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!

Business Case:

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.

Import the dataset and do usual exploratory data analysis steps like checking the structure & characteristics of the dataset

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

```
data = pd.read_csv('Yulu_BusinessCase_Data.csv')
data
```

	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	count
0	2011-01-01 00:00:00	1	0	0	1	9.84	14.395	81	0.0000	3	13	16
1	2011-01-01 01:00:00	1	0	0	1	9.02	13.635	80	0.0000	8	32	40
2	2011-01-01 02:00:00	1	0	0	1	9.02	13.635	80	0.0000	5	27	32
3	2011-01-01 03:00:00	1	0	0	1	9.84	14.395	75	0.0000	3	10	13
4	2011-01-01 04:00:00	1	0	0	1	9.84	14.395	75	0.0000	0	1	1
...	...	...	...	...	...	...	...	...	...	...	...	...
10881	2012-12-19 19:00:00	4	0	1	1	15.58	19.695	50	26.0027	7	329	336
10882	2012-12-19 20:00:00	4	0	1	1	14.76	17.425	57	15.0013	10	231	241
10883	2012-12-19 21:00:00	4	0	1	1	13.94	15.910	61	15.0013	4	164	168
10884	2012-12-19 22:00:00	4	0	1	1	13.94	17.425	61	6.0032	12	117	129
10885	2012-12-19 23:00:00	4	0	1	1	13.12	16.665	66	8.9981	4	84	88

10886 rows × 12 columns

```
#size of dataset
data.size
```

```
130632
```

```
#shape of the dataset
data.shape
```

```
(10886, 12)
```

```
#dimension of dataset
data.ndim
```

```
2
```

```
data.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
```

conversion of categorical attributes to 'category'

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

```
#changing the datetime column from object to 'datetime' datatype
data['datetime'] = pd.to_datetime(data['datetime'])

#There are 4 different categorical data columns
categorical_col = ['season','holiday','workingday','weather']

for col in categorical_col:
  data[col] = data[col].astype(object)


data.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
```

Statistical Summary

```
data.describe(include = 'all').T
```

```
<ipython-input-41-dcb9e48885d3>:1: FutureWarning: Treating datetime data as categorical rather than numeric in `.describe` is deprecated
  data.describe(include = 'all').T
```

	count	unique	top	freq	first	last	mean	std	min	25%	50%	75%
datetime	10886	10886	2011-01-01 00:00:00	1	2011-01-01	2012-12-19 23:00:00	NaN	NaN	NaN	NaN	NaN	NaN
season	10886.0	4.0	4.0	2734.0	NaT	NaT	NaN	NaN	NaN	NaN	NaN	NaN
holiday	10886.0	2.0	0.0	10575.0	NaT	NaT	NaN	NaN	NaN	NaN	NaN	NaN
workingday	10886.0	2.0	1.0	7412.0	NaT	NaT	NaN	NaN	NaN	NaN	NaN	NaN
weather	10886.0	4.0	1.0	7192.0	NaT	NaT	NaN	NaN	NaN	NaN	NaN	NaN
temp	10886.0	NaN	NaN	NaN	NaT	NaT	20.23086	7.79159	0.82	13.94	20.5	26.24
atemp	10886.0	NaN	NaN	NaN	NaT	NaT	23.655084	8.474601	0.76	16.665	24.24	31.06
registered	10886.0	NaN	NaN	NaN	NaT	NaT	155.552177	151.039033	0.0	36.0	118.0	222.0

- There are no missing values in the dataset.
- Casual and registered attributes might have outliers because their mean and median are very far away to one another and the value of standard deviation is also high which tells us that there is high variance in the data of these attributes.

Checking for NULL values

```
# detecting missing values in the dataset
data.isnull().sum()
```

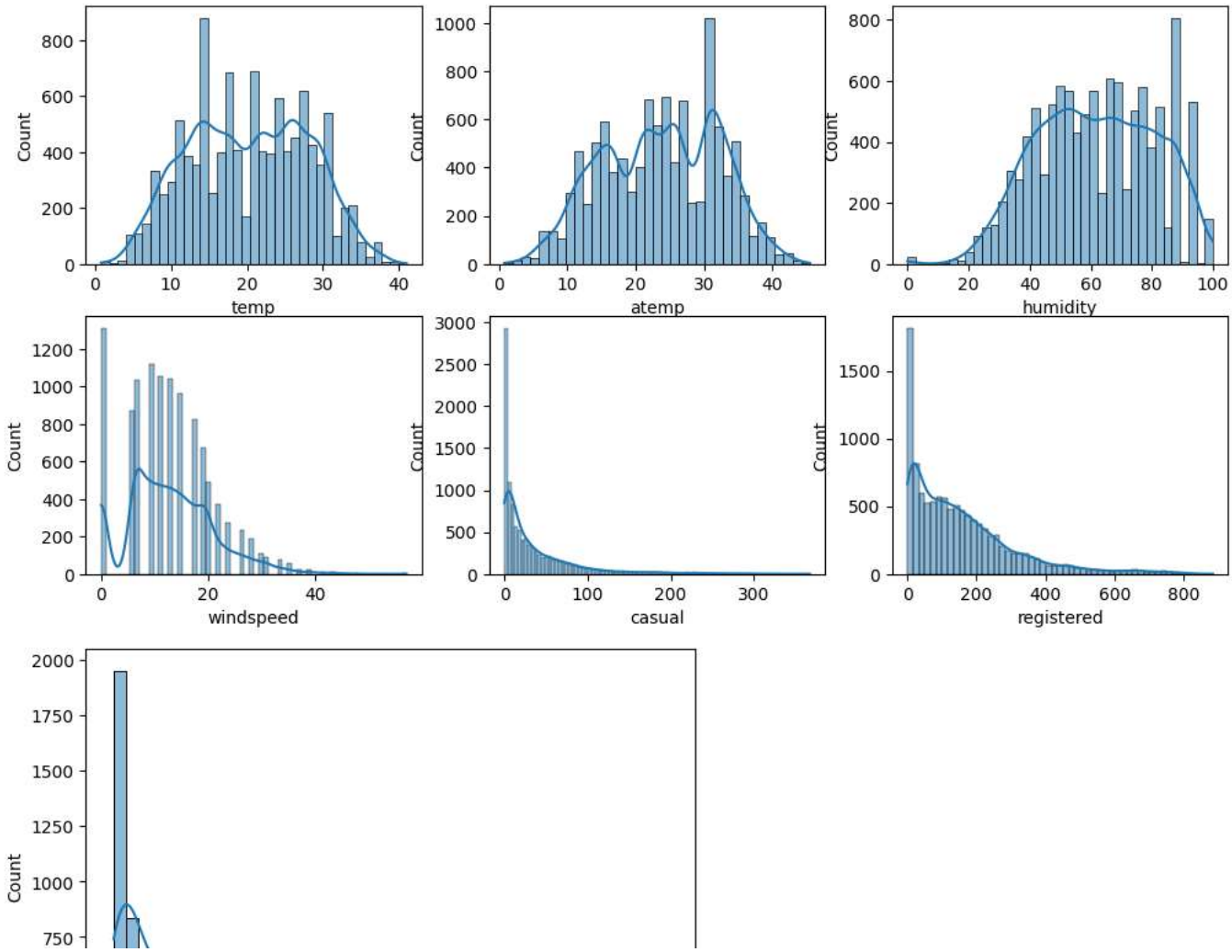
```
datetime      0
season        0
holiday       0
workingday    0
weather       0
temp          0
atemp         0
humidity      0
windspeed     0
casual        0
registered    0
count         0
dtype: int64
```

There are no missing values present in dataset

Establishing a relation between the dependent and independent variable (Dependent "Count" & Independent: Workingday, Weather, Season etc)

1. Univariate Analysis:

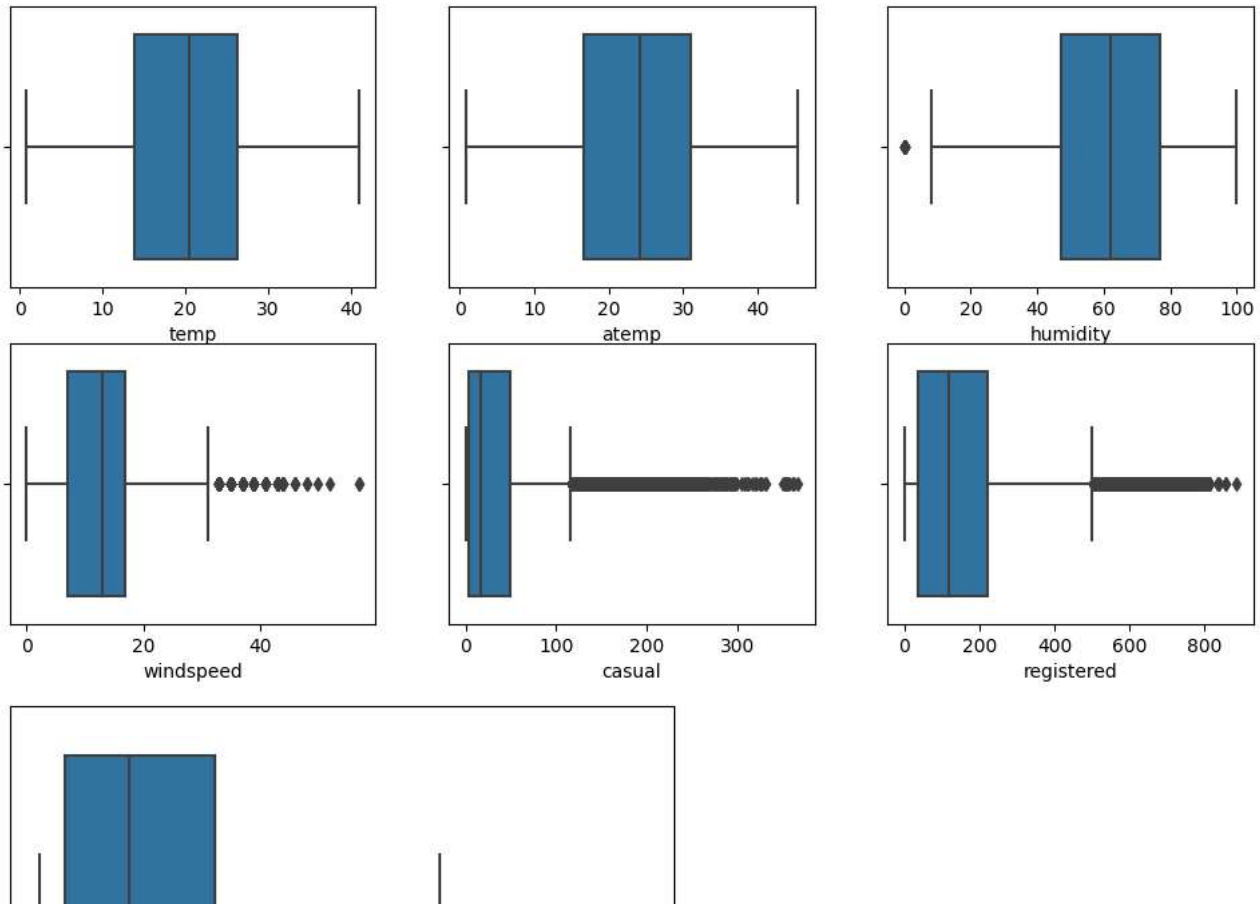
```
# understanding the distribution for numerical variables
cols = ['temp', 'atemp', 'humidity', 'windspeed', 'casual', 'registered','count']

fig, axis = plt.subplots(nrows=2, ncols=3, figsize=(12,6))
index = 0
for row in range(2):
    for col in range(3):
        sns.histplot(data[cols[index]], ax=axis[row, col], kde=True)
        index += 1

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



Observations:

- Casual, registered and count somewhat looks like Log Normal Distribution
- temp, atemp and humidity looks like they follows the Normal Distribution
- windspeed follows the binomial distribution

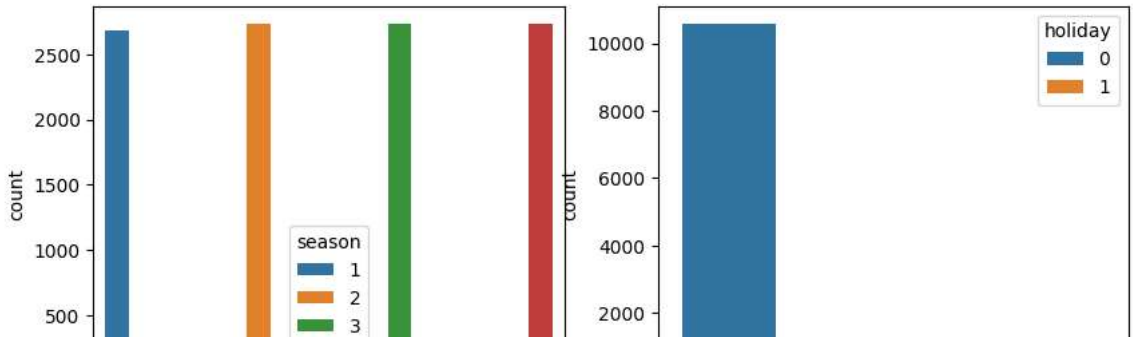
```
# plotting box plots to detect outliers in the data
fig, axis = plt.subplots(nrows=2, ncols=3, figsize=(12, 6))

index = 0
for row in range(2):
    for col in range(3):
        sns.boxplot(x=data[cols[index]], ax=axis[row, col])
        index += 1

plt.show()
sns.boxplot(x=data[cols[-1]])
plt.show()
```



Observations :

- **humidity, casual, registered and count** have outliers in the data.


```
# countplot of each categorical column

cat_cols = ['season','holiday','workingday','weather']
fig, axis = plt.subplots(nrows=2, ncols=2, figsize=(10,8))

index = 0
for row in range(2):
    for col in range(2):
        sns.countplot(data=data, x=cat_cols[index], ax=axis[row, col], hue = cat_cols[index])
        index += 1

plt.show()
```

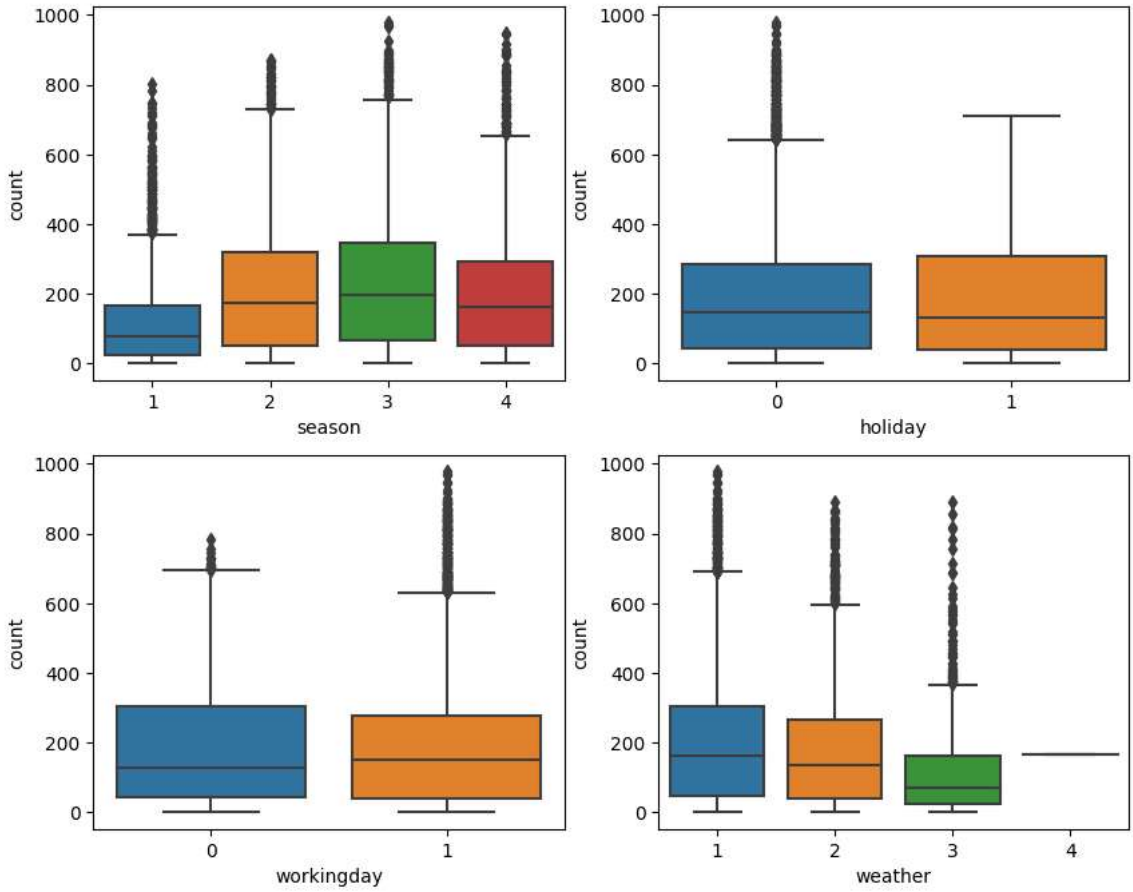


Observations:

- Equal number of days in each season
- More of working days used
- weather is mostly **Clear, Few clouds, partly cloudy, partly cloudy.**

2. Bi-variate Analysis

```
# plotting categorical variables againt count using boxplots
fig, axis = plt.subplots(nrows=2, ncols=2, figsize=(10, 8))

index = 0
for row in range(2):
    for col in range(2):
        sns.boxplot(data=data, x=cat_cols[index], y='count', ax=axis[row, col])
        index += 1

plt.show()
```

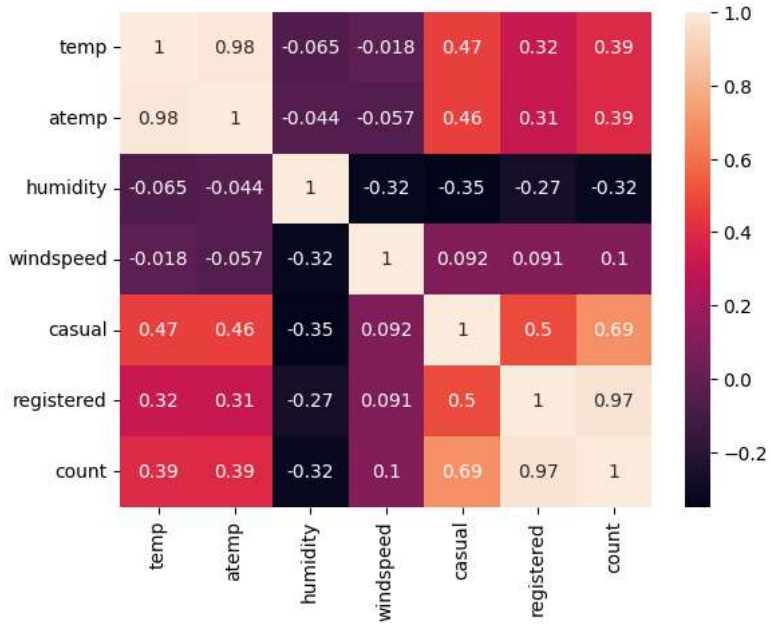


Observations :

- 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.

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

```
<ipython-input-47-5fe07b88f8f7>:2: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version
  data.corr()['count']
<ipython-input-47-5fe07b88f8f7>:3: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version
  sns.heatmap(data.corr(), annot=True)
```



2. 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

```
dg1 = data[data['workingday']==0]['count'].values
dg2 = data[data['workingday']==1]['count'].values

print(np.var(dg1), np.var(dg2))
np.var(dg2)// np.var(dg1)
```

```
30171.346098942427 34040.69710674686
1.0
```

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

```
stats.ttest_ind(a=dg1, b=dg2, equal_var=True)
```

```
TtestResult(statistic=-1.2096277376026694, pvalue=0.22644804226361348, df=10884.0)
```

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

- 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(data['season'], data['weather'])
print("Observed values:")
data_table
```

Observed values:

weather	1	2	3	4
season				
1	1759	715	211	1
2	1801	708	224	0
3	1930	604	199	0
4	1702	807	225	0

```
val = stats.chi2_contingency(data_table)
print(val)
expected_values = val[3]
print(expected_values)
nrows, ncols = 4, 4
dof = (nrows-1)*(ncols-1)
print("degrees of freedom: ", dof)
alpha = 0.05


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}")

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")
```

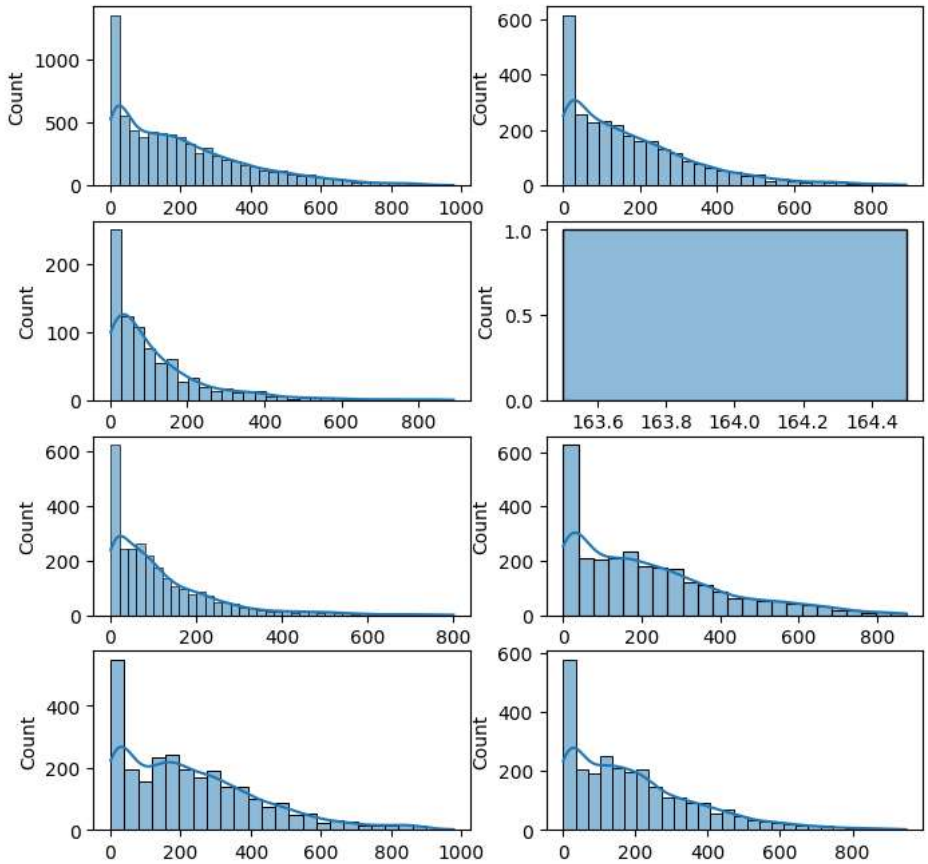
```
Chi2ContingencyResult(statistic=49.158655596893624, pvalue=1.549925073686492e-07, dof=9, expected_freq=array([[1.77454639e+03, 6.9925813
       [1.80559765e+03, 7.11493845e+02, 2.15657450e+02, 2.51056403e-01],
       [1.80559765e+03, 7.11493845e+02, 2.15657450e+02, 2.51056403e-01],
       [1.80625831e+03, 7.11754180e+02, 2.15736359e+02, 2.51148264e-01]]))
[[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-square test statistic:  44.09441248632364
critical value: 16.918977604620448
p-value: 1.3560001579371317e-06

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

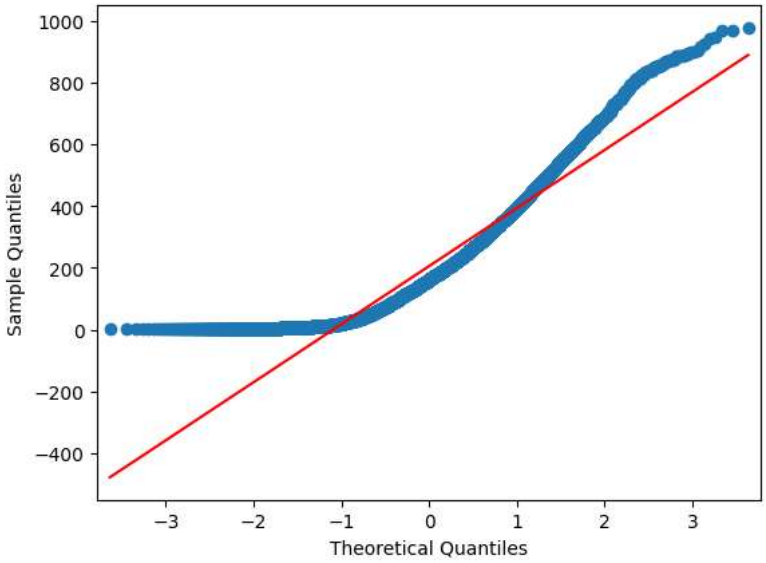
1. weather
2. Season

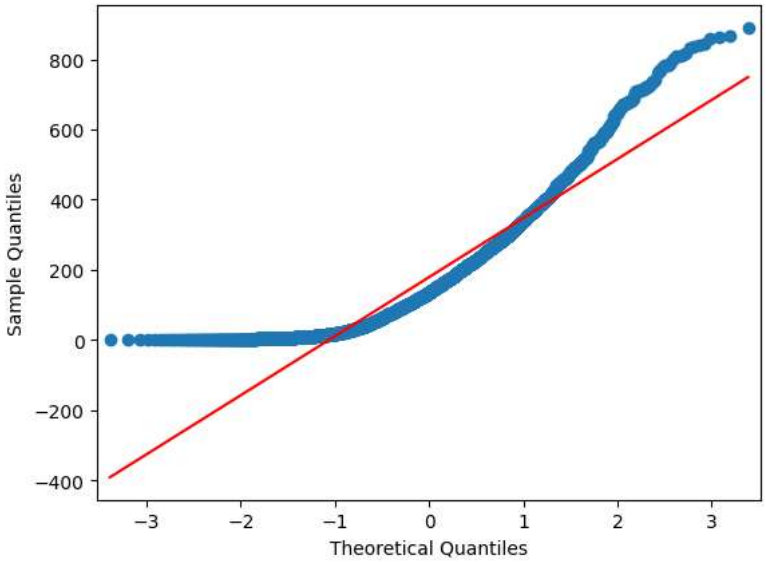
- **Null Hypothesis:** Number of cycles rented is similar in different weather and season.
- **Alternate Hypothesis:** Number of cycles rented is not similar in different weather and season.
- **Significance level (alpha): 0.05**

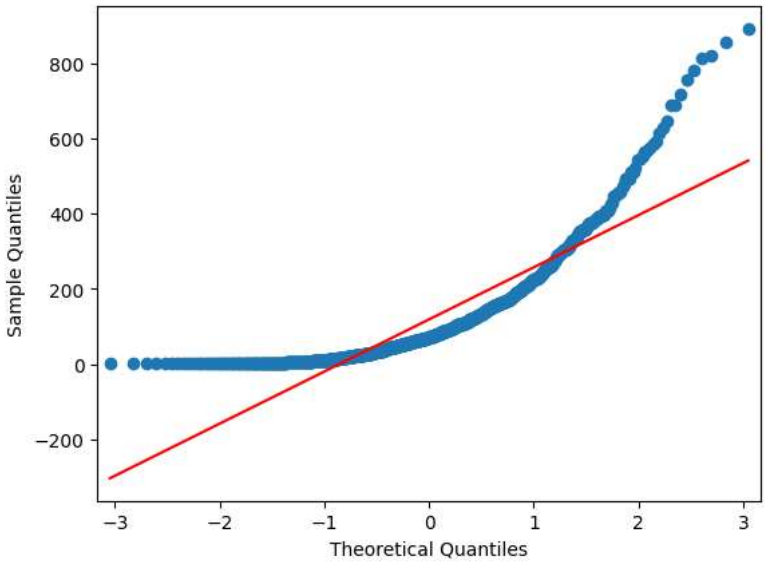
```
# defining the data groups for the ANOVA
from statsmodels.graphics.gofplots import qqplot
gp1 = data[data['weather']==1]['count'].values
gp2 = data[data['weather']==2]['count'].values
gp3 = data[data['weather']==3]['count'].values
gp4 = data[data['weather']==4]['count'].values

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


fig, axis = plt.subplots(nrows=4, ncols=2, figsize=(8, 8))

index = 0
for row in range(4):
    for col in range(2):
        sns.histplot(groups[index], ax=axis[row, col], kde=True)
        index += 1

plt.show()
```



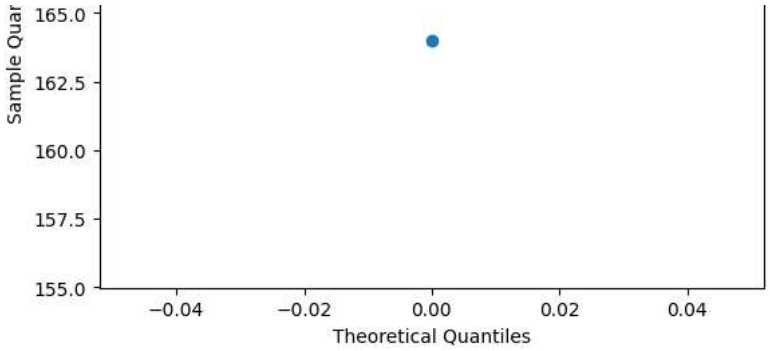

```
index = 0
for row in range(4):
    for col in range(2):
        qqplot(groups[index], line="s")
        index += 1

plt.show()
```

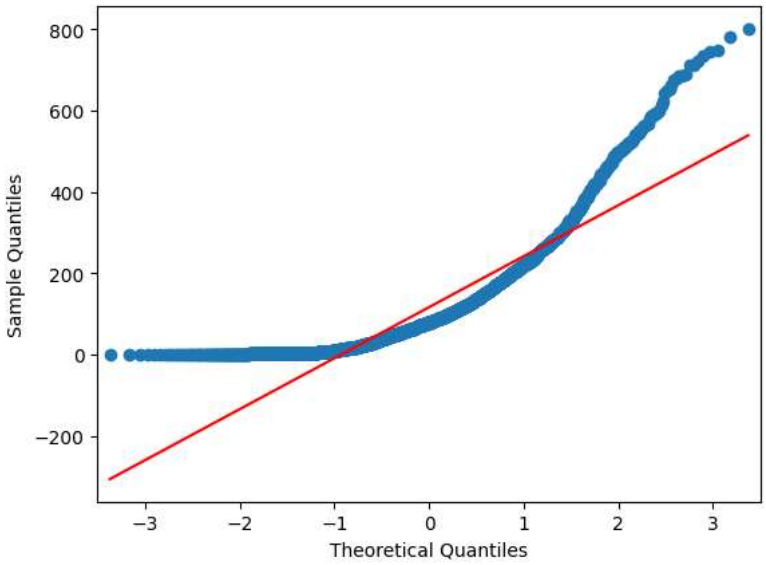

Sample Quantiles
Theoretical Quantiles
1000
800
600
400
200
0
−200
−400
−3
−2
−1
0
1
2
3

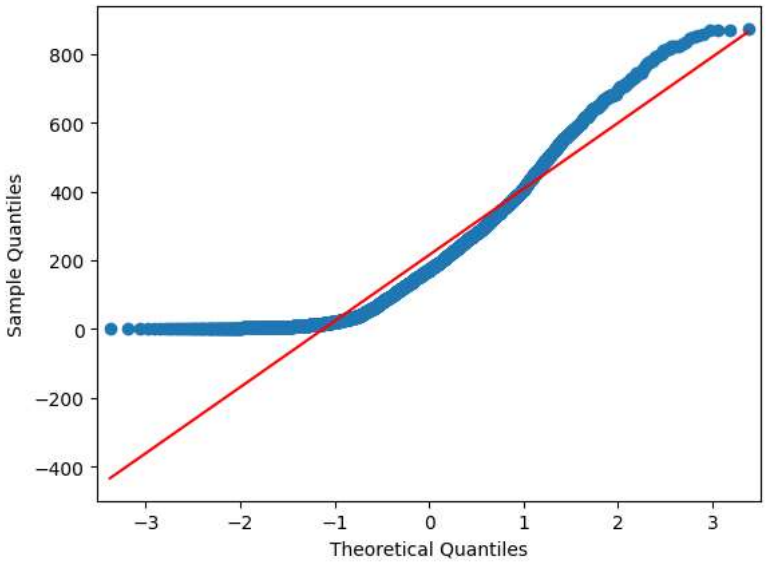

Sample Quantiles
Theoretical Quantiles
800
600
400
200
0
−200
−400
−3
−2
−1
0
1
2
3

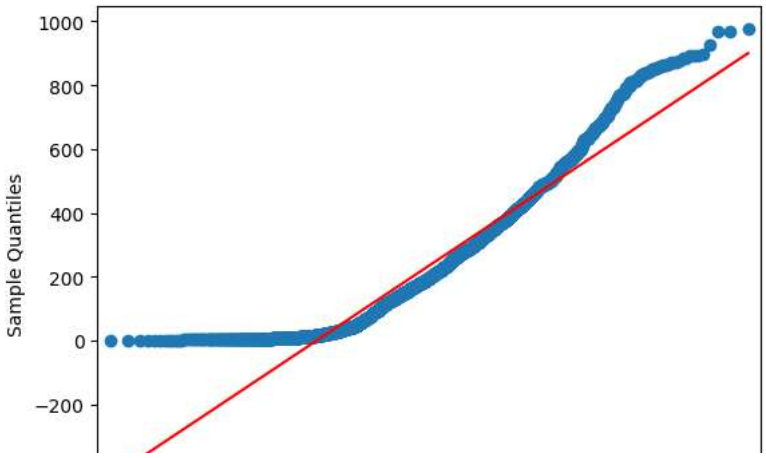

Sample Quantiles
Theoretical Quantiles
800
600
400
200
0
−200
−3
−2
−1
0
1
2
3

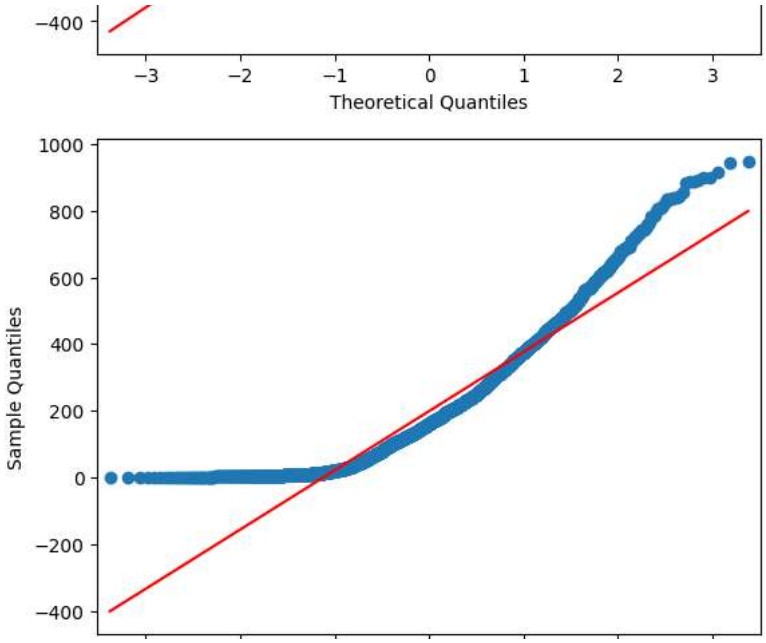

172.5
170.0
167.5


Sample Quar
165.0
162.5
160.0
157.5
155.0
−0.04
−0.02
0.00
0.02
0.04
Theoretical Quantiles


Sample Quantiles
800
600
400
200
0
−200
−3
−2
−1
0
1
2
3
Theoretical Quantiles


Sample Quantiles
800
600
400
200
0
−200
−400
−3
−2
−1
0
1
2
3
Theoretical Quantiles


Sample Quantiles
1000
800
600
400
200
0
−200



As per above graphs, all groups are not following

1. Gaussian distribution
2. Data is Independent
3. 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)
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")
```

```
3.463531888897594e-148
Reject the Null hypothesis.Variances are not equal
```

- As per QQ plot and Levene's Test, We cannot ANOVA Test.
- Assumptions of ANOVA fail, use Kruskal

```
#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
```

Since p-value is less than 0.05, we reject the null hypothesis. This implies that Number of cycles rented is not similar in different weather and season conditions

Insights

- 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.
- 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.

Recommendations

- 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.
- With a significance level of 0.05, workingday has no effect on the number of bikes being rented.
- In very low humid days, company should have less bikes in the stock to be rented.
- Whenever temperature is less than 10 or in very cold days, company should have less bikes.
- Whenever the windspeed is greater than 35 or in thunderstorms, company should have less bikes in stock to be rented.