values ('cnt')

y_train_pred = lr.predict(X_train_model_8)
```

In [118…

```
# To check whether the error terms are normally distributed, plot the histogram of error terms

fig = plt.figure()
sns.distplot((y_train - y_train_pred), bins = 20)
fig.suptitle('Error Terms', fontsize = 20)
plt.xlabel('Errors', fontsize = 10)
plt.show()
```



Inferences : Error terms are normally distributed with mean 0.0

f. Making Predictions Using the Final Model (MODEL 8)

Now that we have fitted the model and checked the normality of error terms, Let us make predictions using the final model, i.e. MODEL 8. Apply the same pre-processing steps for test set as we did for train set.

In [119…

```
# Rescale features of test set
# Apply scaler() to the columns 'temp','hum','windspeed','cnt'

scale_vars = ['temp','hum','windspeed','cnt']

df_test[scale_vars] = scaler.transform(df_test[scale_vars])
```

In [120…

```
# First 5 records of the test set after scaling
# All values of the test set now ranges from 0 to 1

df_test.head()
```

Out[120…

	Spring	Summer	Winter	Light Snow or Rain	Misty	End of Month	Mid Month	Monday	Saturday	Sunday	...	November	October	September	yr	holiday	workingday
202	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0	0	1
497	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	1.0	...	0.0	0.0	0.0	1	0	0
370	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	1	0	1
630	0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	...	0.0	0.0	1.0	1	0	0
646	0.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	...	0.0	1.0	0.0	1	0	1

5 rows × 31 columns

Defining X_test and y_test

In [121…

```
y_test = df_test.pop('cnt')
X_test = df_test
```

In [122…

```
# Add only the features selected in Model 8 to the predictor dataframe of test set

X_test_new = X_test[X_train_model_8.columns]
```

In [123…

```
# Adding constant variable to X_test

X_test_new = sm.add_constant(X_test_new)
```

In [124…

```
# Making predictions using MODEL 8

y_test_pred = lm_8.predict(X_test_new)
```

5. MODEL EVALUATION

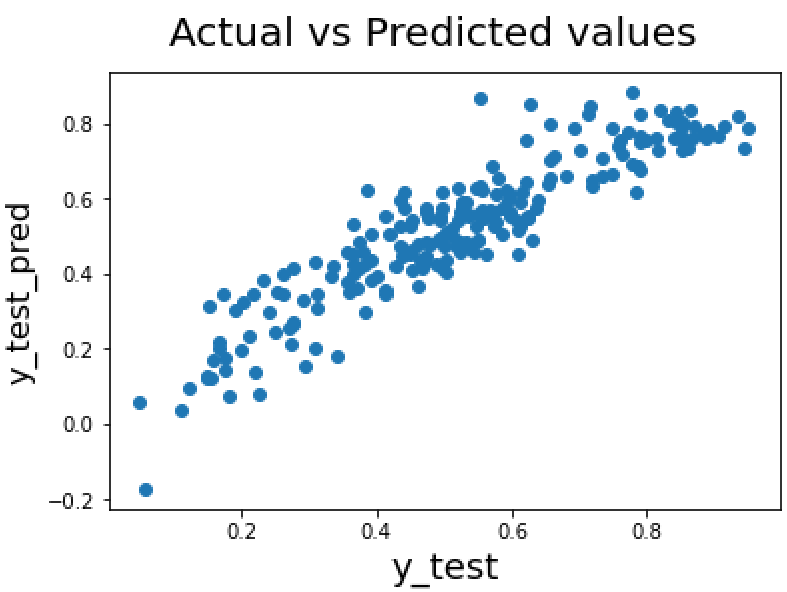
In [125…

```
# Plotting y_test and y_test_pred to understand the spread

fig = plt.figure()
plt.scatter(y_test, y_test_pred)
fig.suptitle('Actual vs Predicted values', fontsize = 20)
plt.xlabel('y_test', fontsize = 18)
plt.ylabel('y_test_pred', fontsize = 16)
```

Out[125…

```
Text(0, 0.5, 'y_test_pred')
```



Inference from the scatter plot of 'Actual vs Predicted values' of target variable 'cnt':

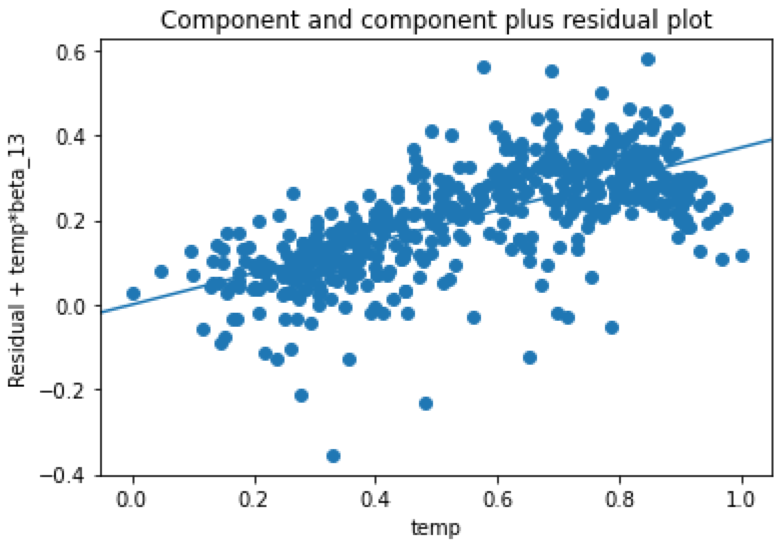
It is evident that there's a **strong correlation** between the model's predictions and its actual results, which means the model is much more **accurate**.

Validation of ASSUMPTION 1 - Linearity Check

In [126…

```
# Component and component plus residual plot

sm.graphics.plot_ccpr(lm_8, 'temp')
plt.show()
```



Inference :

The plot of temp with its corresponding predicted coefficient for demand shows that the predictor and the target variable are linearly related **[Linear relationship between X and y - ASSUMPTION No.1 VALIDATED]**

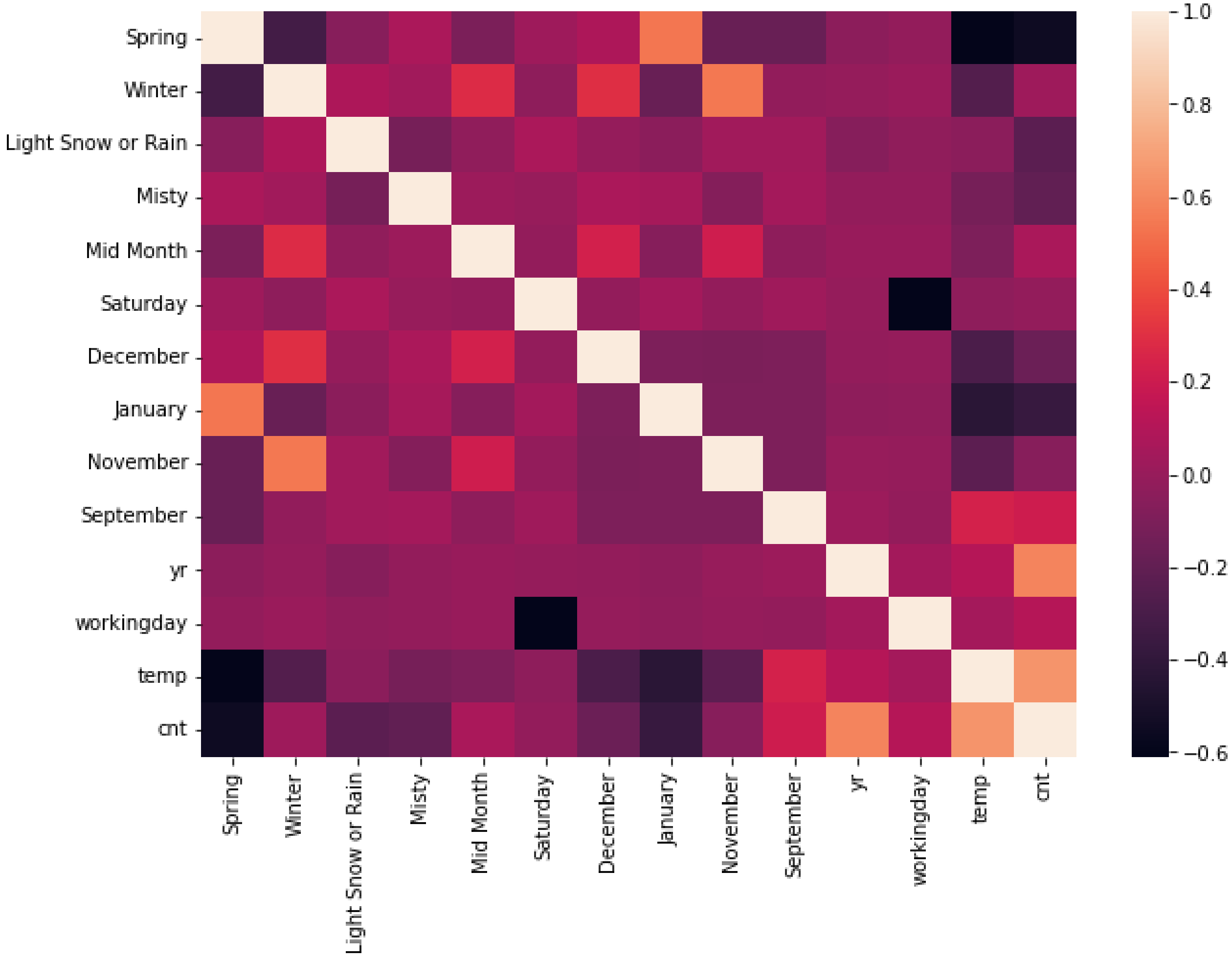
In [127…

```
df = pd.concat([X_train_model_8,BikeSharing['cnt']], axis = 1)
```

In [128…

```
# Analysing the correlation of predictors from final model with the target variable using heatmap

plt.figure(figsize = (10,7))
sns.heatmap(df.corr())
plt.show()
```



Inference from the heatmap of predictors from the final model and the target variable :

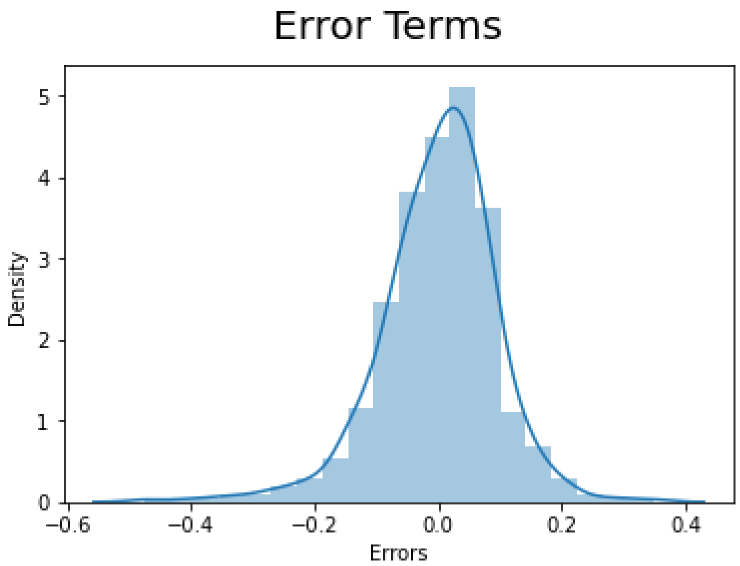
Heatmap shows that the target variable 'cnt' is correlated with the predictors from the final model either positively or negatively **[ASSUMPTION No.1 VALIDATED]**

Validation of ASSUMPTION 2 - Error terms are normally distributed

In [129…

```
# To check whether the error terms are normally distributed, plot the histogram of error terms
```

```
fig = plt.figure()
sns.distplot((y_train - y_train_pred), bins = 20)
fig.suptitle('Error Terms', fontsize = 20)
plt.xlabel('Errors', fontsize = 10)
plt.show()
```



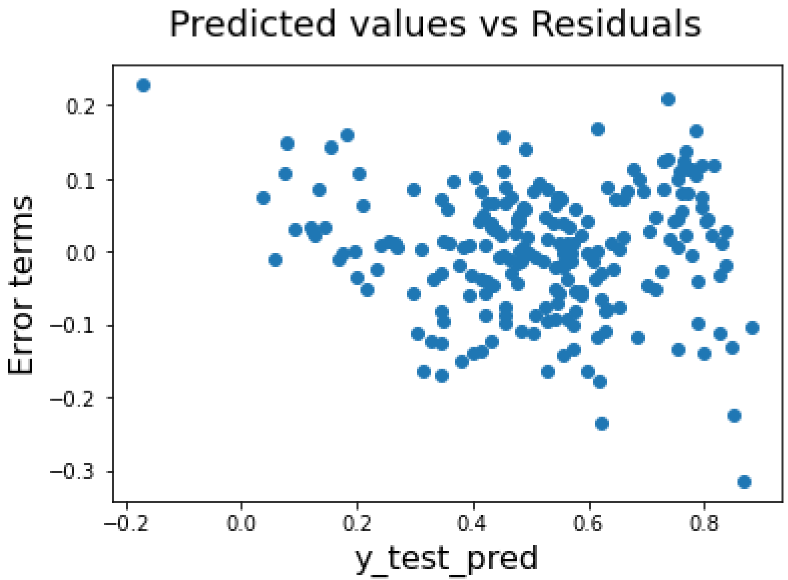
Inferences : Error terms are normally distributed with mean 0.0 [ASSUMPTION No.2 - VALIDATED]

Validation of ASSUMPTION 3 - Error terms are independent to each other

In [130…

```
# Scatter plot for 'y_test_pred' vs 'errors'
# Residual = Actual value - Predicted value

res = y_test-y_test_pred
fig = plt.figure()
plt.scatter(y_test_pred,res)
fig.suptitle('Predicted values vs Residuals', fontsize = 18)
plt.xlabel('y_test_pred', fontsize = 16)
plt.ylabel('Error terms', fontsize = 16)
plt.show()
```



Inference from the scatter plot of 'Predicted values vs Residuals' :

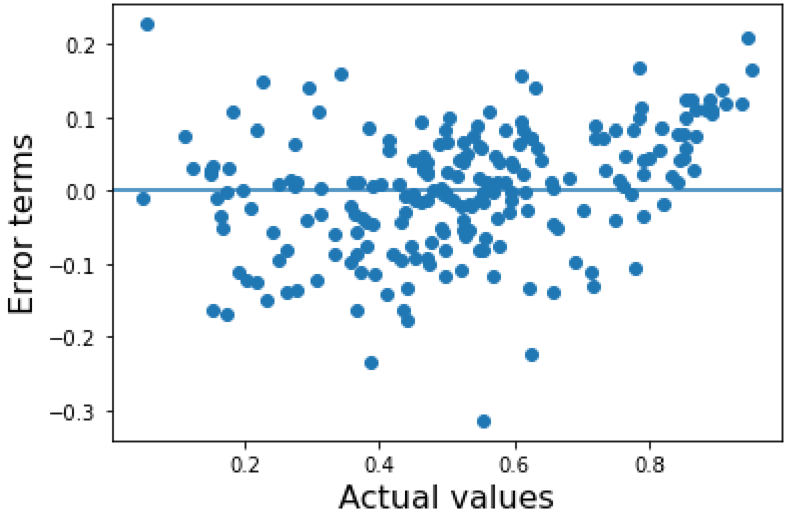
No specific patterns observed from the scatter plot of 'Predicted value' vs 'Residuals'. Thus, it shows that **the error terms are independent** to each other with respect to predicted values **[ASSUMPTION No.3 - VALIDATED]**

Validation of ASSUMPTION 4 - Homoscedasticity

In [131…

```
# Scatter plot for Actual values in test set vs residuals
# This is plotted to prove 'homoscedasticity', which is one of the assumptions made for building linear regression model

residuals = y_test - y_test_pred
plt.scatter(x=y_test, y=residuals)
plt.axhline(0)
fig.suptitle('Actual values vs Residuals', fontsize = 18)
plt.xlabel('Actual values', fontsize = 16)
plt.ylabel('Error terms', fontsize = 16)
plt.show()
```



Inference from the scatter plot of 'Actual values vs Residuals' :

The plot **doesnot show any specific cone or wedge shape**. Hence, it is clear that the variance of the error terms remain constant along the values of the dependent variable; i.e.; **homoscedasticity** exists **[ASSUMPTION No. 4 - VALIDATED]**

R - Squared

R-squared (R2) is a statistical measure that represents the proportion of the variance for a dependent variable that's explained by independent variables in a regression model. **R-squared** is a **relative measure** of fit.

In [132…

```
# Find out the R squared values of actual predicted values of train set and test set

print("R - squared value for train set : ",r2_score(y_train,y_train_pred))
print("R - squared value for test set : ",r2_score(y_test,y_test_pred))
```

```
R - squared value for train set :  0.8287460725636523
R - squared value for test set :  0.8305715157778124
```

Inference - R - squared values of the test set (83.05%) and train set (82.87%) are significantly good

Root Mean Squared Error (RMSE)

Root Mean Squared Error (RMSE) indicates the absolute fit of the model to the data – **how close the observed data points are to the model's predicted values. RMSE** is an **absolute measure** of fit. Lower values of RMSE indicate better fit. RMSE is a good measure of how accurately the model predicts the response.

In [133…

```
# Root Mean Squared Error (RMSE)

print("Root Mean Squared Error of the model is given by : ",np.sqrt(mean_squared_error(y_test, y_test_pred)))
```

```
Root Mean Squared Error of the model is given by :  0.08508689737842606
```

Inference - Since RMSE of our model is very low, the model fit is very good in predicting the response

Plot for Error terms

In [134…

```
# Function for plotting error terms

def plot_err_terms(actual,predicted):
    c = [i for i in range(1,len(y_test)+1,1)]
    fig = plt.figure(figsize=(10,5))
    plt.plot(c,actual-predicted, color="red", marker='o', linewidth=1, linestyle="")
    fig.suptitle('Error Terms', fontsize=18)
    plt.xlabel('Index', fontsize=16)
    plt.ylabel('Actual demand - Predicted demand', fontsize=16)
```

In [135…

```
# Plot error terms for test data

plot_err_terms(y_test,y_test_pred)
```

Inference - No pattern observed which means the output is explained well by the model.

Plot for Actual and Predicted demand of test set

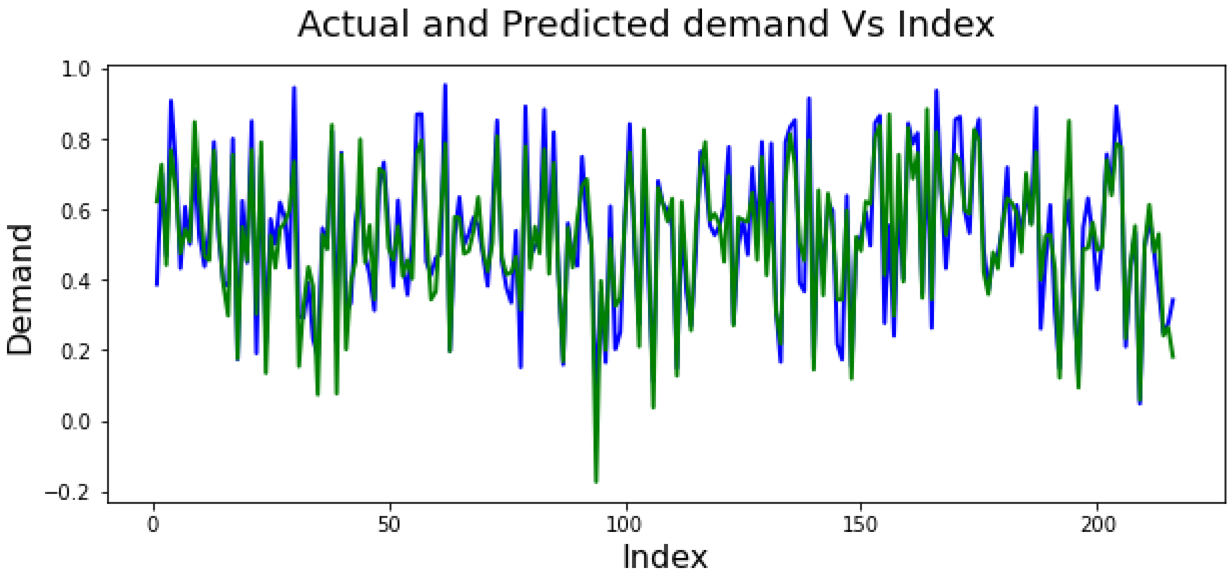
In [136…

```
# Function to plot Actual vs Predicted
# Plot of Actual and predicted values with respect to index

def plot_actual_predicted(actual,predicted):
    c = [i for i in range(1,len(y_test)+1,1)]
    fig = plt.figure(figsize=(10,4))
    plt.plot(c,actual, color="blue", linewidth=2, linestyle="-")
    plt.plot(c,predicted, color="green",  linewidth=2, linestyle="-")
    fig.suptitle('Actual and Predicted demand Vs Index', fontsize=18)
    plt.xlabel('Index', fontsize=16)
    plt.ylabel('Demand', fontsize=16)
```

In [137…

```
plot_actual_predicted(y_test,y_test_pred)
```



Inference - The plot shows that the actual and predicted values are very close to each other

6. INTERPRETATION OF RESULTS

Based on the built linear regression model, **BoomBikes** company should focus on the following features :

- It would be better to focus on how to expand business during **Spring** by advertising special offers and discounts planned for the season.

- Company can improve business by encouraging the customers through **'year end' offers** and **special free rides** based on lot - system so that the business doesnot drop down during 'year-end' period, **(i.e. September to December)**.

- Rental bikes are not preffered when the weather is **misty, snowy or rainy**. In these situations, company can improve the business by providing **Waterproof Bike Shield Covers** to the customers so that they wont be drenched in rain or snow.

- Based on the **demographical data, weather and season**, the spread of the number of bikes over 'docks' has to be determined so that the **available resources are utilized well**.

- Every end of the month there is a **decline** in bike sharing users. This may be due to the insufficient monetary trends at the **month-end**. Hence, give some flexible time for payment based on the membership and profile of the users.

Significant features to predict the DEMAND for shared bikes are :

1. Season (Spring, Winter)
2. Weather (Light Snow or Rain, Misty)
3. Mid Month
4. Weekday (Saturday)
5. Month (January, September, November, December)
6. Year
7. Workingday
8. Temperature

Coefficients of the significant variables (how well these variables describe the bike demands) :

Coefficient of the feature - Spring : -0.11549906427979702
Coefficient of the feature - Winter : 0.07107811036970689
Coefficient of the feature - Light Snow or Rain : -0.32528257723638254
Coefficient of the feature - Misty : -0.07913116256968115
Coefficient of the feature - Mid Month : 0.04039813583014747
Coefficient of the feature - Saturday : 0.06440997256785737
Coefficient of the feature - December : -0.07308927186144788
Coefficient of the feature - January : -0.05750498734447426
Coefficient of the feature - November : -0.09136891118991015
Coefficient of the feature - September : 0.049971247277551374
Coefficient of the feature - yr : 0.2343096537075494
Coefficient of the feature - workingday : 0.058977436299761675
Coefficient of the feature - temp : 0.3714075618040843
Intercept value of the built model : 0.1925153799731144

The linear equation for predicting the demand for shared bike is given by :

DEMAND = (-0.11549906427979702 ***SPRING***) + *(0.07107811036970689* **WINTER**) + (-0.32528257723638254 *LIGHT SNOW OR >* ***RAIN***) + *(-0.07913116256968115* **MISTY**) + (0.04039813583014747 ***MID MONTH***) + *(0.06440997256785737* **SATURDAY**) + (-0.07308927186144788 ***DECEMBER***) + *(-0.05750498734447426* **JANUARY**) + (-0.09136891118991015 ***NOVEMBER***) + *(0.049971247277551374* **SEPTEMBER**) + (0.2343096537075494 ***YEAR***) + *(0.058977436299761675* **WORKINGDAY**) + (0.3714075618040843 * **TEMP**) + 0.1925153799731144

In [ ]: