Business Case: Yulu - Hypothesis Testing

About Yulu

Yulu is India’s leading micro-mobility service provider, which offers unique vehicles for the daily commute. Starting off as a mission to eliminate traffic congestion in India, Yulu provides the safest commute solution through a user-friendly mobile app to enable shared, solo and sustainable commuting.

Yulu zones are located at all the appropriate locations (including metro stations, bus stands, office spaces, residential areas, corporate offices, etc) to make those first and last miles smooth, affordable, and convenient!

Yulu has recently suffered considerable dips in its revenues. They have contracted a consulting company to understand the factors on which the demand for these shared electric cycles depends. Specifically, they want to understand the factors affecting the demand for these shared electric cycles in the Indian market.

Define Problem Statement and perform Exploratory Data Analysis

Import Libararies

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from scipy import stats
```

Loading the Data Set

```
df = pd.read_csv("Yulu.csv")

df
```

```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 10886,\n  \"fields\":
[\n    {\n      \"column\": \"datetime\",\n      \"properties\": {\n
\"dtype\": \"object\",\n        \"num_unique_values\": 10886,\n
\"samples\": [\n          \"2011-07-19 11:00:00\",\n          \"2012-
01-16 06:00:00\",\n          \"2011-12-11 18:00:00\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"season\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\": 1,\n
\"min\": 1,\n        \"max\": 4,\n        \"num_unique_values\": 4,\n
\"samples\": [\n          2,\n          4,\n          1\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"holiday\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\": 0,\n
\"min\": 0,\n        \"max\": 1,\n        \"num_unique_values\": 2,\n
```

```
\"samples\": [\n          1,\n          0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"workingday\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0,\n        \"min\": 0,\n        \"max\": 1,\n
\"num_unique_values\": 2,\n        \"samples\": [\n          1,\n
0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"weather\",\n      \"properties\": {\n        \"dtype\": \"number\",\
n        \"std\": 0,\n        \"min\": 1,\n        \"max\": 4,\n
\"num_unique_values\": 4,\n        \"samples\": [\n          2,\n
4\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"temp\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 7.791589843987567,\n        \"min\": 0.82,\n        \"max\":
41.0,\n        \"num_unique_values\": 49,\n        \"samples\": [\n
6.56,\n          1.64\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"atemp\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 8.474600626484948,\n        \"min\": 0.76,\n        \"max\":
45.455,\n        \"num_unique_values\": 60,\n        \"samples\": [\n
14.395,\n          16.665\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"humidity\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 19,\n        \"min\": 0,\n
\"max\": 100,\n        \"num_unique_values\": 89,\n
\"samples\": [\n          29,\n          61\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"windspeed\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
8.164537326838689,\n        \"min\": 0.0,\n        \"max\": 56.9969,\n
\"num_unique_values\": 28,\n        \"samples\": [\n
22.0028,\n          43.0006\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"casual\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 49,\n        \"min\": 0,\n
\"max\": 367,\n        \"num_unique_values\": 309,\n
\"samples\": [\n          287,\n          47\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"registered\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
151,\n        \"min\": 0,\n        \"max\": 886,\n
\"num_unique_values\": 731,\n        \"samples\": [\n          566,\n
9\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"count\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 181,\n        \"min\": 1,\n        \"max\": 977,\n
\"num_unique_values\": 822,\n        \"samples\": [\n          626,\n
256\n        ],\n        \"semantic_type\": \"\",\n
```

```
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"df"}
```

Lets Check Basic Details

```
df.head()
```

```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 10886,\n  \"fields\":
[\n    {\n      \"column\": \"datetime\",\n      \"properties\": {\n
\"dtype\": \"object\",\n        \"num_unique_values\": 10886,\n
\"samples\": [\n          \"2011-07-19 11:00:00\",\n          \"2012-
01-16 06:00:00\",\n          \"2011-12-11 18:00:00\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"season\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\": 1,\n
\"min\": 1,\n        \"max\": 4,\n        \"num_unique_values\": 4,\n
\"samples\": [\n          2,\n          4,\n          1\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"holiday\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\": 0,\n
\"min\": 0,\n        \"max\": 1,\n        \"num_unique_values\": 2,\n
\"samples\": [\n          1,\n          0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"workingday\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0,\n        \"min\": 0,\n        \"max\": 1,\n
\"num_unique_values\": 2,\n        \"samples\": [\n          1,\n
0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"weather\",\n      \"properties\": {\n        \"dtype\": \"number\",\
n        \"std\": 0,\n        \"min\": 1,\n        \"max\": 4,\n
\"num_unique_values\": 4,\n        \"samples\": [\n          2,\n
4\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"temp\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 7.791589843987567,\n        \"min\": 0.82,\n        \"max\":
41.0,\n        \"num_unique_values\": 49,\n        \"samples\": [\n
6.56,\n          1.64\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"atemp\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 8.474600626484948,\n        \"min\": 0.76,\n        \"max\":
45.455,\n        \"num_unique_values\": 60,\n        \"samples\": [\n
14.395,\n          16.665\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"humidity\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 19,\n        \"min\": 0,\n
\"max\": 100,\n        \"num_unique_values\": 89,\n
\"samples\": [\n          29,\n          61\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
```

n   },\n   {\n   \"column\": \"windspeed\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 8.164537326838689,\n   \"min\": 0.0,\n   \"max\": 56.9969,\n   \"num_unique_values\": 28,\n   \"samples\": [\n   22.0028,\n   43.0006\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"casual\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 49,\n   \"min\": 0,\n   \"max\": 367,\n   \"num_unique_values\": 309,\n   \"samples\": [\n   287,\n   47\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"registered\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 151,\n   \"min\": 0,\n   \"max\": 886,\n   \"num_unique_values\": 731,\n   \"samples\": [\n   566,\n   9\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"count\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 181,\n   \"min\": 1,\n   \"max\": 977,\n   \"num_unique_values\": 822,\n   \"samples\": [\n   626,\n   256\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   }\n]\n}","type":"dataframe","variable_name":"df"}

```
df.tail()
```

{"repr_error":"0","type":"dataframe"}

```
df.shape
```

```
(10886, 12)
```

```
df.ndim
```

```
2
```

```
df.describe()
```

{"summary":"{\n \"name\": \"df\",\n \"rows\": 8,\n \"fields\": [\n   {\n   \"column\": \"season\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 3847.8922948663444,\n   \"min\": 1.0,\n   \"max\": 10886.0,\n   \"num_unique_values\": 7,\n   \"samples\": [\n   10886.0,\n   2.5066139996325556,\n   3.0\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"holiday\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 3848.7218602181274,\n   \"min\": 0.0,\n   \"max\": 10886.0,\n   \"num_unique_values\": 5,\n   \"samples\": [\n   0.02856880396839978,\n   1.0,\n   0.16659885062470958\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n

```
}\n    },\n    {\n      \"column\": \"workingday\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
3848.5727758849685,\n        \"min\": 0.0,\n        \"max\": 10886.0,\
n        \"num_unique_values\": 5,\n        \"samples\": [\n
0.6808745177291935,\n          1.0,\n          0.4661591687997356\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"weather\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
3848.224134081727,\n        \"min\": 0.6338385858190958,\n
\"max\": 10886.0,\n        \"num_unique_values\": 6,\n
\"samples\": [\n          10886.0,\n          1.418427337865148,\n
4.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"temp\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 3842.208812643129,\n        \"min\": 0.82,\n        \"max\":
10886.0,\n        \"num_unique_values\": 8,\n        \"samples\": [\n
20.23085981995223,\n          20.5,\n          10886.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"atemp\",\n      \"properties\": {\
n        \"dtype\": \"number\",\n        \"std\": 3841.214609020895,\n
\"min\": 0.76,\n        \"max\": 10886.0,\n
\"num_unique_values\": 8,\n        \"samples\": [\n
23.655084052912,\n          24.24,\n          10886.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"humidity\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\":
3830.3684503021896,\n        \"min\": 0.0,\n        \"max\": 10886.0,\
n        \"num_unique_values\": 8,\n        \"samples\": [\n
61.88645967297446,\n          62.0,\n          10886.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"windspeed\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
3843.014939445678,\n        \"min\": 0.0,\n        \"max\": 10886.0,\n
\"num_unique_values\": 8,\n        \"samples\": [\n
12.7993954069447,\n          12.998,\n          10886.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"casual\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\":
3824.2753676913135,\n        \"min\": 0.0,\n        \"max\": 10886.0,\
n        \"num_unique_values\": 8,\n        \"samples\": [\n
36.02195480433584,\n          17.0,\n          10886.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"registered\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
3779.869612125704,\n        \"min\": 0.0,\n        \"max\": 10886.0,\n
\"num_unique_values\": 8,\n        \"samples\": [\n
155.5521771082124,\n          118.0,\n          10886.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"count\",\n      \"properties\": {\
```

n   \"dtype\": \"number\",\n   \"std\": 3769.174237043881,\n
\"min\": 1.0,\n   \"max\": 10886.0,\n
\"num_unique_values\": 8,\n   \"samples\": [\n
191.57413191254824,\n   145.0,\n   10886.0\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   }\n]\n}","type":"dataframe"}

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10886 entries, 0 to 10885
Data columns (total 12 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   datetime    10886 non-null  object
 1   season      10886 non-null  int64
 2   holiday     10886 non-null  int64
 3   workingday  10886 non-null  int64
 4   weather     10886 non-null  int64
 5   temp        10886 non-null  float64
 6   atemp       10886 non-null  float64
 7   humidity    10886 non-null  int64
 8   windspeed   10886 non-null  float64
 9   casual      10886 non-null  int64
 10  registered  10886 non-null  int64
 11  count       10886 non-null  int64
dtypes: float64(3), int64(8), object(1)
memory usage: 1020.7+ KB
```

Datatype of following attributes needs to change to proper data type :-

datetime - to datetime

season - to categorical

holiday - to categorical

workingday - to categorical

weather - to categorical

```
df["datetime"] = pd.to_datetime(df["datetime"])

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10886 entries, 0 to 10885
Data columns (total 12 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   datetime    10886 non-null  datetime64[ns]
 1   season      10886 non-null  int64
 2   holiday     10886 non-null  int64
 3   workingday  10886 non-null  int64
 4   weather     10886 non-null  int64
 5   temp        10886 non-null  float64
 6   atemp       10886 non-null  float64
 7   humidity    10886 non-null  int64
 8   windspeed   10886 non-null  float64
 9   casual      10886 non-null  int64
 10  registered  10886 non-null  int64
 11  count       10886 non-null  int64
dtypes: datetime64[ns](1), float64(3), int64(8)
memory usage: 1020.7 KB

cat_cols =["season" ,"holiday" , "workingday" , "weather"]
for col in cat_cols:
  df[col] = df[col].astype("object")

  # cat col is defined as where we store

df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10886 entries, 0 to 10885
Data columns (total 12 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   datetime    10886 non-null  datetime64[ns]
 1   season      10886 non-null  object
 2   holiday     10886 non-null  object
 3   workingday  10886 non-null  object
 4   weather     10886 non-null  object
 5   temp        10886 non-null  float64
 6   atemp       10886 non-null  float64
 7   humidity    10886 non-null  int64
 8   windspeed   10886 non-null  float64
 9   casual      10886 non-null  int64
 10  registered  10886 non-null  int64
 11  count       10886 non-null  int64
dtypes: datetime64[ns](1), float64(3), int64(4), object(4)
memory usage: 1020.7+ KB
```

Missing Value Detection:

```
df.isnull()
```

```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 10886,\n  \"fields\":
[\n    {\n      \"column\": \"datetime\",\n      \"properties\": {\n
\"dtype\": \"boolean\",\n        \"num_unique_values\": 1,\n
\"samples\": [\n          false\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"season\",\n      \"properties\":
{\n        \"dtype\": \"boolean\",\n        \"num_unique_values\": 1,\
n        \"samples\": [\n          false\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"holiday\",\n      \"properties\":
{\n        \"dtype\": \"boolean\",\n        \"num_unique_values\": 1,\
n        \"samples\": [\n          false\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"workingday\",\n
\"properties\": {\n        \"dtype\": \"boolean\",\n
\"num_unique_values\": 1,\n        \"samples\": [\n          false\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"weather\",\n
\"properties\": {\n        \"dtype\": \"boolean\",\n
\"num_unique_values\": 1,\n        \"samples\": [\n          false\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"temp\",\n      \"properties\":
{\n        \"dtype\": \"boolean\",\n        \"num_unique_values\": 1,\
n        \"samples\": [\n          false\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"atemp\",\n      \"properties\": {\
n        \"dtype\": \"boolean\",\n        \"num_unique_values\": 1,\n
\"samples\": [\n          false\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"humidity\",\n      \"properties\":
{\n        \"dtype\": \"boolean\",\n        \"num_unique_values\": 1,\
n        \"samples\": [\n          false\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"windspeed\",\n
\"properties\": {\n        \"dtype\": \"boolean\",\n
\"num_unique_values\": 1,\n        \"samples\": [\n          false\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"casual\",\n      \"properties\":
{\n        \"dtype\": \"boolean\",\n        \"num_unique_values\": 1,\
n        \"samples\": [\n          false\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"registered\",\n
\"properties\": {\n        \"dtype\": \"boolean\",\n
\"num_unique_values\": 1,\n        \"samples\": [\n          false\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"count\",\n      \"properties\":
{\n        \"dtype\": \"boolean\",\n        \"num_unique_values\": 1,\
n        \"samples\": [\n          false\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe"}
```

There are no missing values present in the dataset.

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

```
datetime      0
season        0
holiday       0
workingday    0
weather       0
temp          0
atemp         0
humidity      0
windspeed     0
casual        0
registered    0
count         0
dtype: int64
```

```
df["datetime"].min()
```

```
Timestamp('2011-01-01 00:00:00')
```

```
df["datetime"].max()
```

```
Timestamp('2012-12-19 23:00:00')
```

```
# number of unique values in each categorical columns
df[cat_cols].melt().groupby(['variable','value'])[["value"]].count()
```

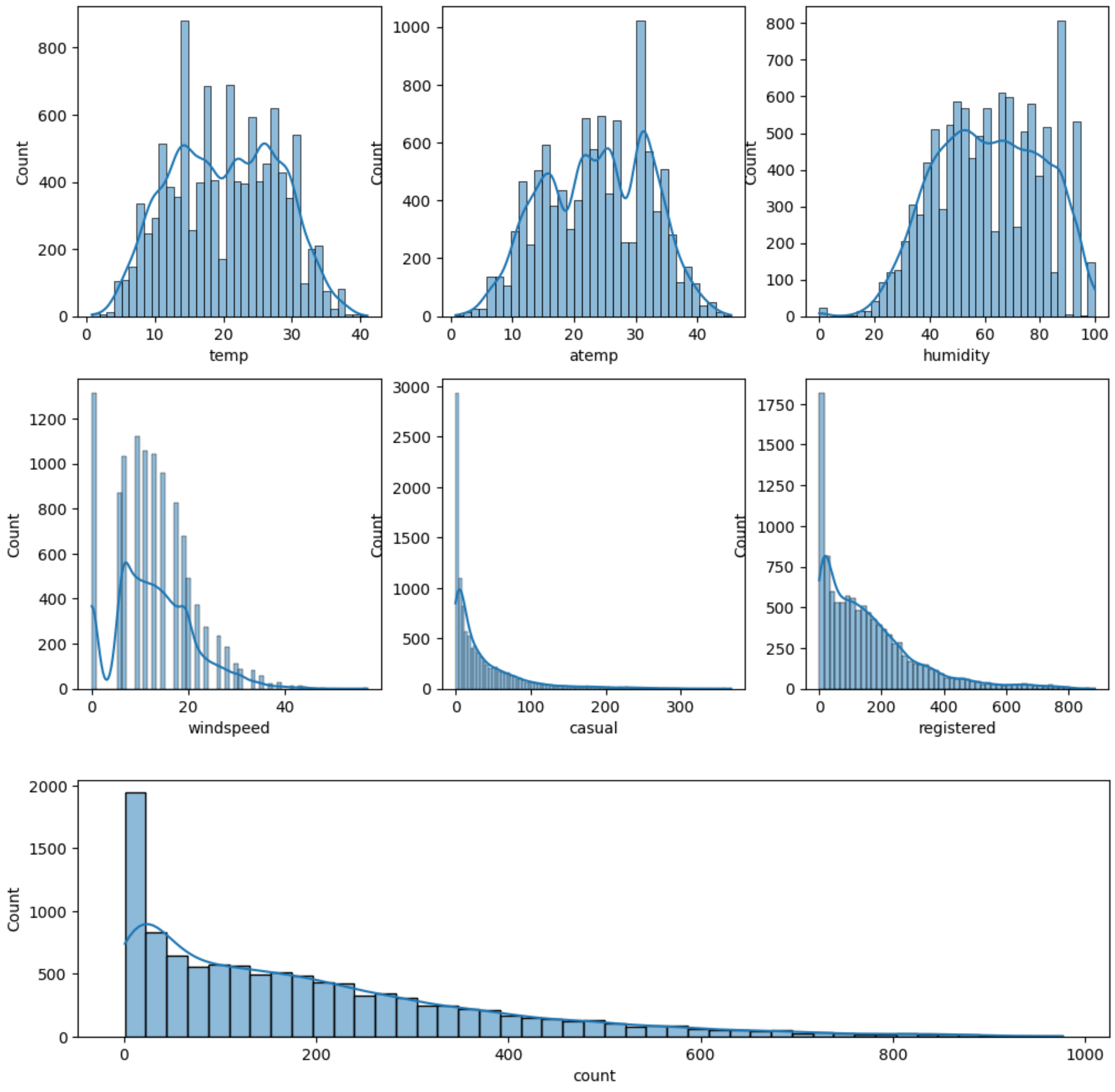
```
{"summary":"{\n  \"name\": \"df[cat_cols]\",\n  \"rows\": 12,\n  \"fields\": [\n    {\n      \"column\": \"value\",\n      \"properties\": {\n        \"dtype\": \"number\",\n        \"std\": 3177,\n        \"min\": 1,\n        \"max\": 10575,\n        \"num_unique_values\": 11,\n        \"samples\": [\n          7192,\n          10575,\n          3474\n        ],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n      }\n    }\n  ]\n}","type":"dataframe"}
```

Univariate Analysis (distribution plots of all the continuous variable(s) barplots/countplots of all the categorical variables)

Histplot

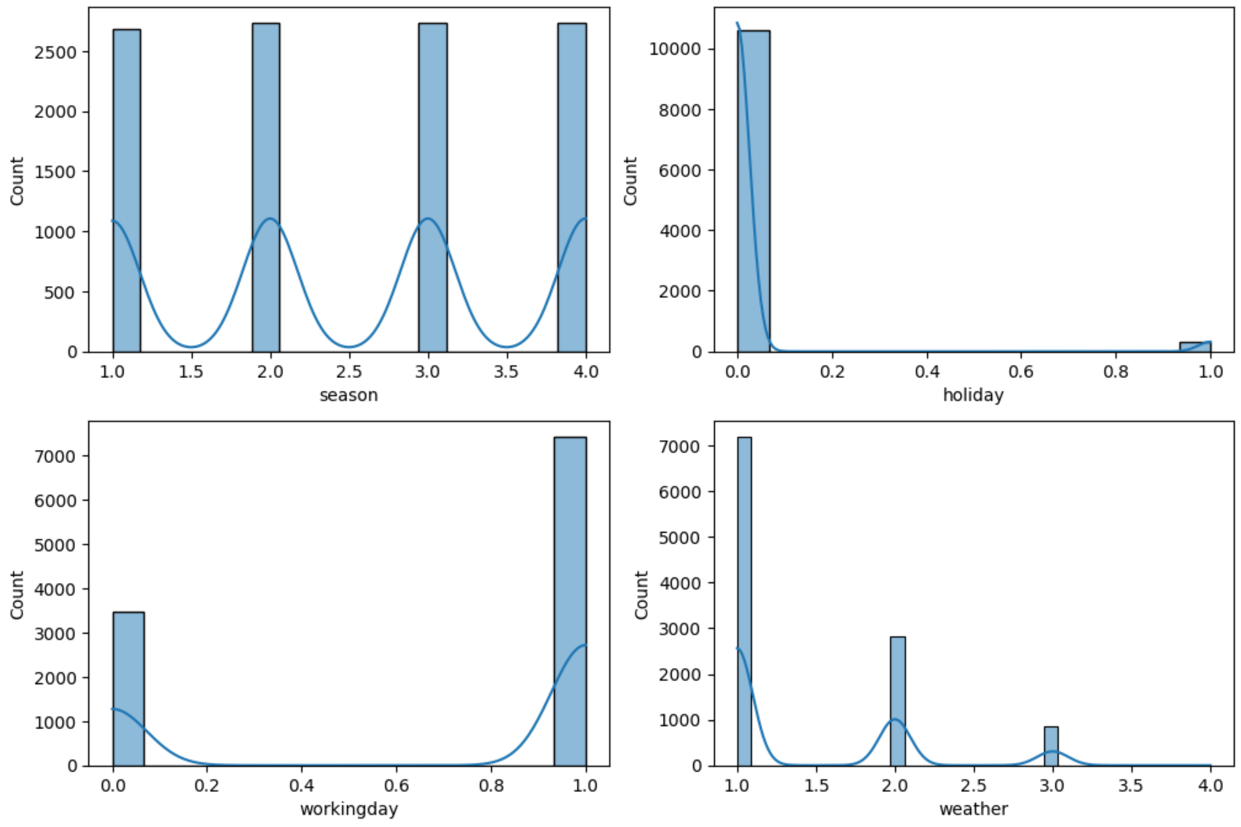
```
num_cols = ['temp', 'atemp', 'humidity', 'windspeed', 'casual',
'registered', 'count']
fig,axis = plt.subplots(nrows= 2, ncols = 3, figsize = (12,8))
index = 0
for row in range(2):
  for col in range(3):
    sns.histplot(df[num_cols[index]], ax = axis[row, col], kde = True)
    index += 1
plt.show()
plt.figure(figsize=(12, 3))
```

```
sns.histplot(df[num_cols[-1]], kde = True)
plt.show()
```



1. casual, registered and count somewhat looks like Log Normal Distribution
2. temp, atemp and humidity looks like they follows the Normal Distribution
3. windspeed follows the binomial distribution

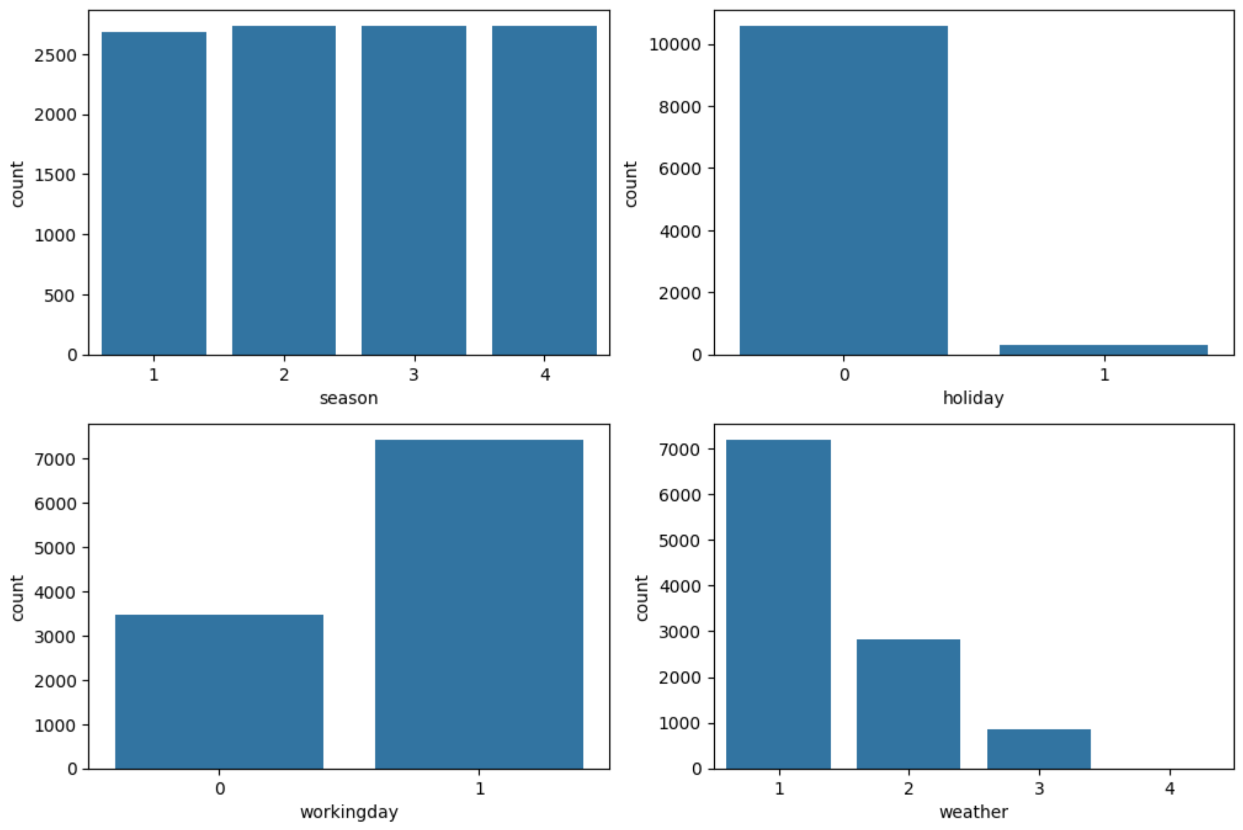
```
cat_cols =["season" ,"holiday" , "workingday" , "weather"]
fig,axis = plt.subplots(nrows= 2, ncols = 2, figsize = (12,8))
index = 0
for row in range(2):
  for col in range(2):
    sns.histplot(df[cat_cols[index]], ax = axis[row, col], kde = True)
    index += 1
plt.show()
```



1. casual, registered and count somewhat looks like Log Normal Distribution
2. temp, atemp and humidity looks like they follows the Normal Distribution
3. windspeed follows the binomial distribution

Countplot

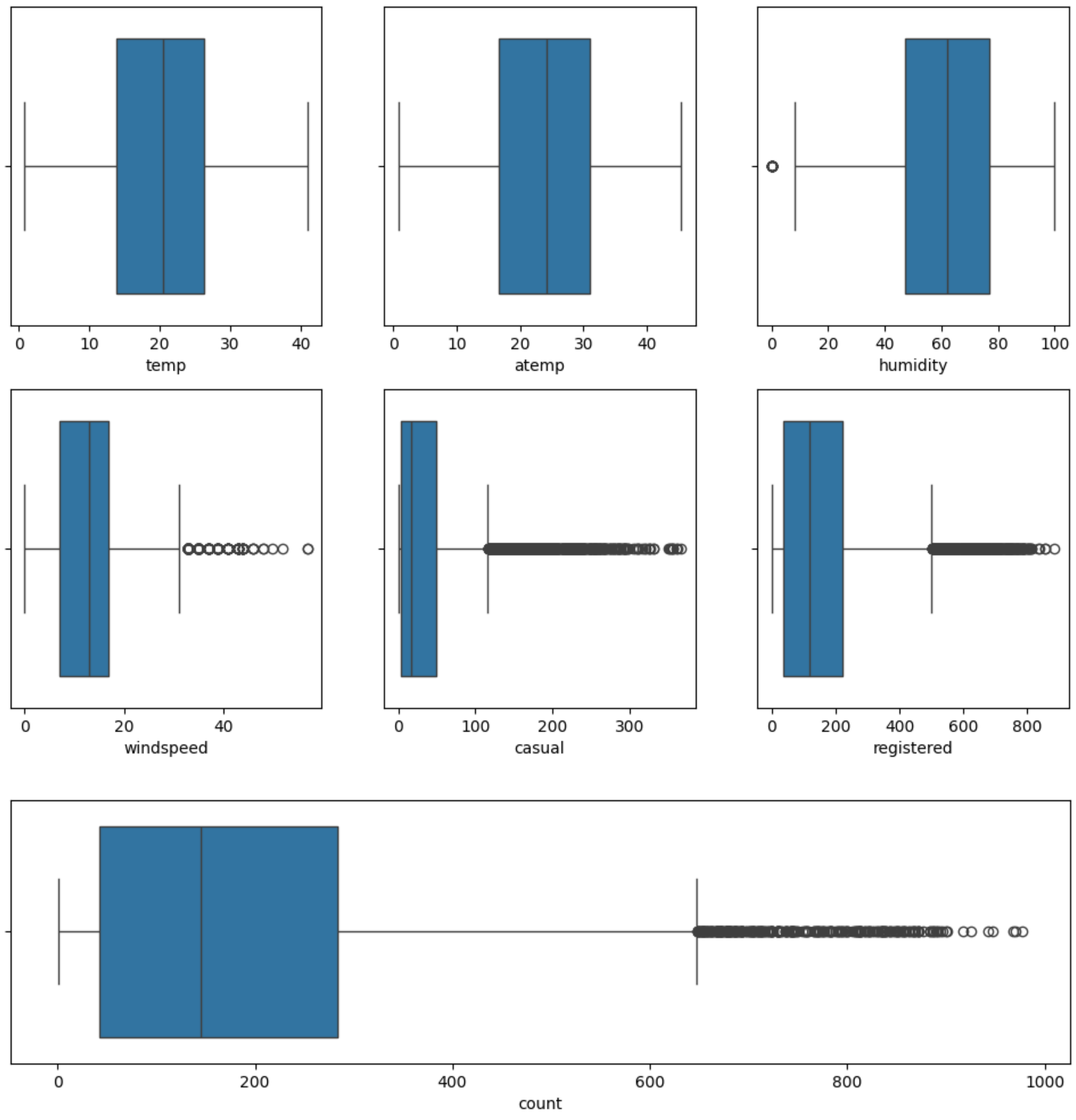
```
cat_cols =["season" ,"holiday" , "workingday" , "weather"]
fig,axis = plt.subplots(nrows= 2, ncols = 2, figsize = (12,8))
index = 0
for row in range(2):
  for col in range(2):
    sns.countplot(data = df, x = cat_cols[index], ax = axis[row, col])
    index += 1

plt.show()
```



Data looks common as it should be like equal number of days in each season, more working days and weather is mostly Clear, Few clouds and partly cloudy.

Boxplot

```
num_cols = ['temp', 'atemp', 'humidity', 'windspeed', 'casual',
'registered', 'count']
fig,axis = plt.subplots(nrows= 2, ncols = 3, figsize = (12,8))
index = 0
for row in range(2):
  for col in range(3):
    sns.boxplot(x = df[num_cols[index]], ax = axis[row, col])
    index += 1
plt.show()
plt.figure(figsize=(12, 3))
sns.boxplot(data = df ,x = df[num_cols[-1]])
plt.show()
```

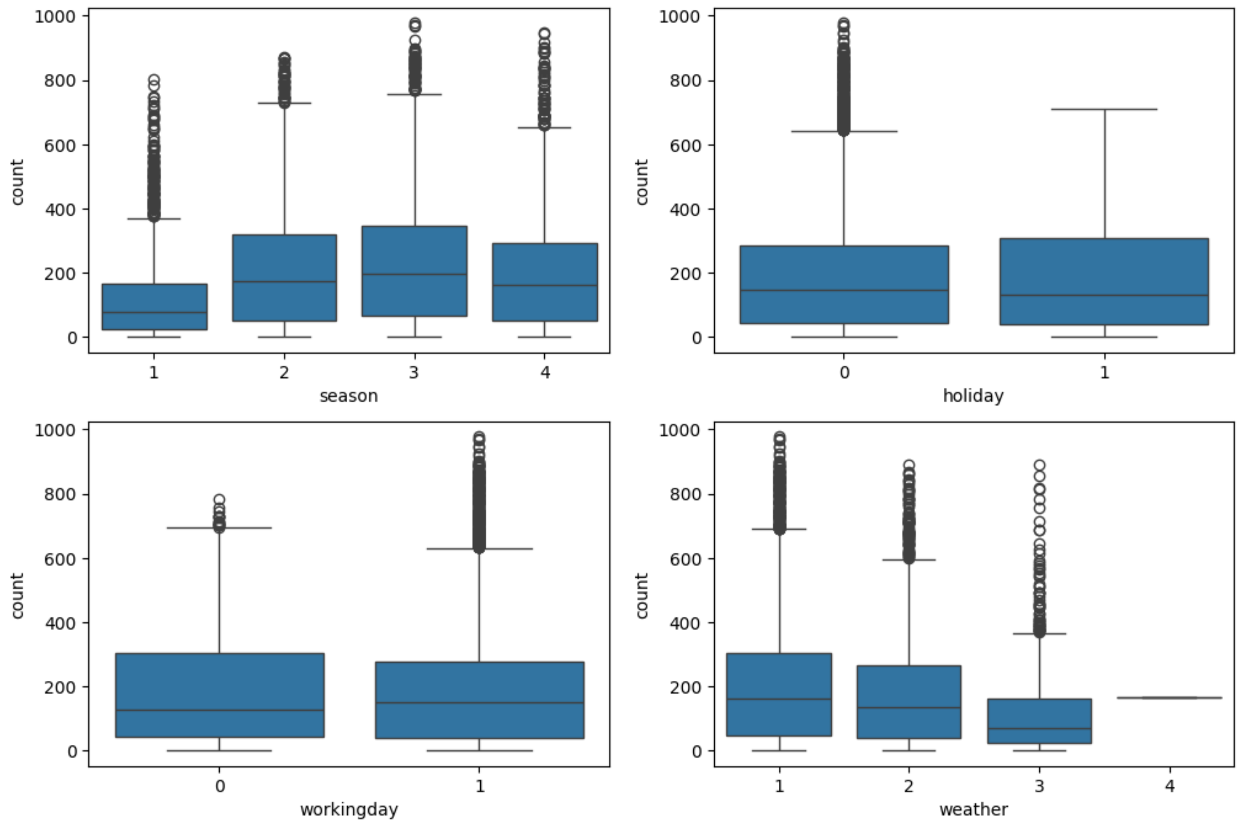


Looks like humidity , casual , registered and count have outliers in the data

Bivariate Analysis (Relationships between important variables such as workday and count, season and count, weather and count.

Boxplot

```
cat_cols = ["season","holiday","workingday","weather"]
fig,axis = plt.subplots(nrows = 2,ncols = 2 ,figsize =  (12,8))
index = 0
for row in range (2):
  for col in range (2):
    sns.boxplot(data = df,x = cat_cols[index] ,y = "count", ax =
axis[row , col])
    index +=1
plt.show()
```



In summer and fall seasons more bikes are rented as compared to other seasons.

Whenever its a holiday more bikes are rented.

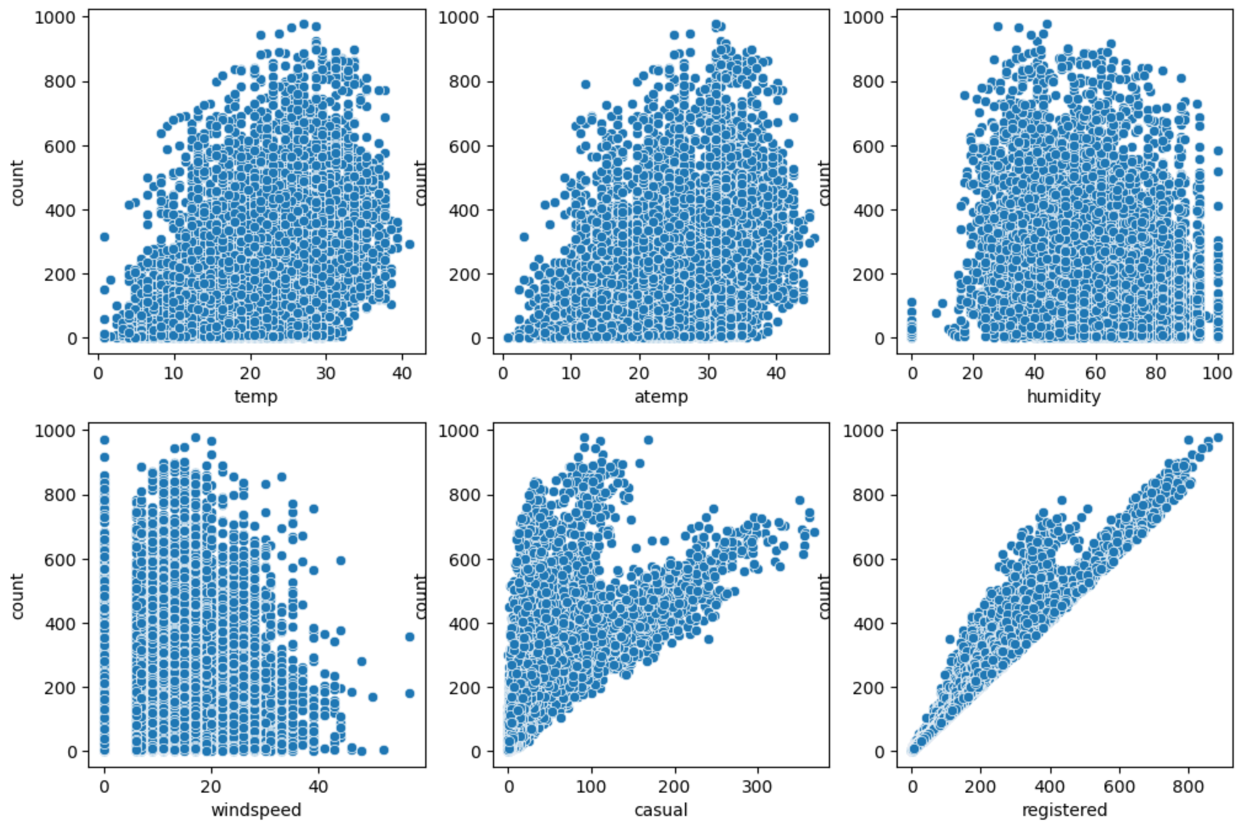
It is also clear from the workingday also that whenever day is holiday or weekend, slightly more bikes were rented.

Whenever there is rain, thunderstorm, snow or fog, there were less bikes were rented.

Scatterplot

```
num_cols = ['temp', 'atemp', 'humidity', 'windspeed', 'casual',
'registered', 'count']
fig,axis = plt.subplots(nrows= 2, ncols = 3, figsize = (12,8))
index = 0
for row in range(2):
  for col in range(3):
    sns.scatterplot(data = df , x = num_cols[index] , y = "count" , ax
= axis[row,col])
```

```
    index += 1
plt.show()
```

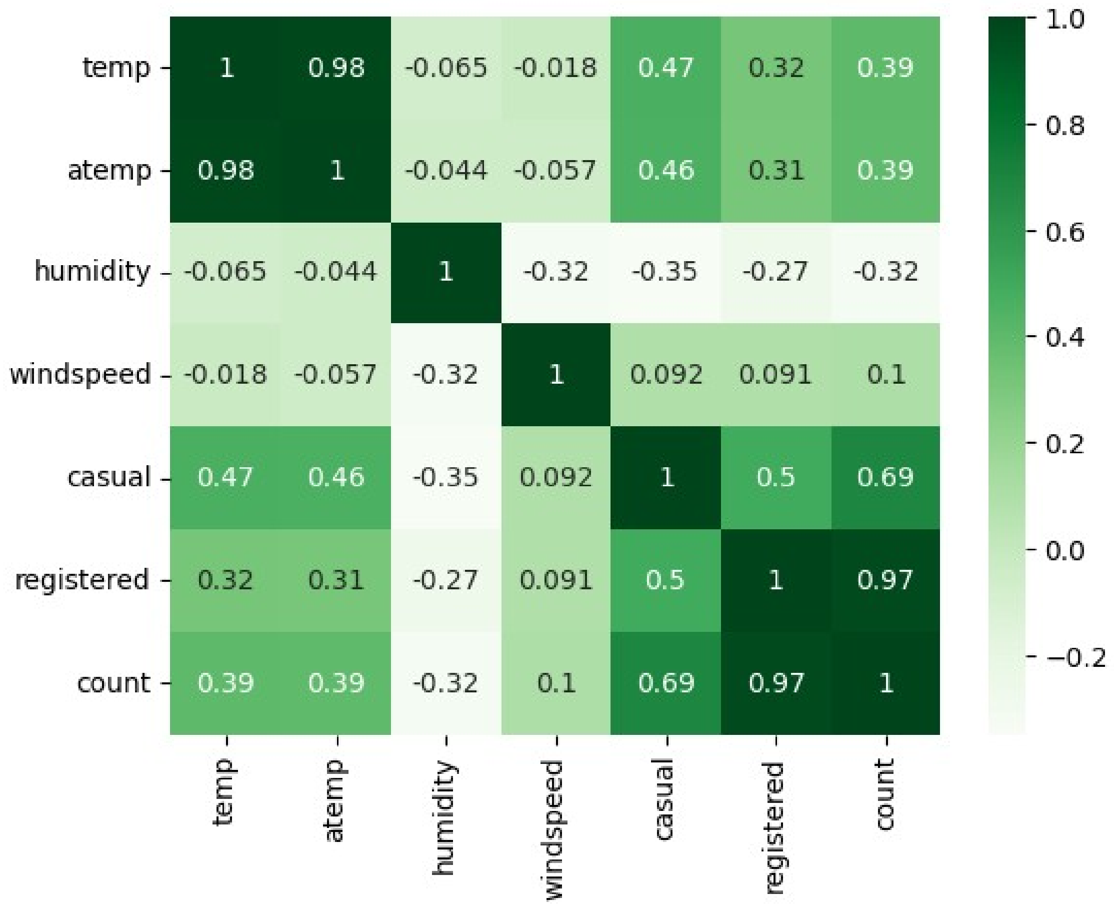


Whenever the humidity is less than 20, number of bikes rented is very very low. Whenever the temperature is less than 10, number of bikes rented is less. Whenever the windspeed is greater than 35, number of bikes rented is less.

Heat Map

```
# understanding the correlation between count and numerical variables
df.corr()['count']
sns.heatmap(df.corr(), annot= True , cmap= 'Greens')
plt.show()
```

```
<ipython-input-65-032db0adf37c>:2: FutureWarning: The default value of
numeric_only in DataFrame.corr is deprecated. In a future version, it
will default to False. Select only valid columns or specify the value
of numeric_only to silence this warning.
  df.corr()['count']
<ipython-input-65-032db0adf37c>:3: FutureWarning: The default value of
numeric_only in DataFrame.corr is deprecated. In a future version, it
will default to False. Select only valid columns or specify the value
of numeric_only to silence this warning.
  sns.heatmap(df.corr(), annot= True , cmap= 'Greens')
```



Variables are positively correlated here, where one increases and the other increases too. In contrast, some are negatively correlated also where the high values of one variable go with the low values of another variable. corr= +1 indicates perfect positive correlation.

Hypothesis Testing

2- Sample T-Test to check if Working Day has an effect on the number of electric cycles rented

Null Hypothesis: Working day has no effect on the number of cycles being rented.

Alternate Hypothesis: Working day has effect on the number of cycles being rented.

Significance level (alpha): 0.05 We will use the 2-Sample T-Test to test the hypothess defined above

Before conducting the two-sample T-Test we need to find if the given data groups have the same variance. If the ratio of the larger data groups to the small data group is less than 4:1 then we can consider that the given data groups have equal variance.

```
data_group1 = df[df['workingday']==0]['count'].values
data_group2 = df[df['workingday']==1]['count'].values
```

```
print(np.var(data_group1), np.var(data_group2))
np.var(data_group2)// np.var(data_group1)

30171.346098942427 34040.69710674686

1.0
```

Here, the ratio is 34040.70 / 30171.35 which is less than 4:1

```
statistic, p_value = stats.ttest_ind(a=data_group1, b=data_group2,
equal_var=True)
print("statistic: ", statistic)
print("p-value: ", p_value)

statistic:  -1.2096277376026694
p-value:  0.22644804226361348

if p_value < 0.05:
    print("Reject H0")
    print("Atleast one group have different mean")
else:
    print("Fail to reject H0")
    print("All groups have same mean")

Fail to reject H0
All groups have same mean
```

Since pvalue is greater than 0.05 so we cannot reject the Null hypothesis. We don't have the sufficient evidence to say that working day has effect on the number of cycles being rented.

Chi-square test to check if Weather is dependent on the season

Chi-square test to check if Weather is dependent on the season
Null Hypothesis (H0): Weather is independent of the season Alternate Hypothesis (H1): Weather is not independent of the season Significance level (alpha): 0.05

```
data_table = pd.crosstab(df['season'], df['weather'])
print("Observed values:")
data_table

Observed values:
```

```
{"summary":"{\n  \"name\": \"data_table\",\n  \"rows\": 4,\n
\"fields\": [\n    {\n      \"column\": \"season\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
1,\n        \"min\": 1,\n        \"max\": 4,\n
\"num_unique_values\": 4,\n        \"samples\": [\n          2,\n
4,\n          1\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\": 1,\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
96,\n        \"min\": 1702,\n        \"max\": 1930,\n
```

\"num_unique_values\": 4,\n   \"samples\": [\n   1801,\n   1702,\n   1759\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": 2,\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 82,\n   \"min\": 604,\n   \"max\": 807,\n   \"num_unique_values\": 4,\n   \"samples\": [\n   708,\n   807,\n   715\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": 3,\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 12,\n   \"min\": 199,\n   \"max\": 225,\n   \"num_unique_values\": 4,\n   \"samples\": [\n   224,\n   225,\n   211\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": 4,\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 0,\n   \"min\": 0,\n   \"max\": 1,\n   \"num_unique_values\": 2,\n   \"samples\": [\n   0,\n   1\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   }\n]\n}","type":"dataframe","variable_name":"data_table"}

```
val = stats.chi2_contingency(data_table)
print(val)
```

```
Chi2ContingencyResult(statistic=49.158655596893624,
pvalue=1.549925073686492e-07, dof=9,
expected_freq=array([[1.77454639e+03, 6.99258130e+02, 2.11948742e+02,
2.46738931e-01],
       [1.80559765e+03, 7.11493845e+02, 2.15657450e+02, 2.51056403e-01],
       [1.80559765e+03, 7.11493845e+02, 2.15657450e+02, 2.51056403e-01],
       [1.80625831e+03, 7.11754180e+02, 2.15736359e+02, 2.51148264e-01]]))
```

```
expected_values = val[3]
print(expected_values)
nrows, ncols = 4, 4
dof = (nrows-1)*(ncols-1)
print("degrees of freedom: ", dof)
alpha = 0.05
```

```
[[1.77454639e+03 6.99258130e+02 2.11948742e+02 2.46738931e-01]
 [1.80559765e+03 7.11493845e+02 2.15657450e+02 2.51056403e-01]
 [1.80559765e+03 7.11493845e+02 2.15657450e+02 2.51056403e-01]
 [1.80625831e+03 7.11754180e+02 2.15736359e+02 2.51148264e-01]]
degrees of freedom:  9
```

```
chi_sqr = sum([(o-e)**2/e for o, e in zip(data_table.values,
expected_values)])
chi_sqr_statistic = chi_sqr[0] + chi_sqr[1]
```

```
print("chi-square test statistic: ", chi_sqr_statistic)

critical_val = stats.chi2.ppf(q=1-alpha, df=dof)
print(f"critical value: {critical_val}")

p_val = 1-stats.chi2.cdf(x=chi_sqr_statistic, df=dof)
print(f"p-value: {p_val}")
```

```
chi-square test statistic:  44.09441248632364
critical value: 16.918977604620448
p-value: 1.3560001579371317e-06
```

```
if p_val <= alpha:
    print("\nSince p-value is less than the alpha 0.05, We reject the
Null Hypothesis. Meaning that\
    Weather is dependent on the season.")
else:
    print("Since p-value is greater than the alpha 0.05, We do not
reject the Null Hypothesis")
```

```

Since p-value is less than the alpha 0.05, We reject the Null
Hypothesis. Meaning that    Weather is dependent on the season.
```

ANNOVA to check if No. of cycles rented is similar or different in different weather and season

ANOVA assumptions

Like other types of statistical methods, ANOVA compares the means of different groups and shows you if there are any statistical differences between the means. ANOVA is classified as an omnibus test statistic. This means that it can’t tell you which specific groups were statistically significantly different from each other, only that at least two of the groups were. ANOVA relies on three main assumptions that must be met for the test results to be valid.

Normality

The first assumption is that the groups each fall into what is called a normal distribution. This means that the groups should have a bell-curve distribution with few or no outliers.

Homogeneity of variance

Also known as homoscedasticity, this means that the variances between each group are the same.

Independence

The final assumption is that each value is independent from each other. This means, for example, that unlike a conjoint analysis the same person shouldn’t be measured multiple times.

Null Hypothesis:

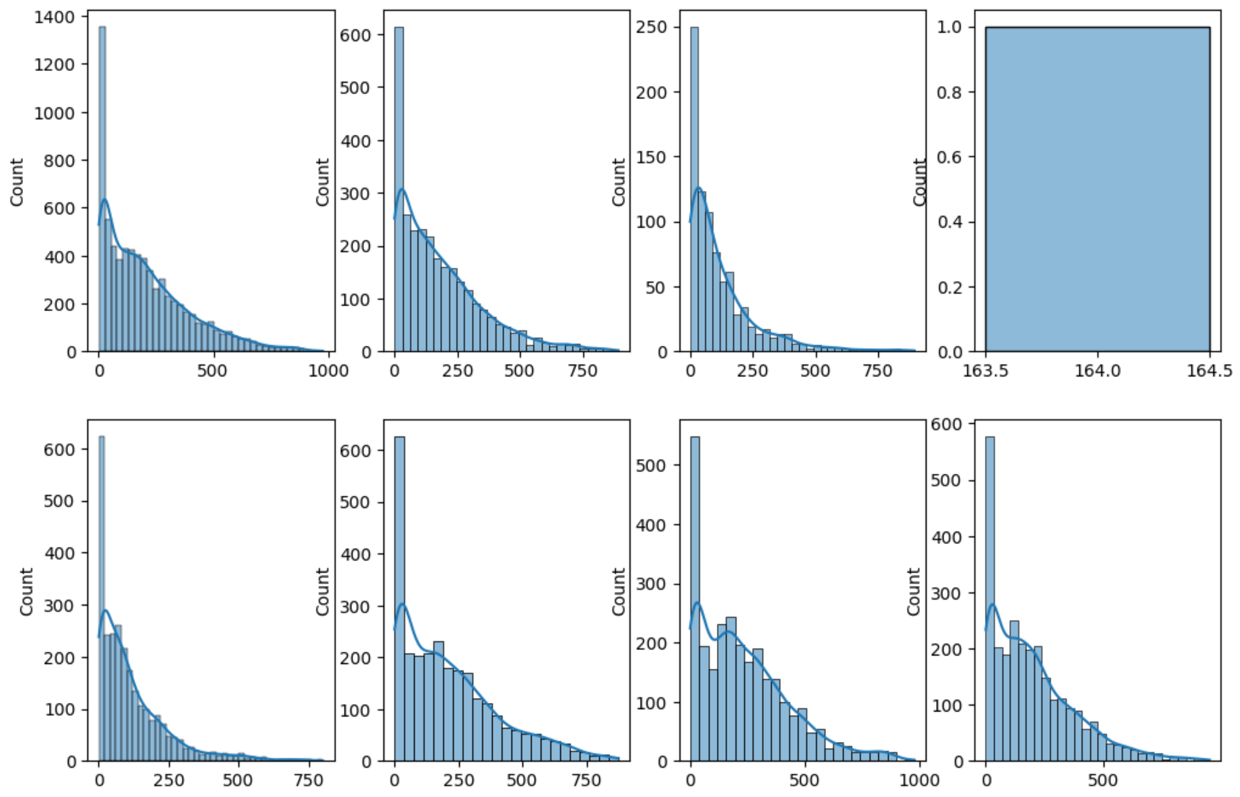
Number of cycles rented is similar in different weather and season.

Alternate Hypothesis:

Number of cycles rented is not similar in different weather and season.

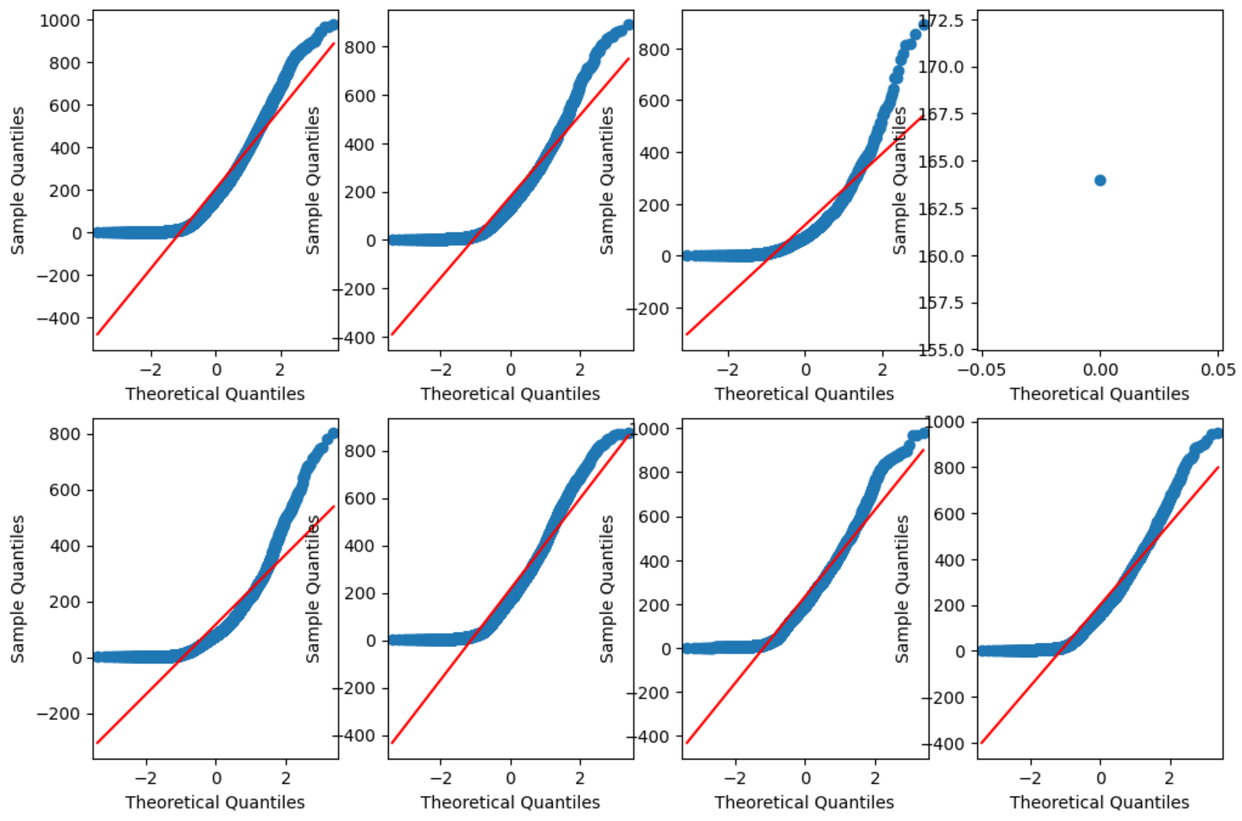
Significance level (alpha): 0.05

```
# defining the data groups for the ANOVA
from statsmodels.graphics.gofplots import qqplot
gp1 = df[df['weather']==1]['count'].values
gp2 = df[df['weather']==2]['count'].values
gp3 = df[df['weather']==3]['count'].values
gp4 = df[df['weather']==4]['count'].values

gp5 = df[df['season']==1]['count'].values
gp6 = df[df['season']==2]['count'].values
gp7 = df[df['season']==3]['count'].values
gp8 = df[df['season']==4]['count'].values
groups=[gp1,gp2,gp3,gp4,gp5,gp6,gp7,gp8]

groups=[gp1,gp2,gp3,gp4,gp5,gp6,gp7,gp8]
fig ,axis =plt.subplots(nrows = 2 , ncols =4 , figsize = (12,8))
index = 0
for row in range(2):
  for col in range(4):
    sns.histplot(groups[index] , ax= axis[row ,col] , kde = True)
    index += 1
plt.show()
```



As one group above in the graph does not reflect as gaussian, So above data also fails the normality test.

```
fig ,axis = plt.subplots( nrows = 2 , ncols = 4 , figsize = (12,8))
index = 0
for row in range(2):
  for col in range(4):
    qqplot(groups[index] ,line = "s" ,ax = axis[row,col])
    index += 1
plt.show()
```



A good way to assess the normality of a data would be to use a Q-Q plot, which gives us a graphical visualization of normality. But, here above Q-Q plot also fails to follow the normality test.

Equal variance: Levene's Test

Null Hypothesis: Variances is similar in different weather and season.

Alternate Hypothesis: Variances is not similar in different weather and season.

Significance level (alpha): 0.05

```
levene_stat, p_value = stats.levene(gp1,gp2,gp3,gp4,gp5,gp6,gp7,gp8)
print(p_value)

3.463531888897594e-148

if p_value < 0.05:
 print("Reject the Null hypothesis.Variances are not equal")
else:
 print("Fail to Reject the Null hypothesis.Variances are equal")

Reject the Null hypothesis.Variances are not equal

#assumptions of ANOVA don't hold, we need Kruskal Wallis

kruskal_stat, p_value = stats.kruskal(gp1,gp2,gp3,gp4,gp5,gp6,gp7,gp8)
```

```
print("p_value===",p_value)
if p_value<0.05:
  print("Since p-value is less than 0.05, we reject the null
hypothesis")
```

```
p_value=== 4.614440933900297e-191
Since p-value is less than 0.05, we reject the null hypothesis
```

Insights

1. In summer and fall seasons more bikes are rented as compared to other seasons.
2. Whenever its a holiday more bikes are rented.
3. It is also clear from the workingday also that whenever day is holiday or weekend, slightly more bikes were rented.
4. Whenever there is rain, thunderstorm, snow or fog, there were less bikes were rented.
5. Whenever the humidity is less than 20, number of bikes rented is very very low.
6. Whenever the temperature is less than 10, number of bikes rented is less.
7. Whenever the windspeed is greater than 35, number of bikes rented is less.
8. A 2-sample T-test on working and non-working days with respect to count,implies that the mean population count of both categories are the same.
9. By performing a Chi2 test on season and weather (categorical variables), we can infer that there is an impact on weather dependent on season.

Recommendations

1. In summer and fall seasons the company should have more bikes in stock to be rented. Because the demand in these seasons is higher as compared to other seasons.
2. With a significance level of 0.05, workingday has no effect on the number of bikes being rented.
3. In very low humid days, company should have less bikes in the stock to be rented.
4. Whenever temperature is less than 10 or in very cold days, company should have less bikes.
5. Whenever the windspeed is greater than 35 or in thunderstorms, company should have less bikes in stock to be rented.