Business case: Yulu - Hypothesis Testing

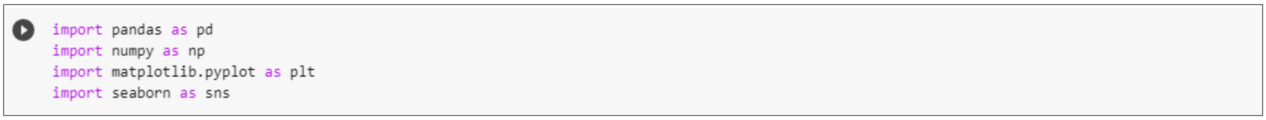
Date: 31-07-2023

Problem statement:

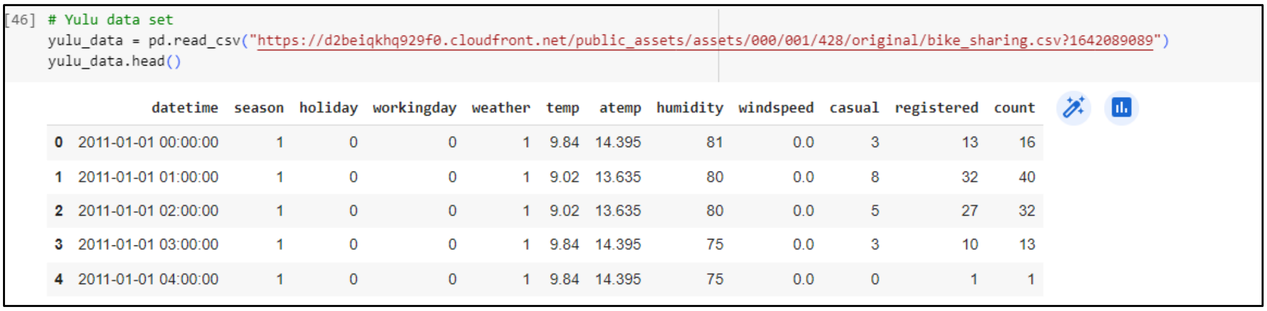
The company wants to know,

- Which variables are significant in predicting the demand for shared electric cycles in the Indian market
- How well those variables desribe the electric cycles demands

----

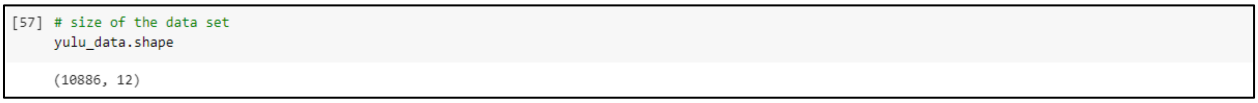


```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

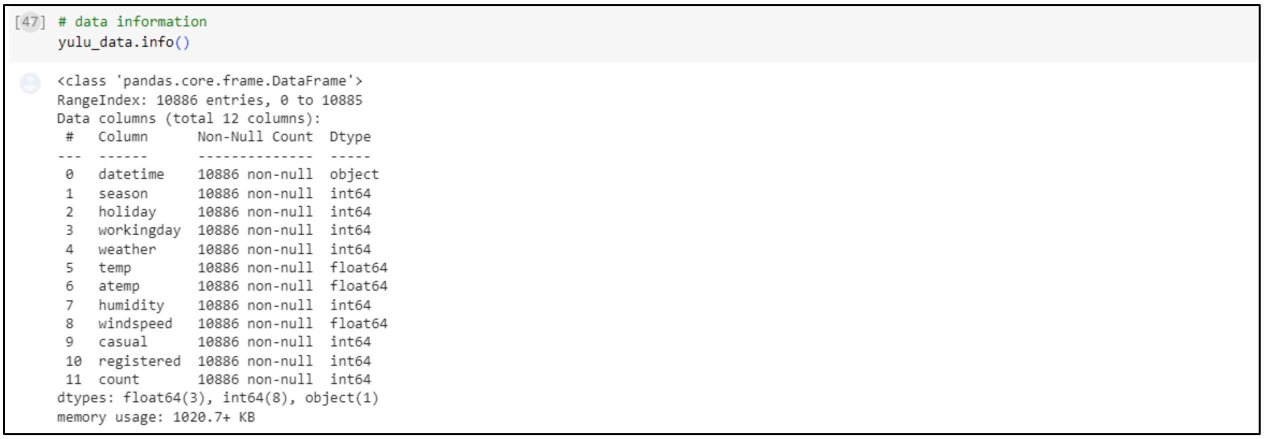


```
[46] # Yulu data set
     yulu_data = pd.read_csv("https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/428/original/bike_sharing.csv?1642089089")
     yulu_data.head()
```

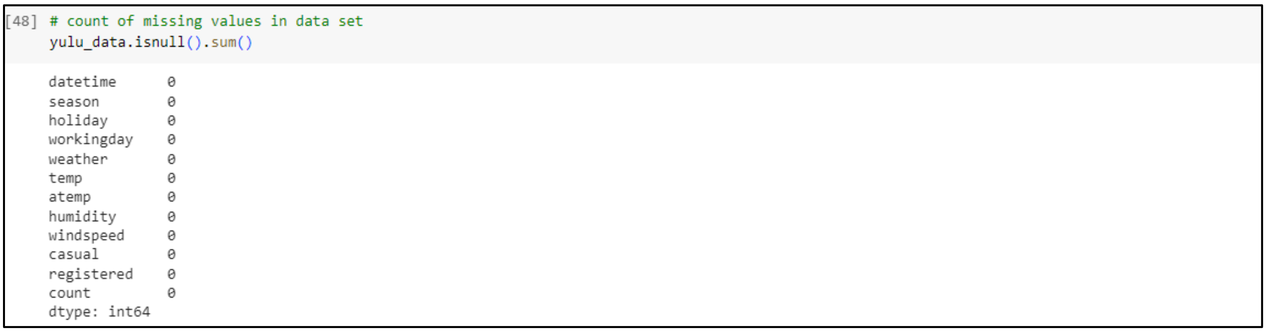
	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	count
0	2011-01-01 00:00:00	1	0	0	1	9.84	14.395	81	0.0	3	13	16
1	2011-01-01 01:00:00	1	0	0	1	9.02	13.635	80	0.0	8	32	40
2	2011-01-01 02:00:00	1	0	0	1	9.02	13.635	80	0.0	5	27	32
3	2011-01-01 03:00:00	1	0	0	1	9.84	14.395	75	0.0	3	10	13
4	2011-01-01 04:00:00	1	0	0	1	9.84	14.395	75	0.0	0	1	1



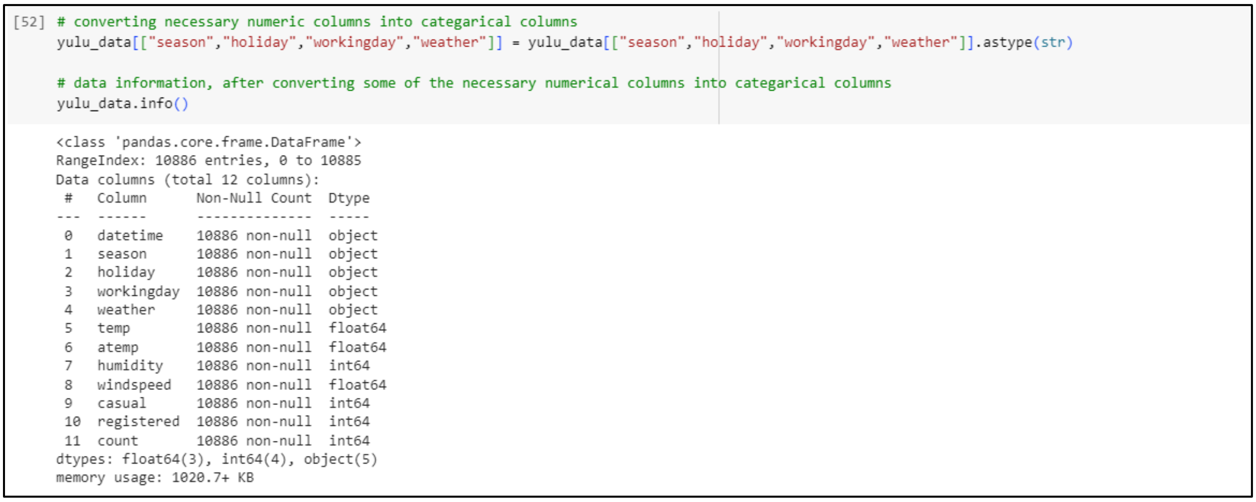
```
[57] # size of the data set
     yulu_data.shape

     (10886, 12)
```



```
[47] # data information
     yulu_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
```

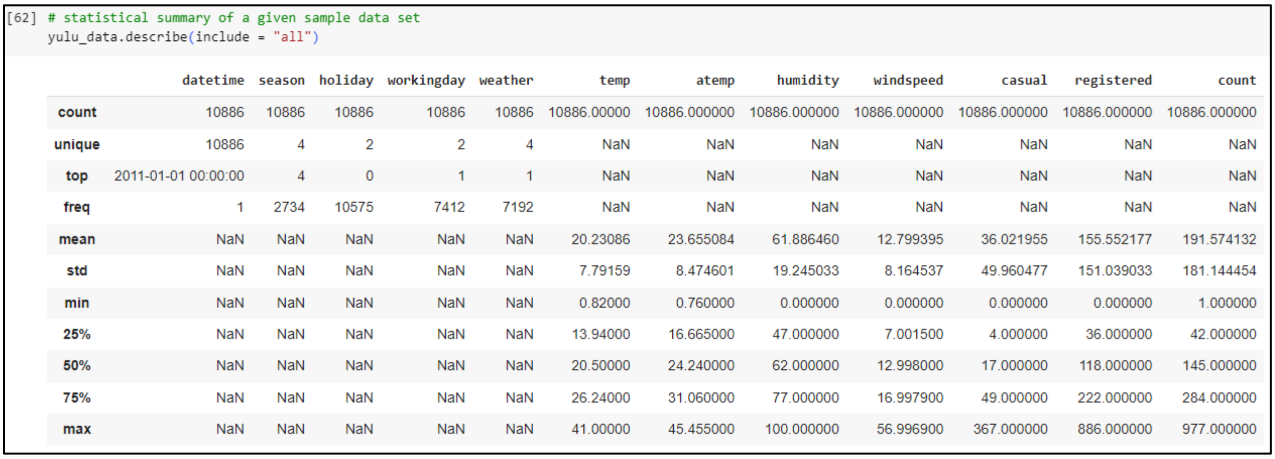


```
[48] # count of missing values in data set
     yulu_data.isnull().sum()

     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
```



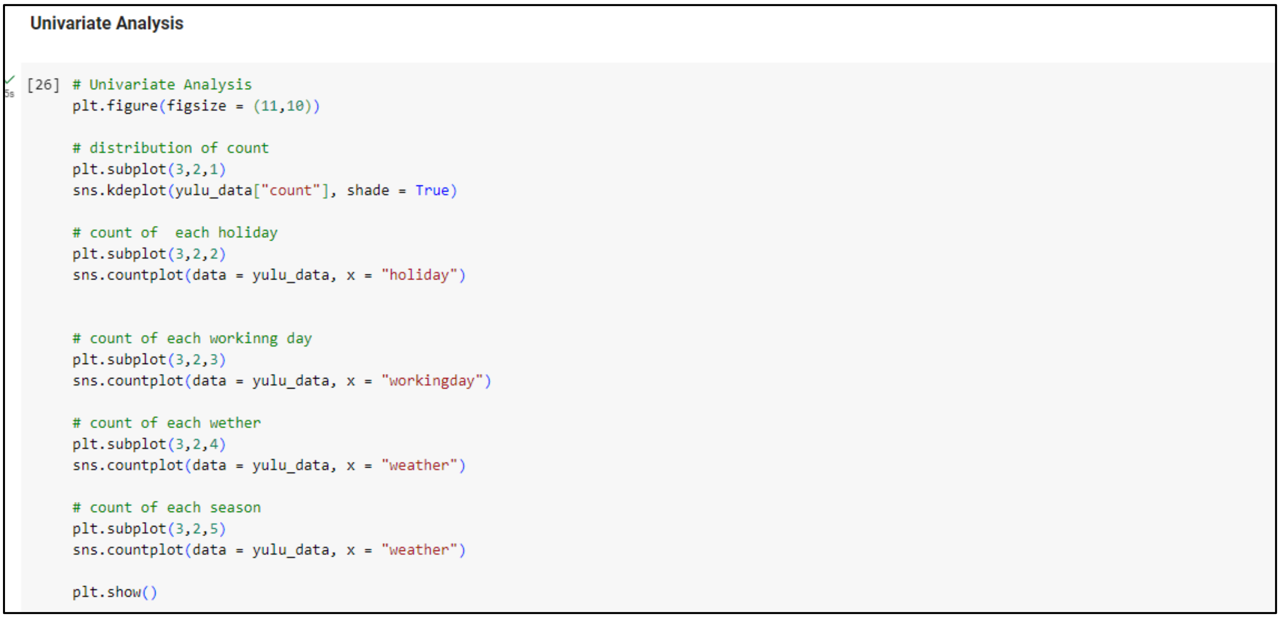
```
[52] # converting necessary numeric columns into categarical columns
     yulu_data[["season","holiday","workingday","weather"]] = yulu_data[["season","holiday","workingday","weather"]].astype(str)

     # data information, after converting some of the necessary numerical columns into categarical columns
     yulu_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  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: float64(3), int64(4), object(5)
memory usage: 1020.7+ KB
```



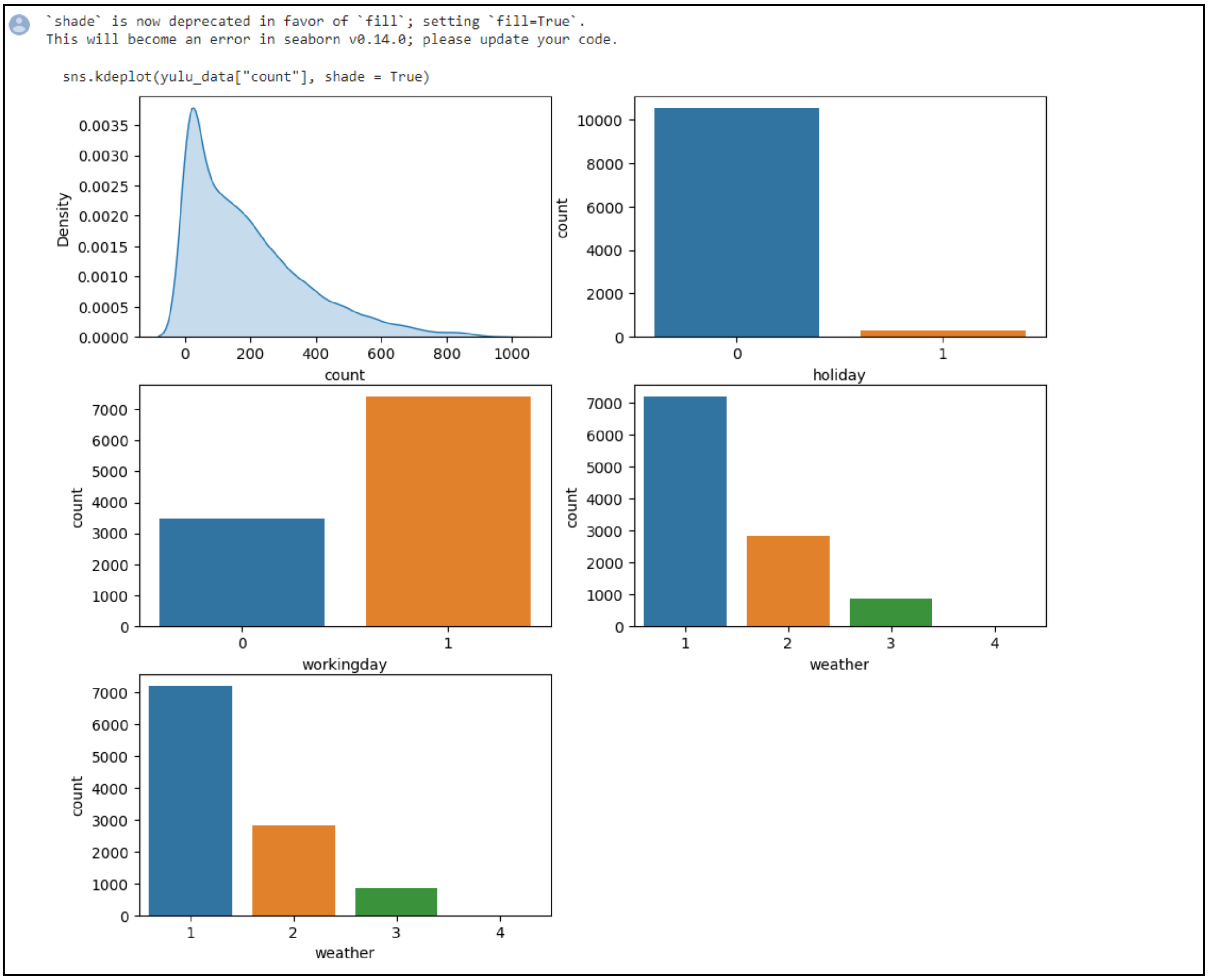
```
[62] # statistical summary of a given sample data set
     yulu_data.describe(include = "all")
```

	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	count
count	10886	10886	10886	10886	10886	10886.00000	10886.000000	10886.000000	10886.000000	10886.000000	10886.000000	10886.000000
unique	10886	4	2	2	4	NaN	NaN	NaN	NaN	NaN	NaN	NaN
top	2011-01-01 00:00:00	4	0	1	1	NaN	NaN	NaN	NaN	NaN	NaN	NaN
freq	1	2734	10575	7412	7192	NaN	NaN	NaN	NaN	NaN	NaN	NaN
mean	NaN	NaN	NaN	NaN	NaN	20.23086	23.655084	61.886460	12.799395	36.021955	155.552177	191.574132
std	NaN	NaN	NaN	NaN	NaN	7.79159	8.474601	19.245033	8.164537	49.960477	151.039033	181.144454
min	NaN	NaN	NaN	NaN	NaN	0.82000	0.760000	0.000000	0.000000	0.000000	0.000000	1.000000
25%	NaN	NaN	NaN	NaN	NaN	13.94000	16.665000	47.000000	7.001500	4.000000	36.000000	42.000000
50%	NaN	NaN	NaN	NaN	NaN	20.50000	24.240000	62.000000	12.998000	17.000000	118.000000	145.000000
75%	NaN	NaN	NaN	NaN	NaN	26.24000	31.060000	77.000000	16.997900	49.000000	222.000000	284.000000
max	NaN	NaN	NaN	NaN	NaN	41.00000	45.455000	100.000000	56.996900	367.000000	886.000000	977.000000

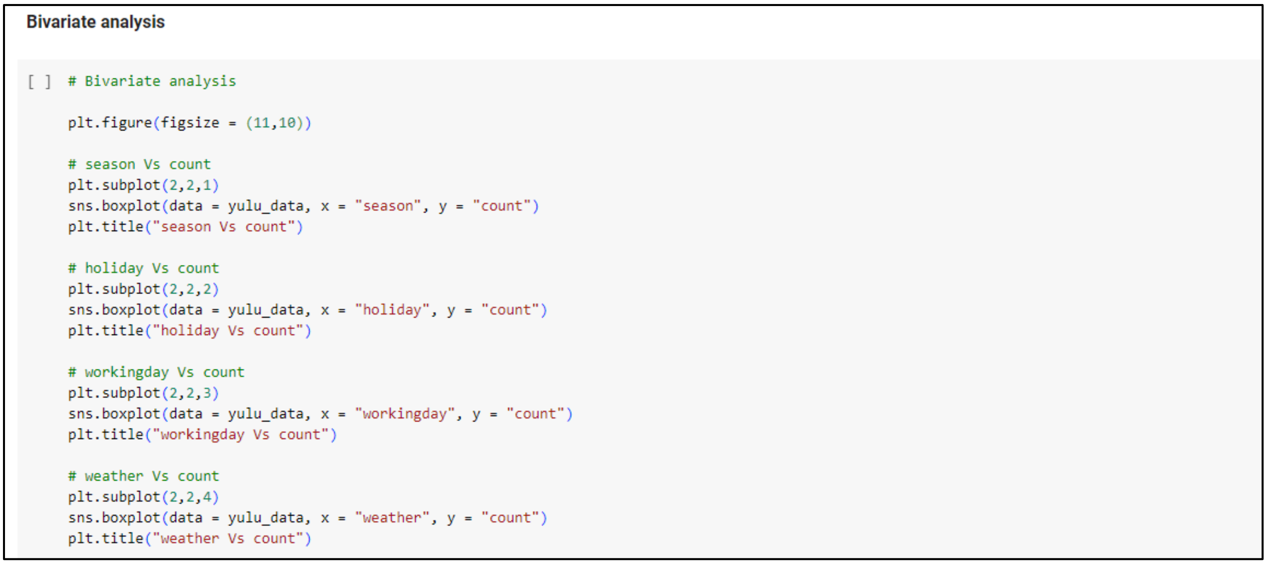


Univariate Analysis

```
[26] # Univariate Analysis
     plt.figure(figsize = (11,10))

     # distribution of count
     plt.subplot(3,2,1)
     sns.kdeplot(yulu_data["count"], shade = True)

     # count of  each holiday
     plt.subplot(3,2,2)
     sns.countplot(data = yulu_data, x = "holiday")


     # count of each workinng day
     plt.subplot(3,2,3)
     sns.countplot(data = yulu_data, x = "workingday")

     # count of each wether
     plt.subplot(3,2,4)
     sns.countplot(data = yulu_data, x = "weather")

     # count of each season
     plt.subplot(3,2,5)
     sns.countplot(data = yulu_data, x = "weather")

     plt.show()
```



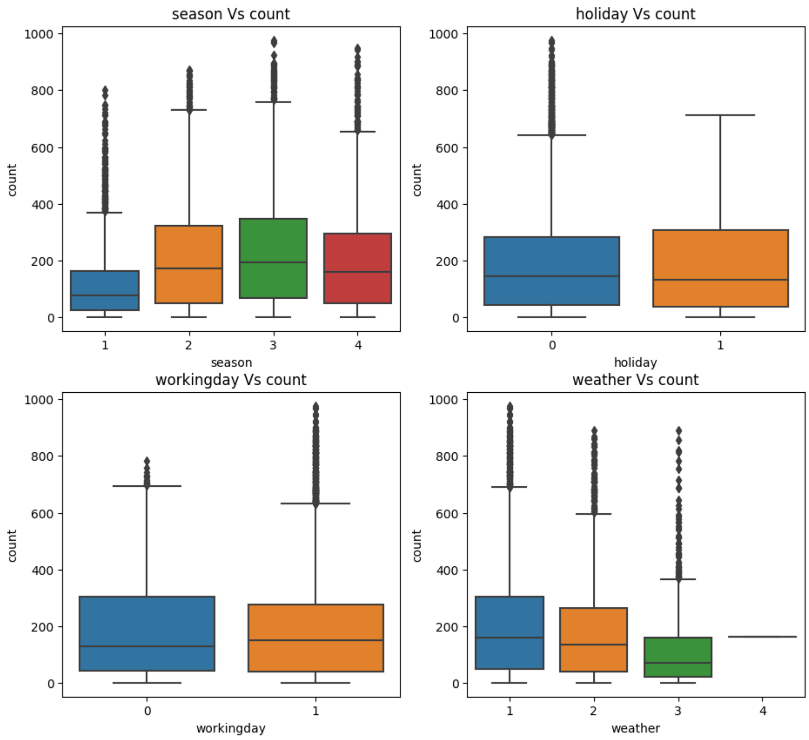
```
`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

  sns.kdeplot(yulu_data["count"], shade = True)
```



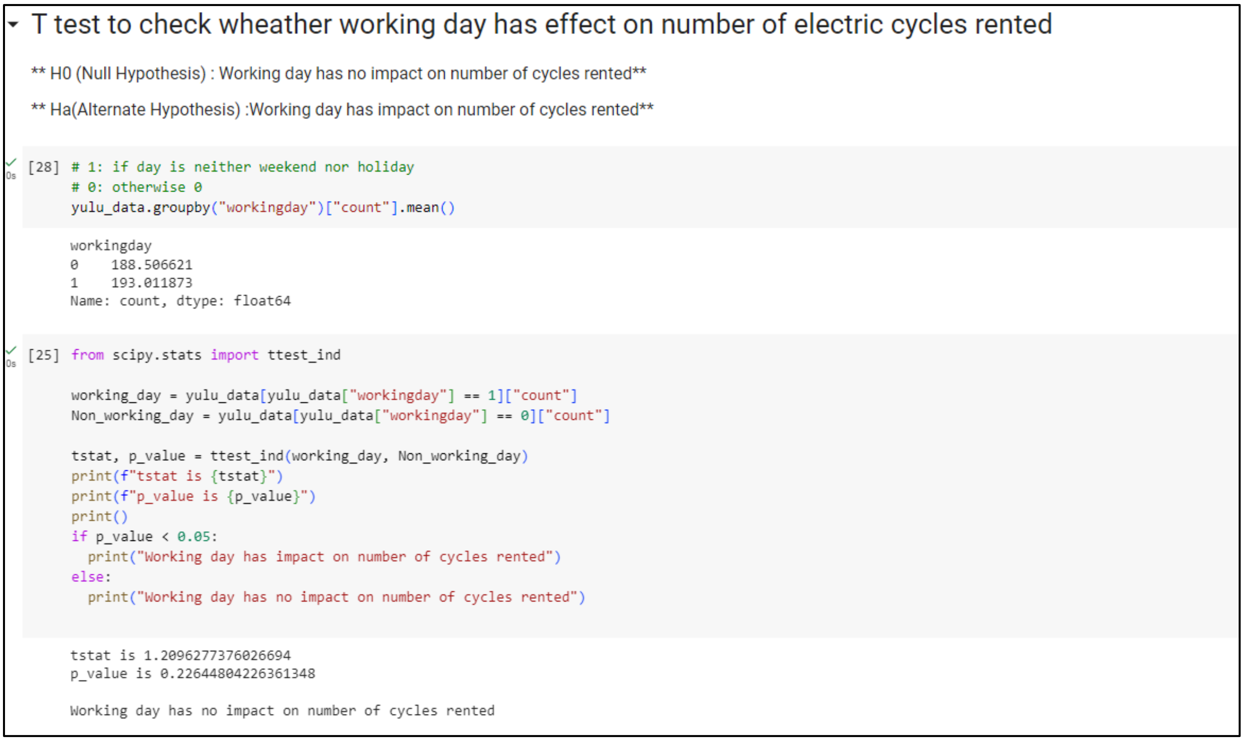
Bivariate analysis

```
[ ] # Bivariate analysis

    plt.figure(figsize = (11,10))

    # season Vs count
    plt.subplot(2,2,1)
    sns.boxplot(data = yulu_data, x = "season", y = "count")
    plt.title("season Vs count")

    # holiday Vs count
    plt.subplot(2,2,2)
    sns.boxplot(data = yulu_data, x = "holiday", y = "count")
    plt.title("holiday Vs count")

    # workingday Vs count
    plt.subplot(2,2,3)
    sns.boxplot(data = yulu_data, x = "workingday", y = "count")
    plt.title("workingday Vs count")

    # weather Vs count
    plt.subplot(2,2,4)
    sns.boxplot(data = yulu_data, x = "weather", y = "count")
    plt.title("weather Vs count")
```

```
Text(0.5, 1.0, 'weather Vs count')
```



1.Does workig day have impact on number of cycles rented?.



T test to check wheather working day has effect on number of electric cycles rented

** H0 (Null Hypothesis) : Working day has no impact on number of cycles rented**

** Ha(Alternate Hypothesis) :Working day has impact on number of cycles rented**

```
[28] # 1: if day is neither weekend nor holiday
     # 0: otherwise 0
     yulu_data.groupby("workingday")["count"].mean()
```

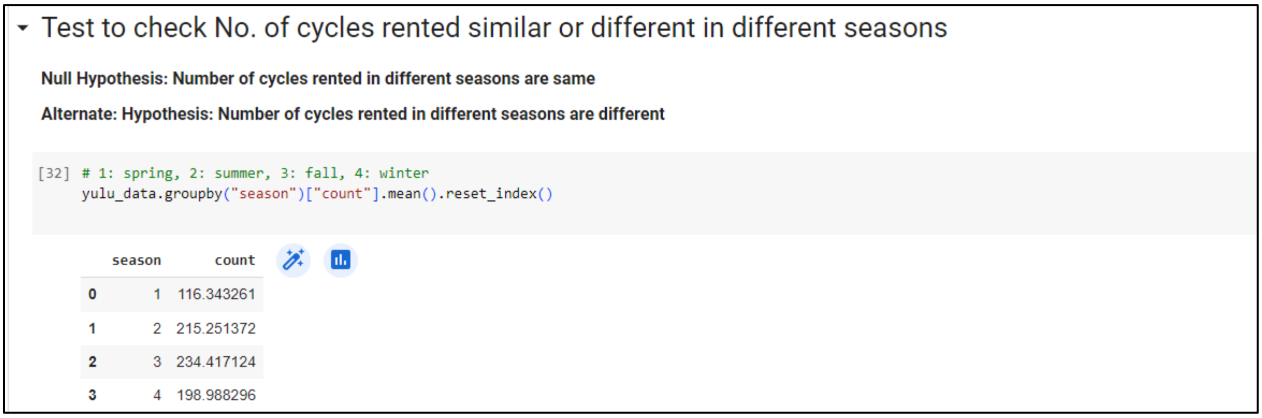
```
workingday
0    188.506621
1    193.011873
Name: count, dtype: float64
```

```
[25] from scipy.stats import ttest_ind

     working_day = yulu_data[yulu_data["workingday"] == 1]["count"]
     Non_working_day = yulu_data[yulu_data["workingday"] == 0]["count"]

     tstat, p_value = ttest_ind(working_day, Non_working_day)
     print(f"tstat is {tstat}")
     print(f"p_value is {p_value}")
     print()
     if p_value < 0.05:
       print("Working day has impact on number of cycles rented")
     else:
       print("Working day has no impact on number of cycles rented")
```

```
tstat is 1.2096277376026694
p_value is 0.22644804226361348

Working day has no impact on number of cycles rented
```

From the above 2- sample t test with 5% significance level, mean number of cycles rented on working day is greater than that of non-working day in the sample (Holiday or Weekends) is happened by chance, not significant. Hence, working day has no impact on number of cycles rented

2. Does season have impact on number of cycles rented?.



Test to check No. of cycles rented similar or different in different seasons

Null Hypothesis: Number of cycles rented in different seasons are same

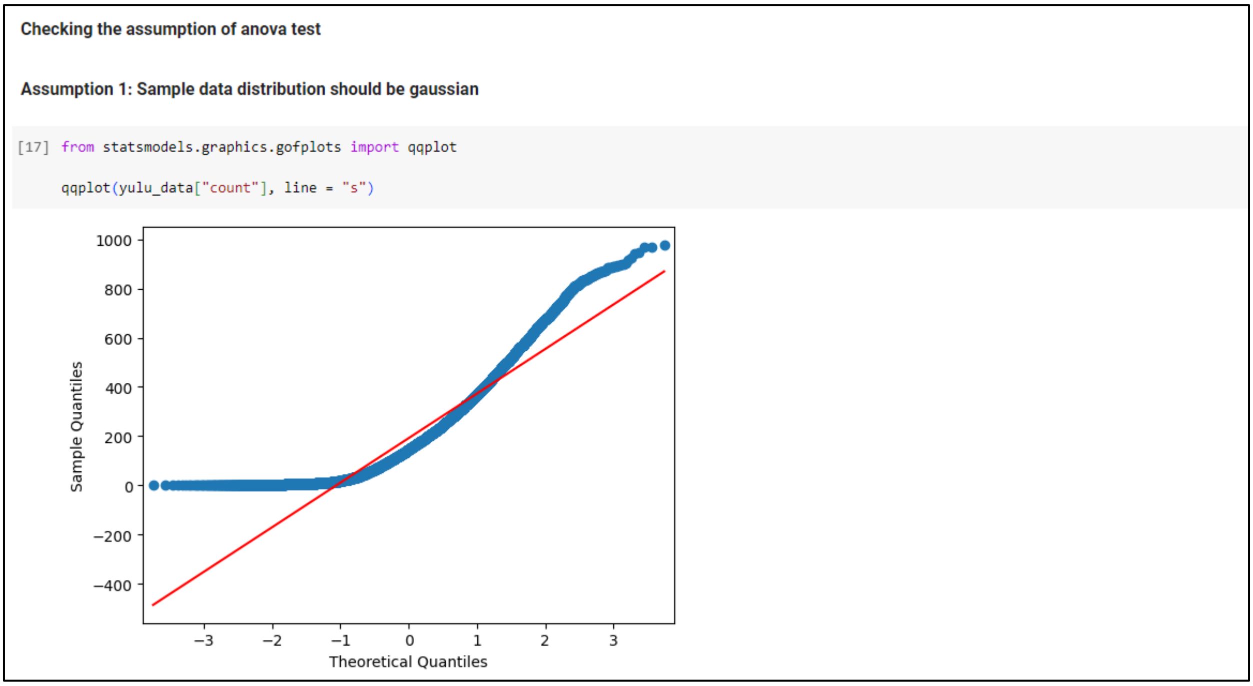
Alternate: Hypothesis: Number of cycles rented in different seasons are different

```
[32] # 1: spring, 2: summer, 3: fall, 4: winter
     yulu_data.groupby("season")["count"].mean().reset_index()
```

	season	count
0	1	116.343261
1	2	215.251372
2	3	234.417124
3	4	198.988296

As shown above, mean of number of cycles rented in different seasons are different. But whether it happened by chance or it is significant is tested below using appropriate test.

Since more than two categorical variables, anova test is used to check the impact of season on number of cycles rented. Before performing the anova test, one has to check whether assumptions of anova test are satisfied or not.

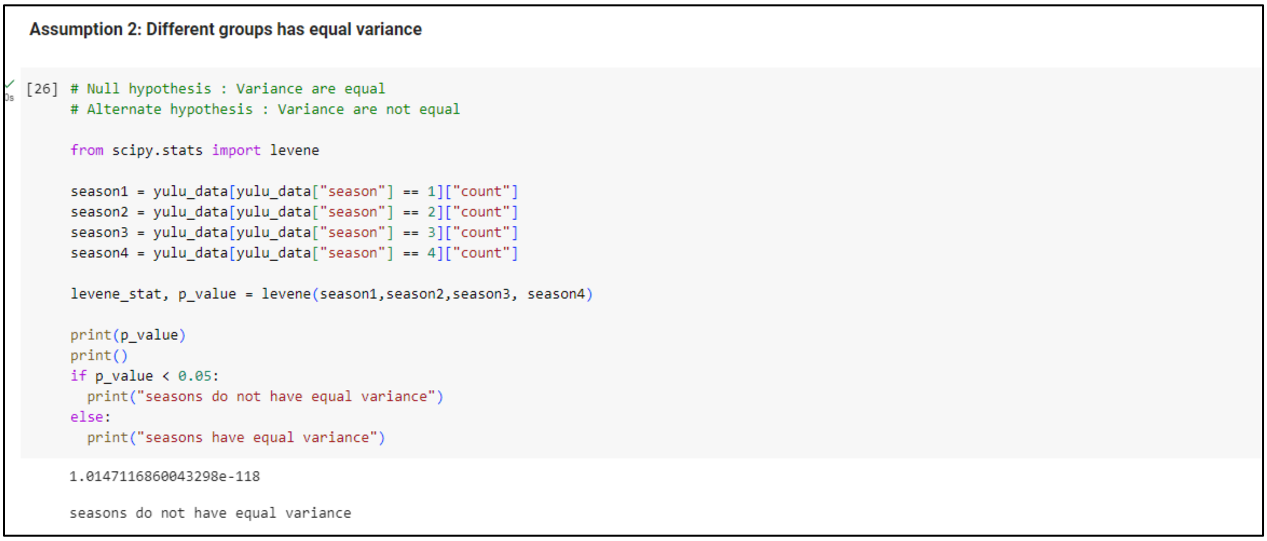


Checking the assumption of anova test

Assumption 1: Sample data distribution should be gaussian

```
[17] from statsmodels.graphics.gofplots import qqplot

     qqplot(yulu_data["count"], line = "s")
```

In the above chart, red linear line is the Gaussian line & blue non-linear curve is the curve of percentiles of sample data Vs percentiles of Gaussian data. Since both the lines are not aligned (curve is non-linear), sample data distribution is not Gaussian. Hence Anova test cannot be performed.



Assumption 2: Different groups has equal variance

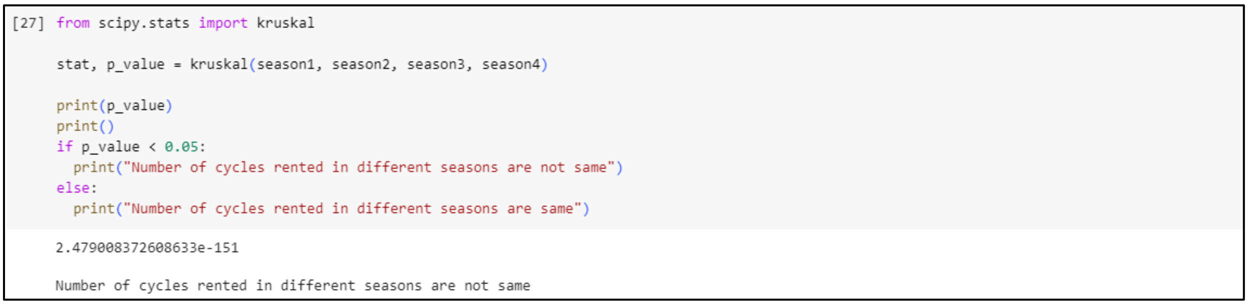
```
[26] # Null hypothesis : Variance are equal
     # Alternate hypothesis : Variance are not equal

     from scipy.stats import levene

     season1 = yulu_data[yulu_data["season"] == 1]["count"]
     season2 = yulu_data[yulu_data["season"] == 2]["count"]
     season3 = yulu_data[yulu_data["season"] == 3]["count"]
     season4 = yulu_data[yulu_data["season"] == 4]["count"]

     levene_stat, p_value = levene(season1,season2,season3, season4)

     print(p_value)
     print()
     if p_value < 0.05:
       print("seasons do not have equal variance")
     else:
       print("seasons have equal variance")
```

```
1.0147116860043298e-118

seasons do not have equal variance
```

As shown above, levene test performed to check whether seasons have equal variance or not with 5% significant level. Since p_value < 0.05, seasons do not have equal variance. Assumption of equal variance is not satisfied which implies that Anova test cannot be performed.

And hence kruskal test is performed to check Number of cycles rented similar or different in seasons.

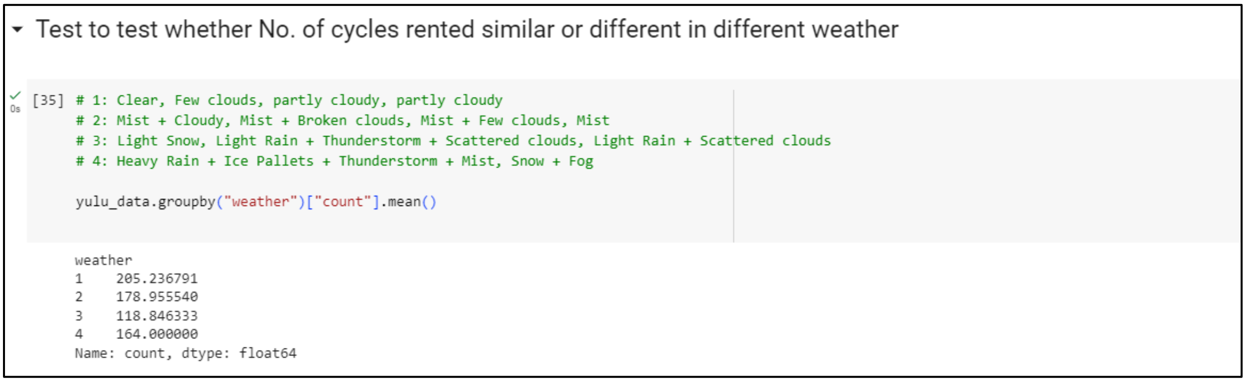
Kruskal Test



```
[27] from scipy.stats import kruskal

     stat, p_value = kruskal(season1, season2, season3, season4)

     print(p_value)
     print()
     if p_value < 0.05:
       print("Number of cycles rented in different seasons are not same")
     else:
       print("Number of cycles rented in different seasons are same")

     2.479008372608633e-151

     Number of cycles rented in different seasons are not same
```

Kruskal test is performed with 5% significance level and concluded that difference in mean of number of cycles rented in seasons is not happened by chance, it is significant.

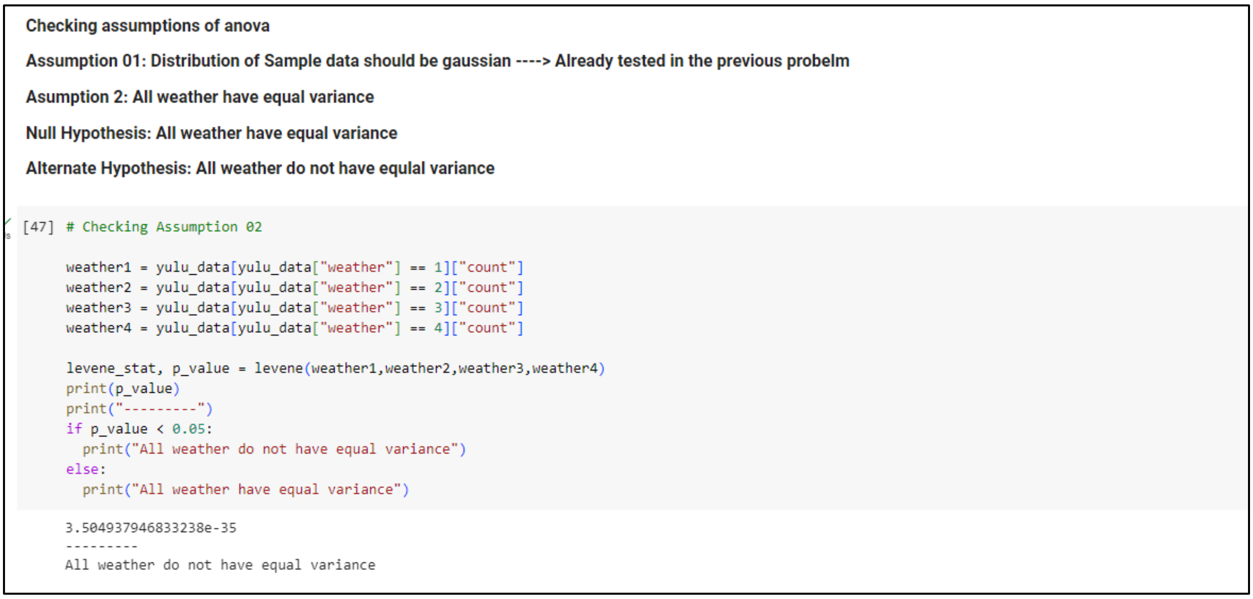
3. Does weather have impact on number of cycles rented?.



Test to test whether No. of cycles rented similar or different in different weather

```
[35] # 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

     yulu_data.groupby("weather")["count"].mean()

     weather
     1    205.236791
     2    178.955540
     3    118.846333
     4    164.000000
     Name: count, dtype: float64
```

From the above, mean number of cycles rented for different weather conditions are different



Checking assumptions of anova

Assumption 01: Distribution of Sample data should be gaussian ----> Already tested in the previous probelm

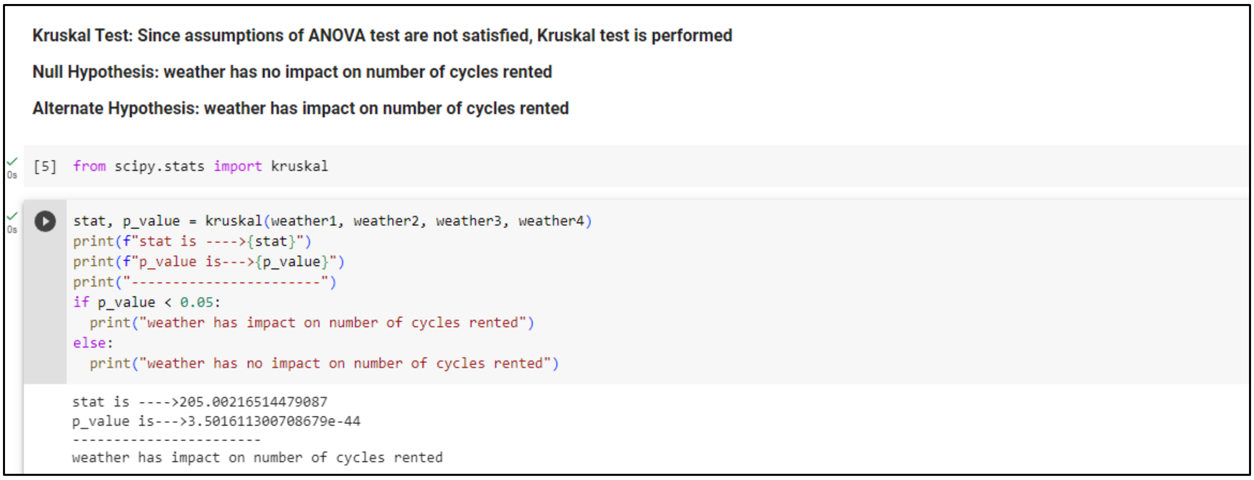
Asumption 2: All weather have equal variance

Null Hypothesis: All weather have equal variance

Alternate Hypothesis: All weather do not have equlal variance

```
[47] # Checking Assumption 02

     weather1 = yulu_data[yulu_data["weather"] == 1]["count"]
     weather2 = yulu_data[yulu_data["weather"] == 2]["count"]
     weather3 = yulu_data[yulu_data["weather"] == 3]["count"]
     weather4 = yulu_data[yulu_data["weather"] == 4]["count"]

     levene_stat, p_value = levene(weather1,weather2,weather3,weather4)
     print(p_value)
     print("---------")
     if p_value < 0.05:
       print("All weather do not have equal variance")
     else:
       print("All weather have equal variance")

     3.504937946833238e-35
     ---------
     All weather do not have equal variance
```

Since assumptions of Gaussian distribution & equal variance are not satisfied, kruskal test is performed to check the impact of weather on number of cycles rented.



Kruskal Test: Since assumptions of ANOVA test are not satisfied, Kruskal test is performed

Null Hypothesis: weather has no impact on number of cycles rented

Alternate Hypothesis: weather has impact on number of cycles rented

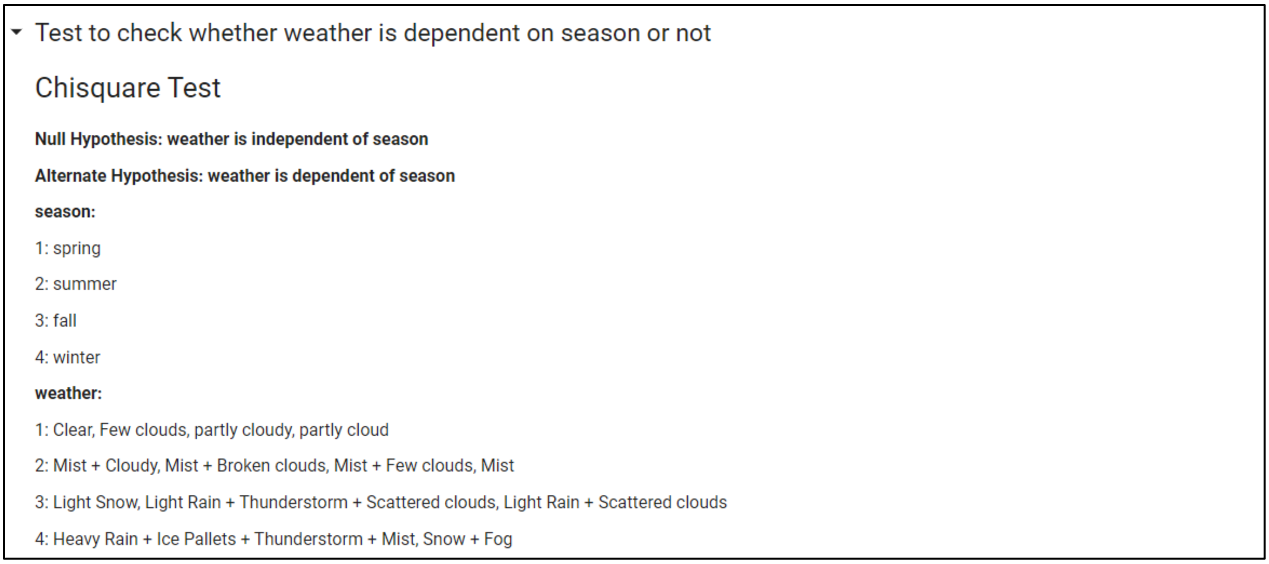
```
[5] from scipy.stats import kruskal
```

```
stat, p_value = kruskal(weather1, weather2, weather3, weather4)
print(f"stat is ---->{stat}")
print(f"p_value is--->{p_value}")
print("-----------------------")
if p_value < 0.05:
  print("weather has impact on number of cycles rented")
else:
  print("weather has no impact on number of cycles rented")
```

```
stat is ---->205.00216514479087
p_value is--->3.501611300708679e-44
-----------------------
weather has impact on number of cycles rented
```

From the above, it is concluded that weather has impact on number of cycles rented

4. Is weather dependent of season?.



Test to check whether weather is dependent on season or not

Chisquare Test

Null Hypothesis: weather is independent of season

Alternate Hypothesis: weather is dependent of season

season:

1: spring

2: summer

3: fall

4: winter

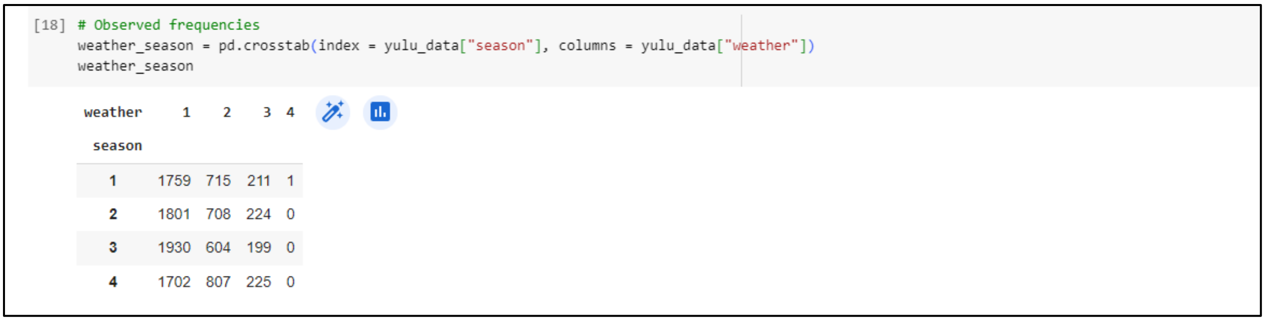
weather:

1: Clear, Few clouds, partly cloudy, partly cloud

2: Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist

3: Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered clouds

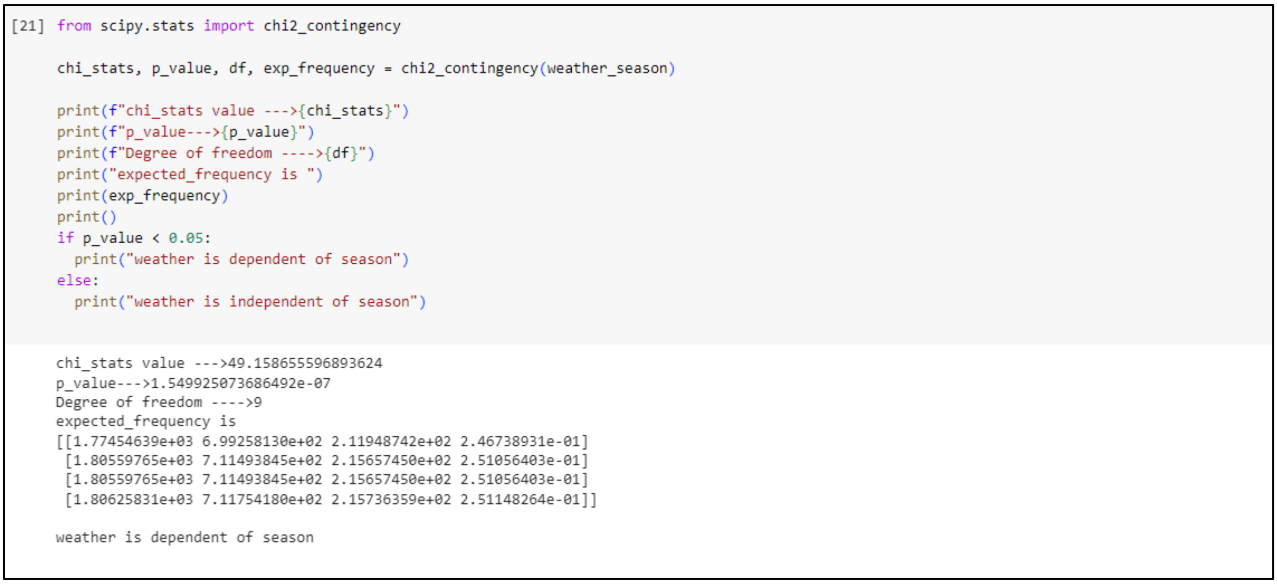
4: Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog



```
[18] # Observed frequencies
     weather_season = pd.crosstab(index = yulu_data["season"], columns = yulu_data["weather"])
     weather_season
```

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

As shown above, weather depends on season for the given sample data. But does it true for the population?, it will be tested below using chi-square test.



```
[21] from scipy.stats import chi2_contingency

     chi_stats, p_value, df, exp_frequency = chi2_contingency(weather_season)

     print(f"chi_stats value --->{chi_stats}")
     print(f"p_value--->{p_value}")
     print(f"Degree of freedom ---->{df}")
     print("expected_frequency is ")
     print(exp_frequency)
     print()
     if p_value < 0.05:
       print("weather is dependent of season")
     else:
       print("weather is independent of season")
```

```
chi_stats value --->49.158655596893624
p_value--->1.549925073686492e-07
Degree of freedom ---->9
expected_frequency is
[[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]]

weather is dependent of season
```

Chi square test performed with significance level of 5%, it is concluded that dependence of weather on season for the given sample data is not by chance, it is significant.

As proved in the previous tests, Number of cycles rented is impacted by weather and also by season. Since weather is depending on season (with 95% confidence), may be only one of these two features is required to build a model (feature engineering is out of scope here).