yulu-case-study-2

July 21, 2024

1 1. Define the Problem Statement, Import the required Libraries and perform Exploratory Data Analysis.

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

[ ]:
```
data=pd.read_csv("/content/bike_sharing.csv")
```

[ ]:
```
data.head()
```

[ ]:
```
              datetime  season  holiday  workingday  weather  temp   atemp  \
0  2011-01-01 00:00:00       1        0           0        1  9.84  14.395
1  2011-01-01 01:00:00       1        0           0        1  9.02  13.635
2  2011-01-01 02:00:00       1        0           0        1  9.02  13.635
3  2011-01-01 03:00:00       1        0           0        1  9.84  14.395
4  2011-01-01 04:00:00       1        0           0        1  9.84  14.395

   humidity  windspeed  casual  registered  count
0        81        0.0       3          13     16
1        80        0.0       8          32     40
2        80        0.0       5          27     32
3        75        0.0       3          10     13
4        75        0.0       0           1      1
```

[ ]:
```
data.shape
```

[ ]: (10886, 12)

[ ]:
```
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
```

[ ]:
```
data.isna().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
```

[ ]:
```
data.describe()
```

[ ]:
```
             season       holiday    workingday       weather         temp  \
count  10886.000000  10886.000000  10886.000000  10886.000000  10886.00000
mean       2.506614      0.028569      0.680875      1.418427     20.23086
std        1.116174      0.166599      0.466159      0.633839      7.79159
min        1.000000      0.000000      0.000000      1.000000      0.82000
25%        2.000000      0.000000      0.000000      1.000000     13.94000
50%        3.000000      0.000000      1.000000      1.000000     20.50000
75%        4.000000      0.000000      1.000000      2.000000     26.24000
max        4.000000      1.000000      1.000000      4.000000     41.00000

              atemp      humidity     windspeed        casual    registered  \
count  10886.000000  10886.000000  10886.000000  10886.000000  10886.000000
mean      23.655084     61.886460     12.799395     36.021955    155.552177
std        8.474601     19.245033      8.164537     49.960477    151.039033
```

```
min        0.760000    0.000000    0.000000    0.000000    0.000000
25%       16.665000   47.000000    7.001500    4.000000   36.000000
50%       24.240000   62.000000   12.998000   17.000000  118.000000
75%       31.060000   77.000000   16.997900   49.000000  222.000000
max       45.455000  100.000000   56.996900  367.000000  886.000000

              count
count  10886.000000
mean     191.574132
std      181.144454
min        1.000000
25%       42.000000
50%      145.000000
75%      284.000000
max      977.000000
```

[ ]:
```
data[data.duplicated()].count()
```

[ ]:
```
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
```

Observation: * The dataframe has 1088 rows and 12 columns. * The datatype of the column datetime is object, and the columns tem, atemp, and widspeed have datatypes float. * The remaining columns have datatypes int. * There are no missing or duplicate values in the dataframe.

Numerical & Categorical variables:

[ ]:
```
data.nunique()
```

[ ]:
```
datetime      10886
season            4
holiday           2
workingday        2
weather           4
temp             49
atemp            60
humidity         89
```

```
windspeed       28
casual         309
registered     731
count          822
dtype: int64
```

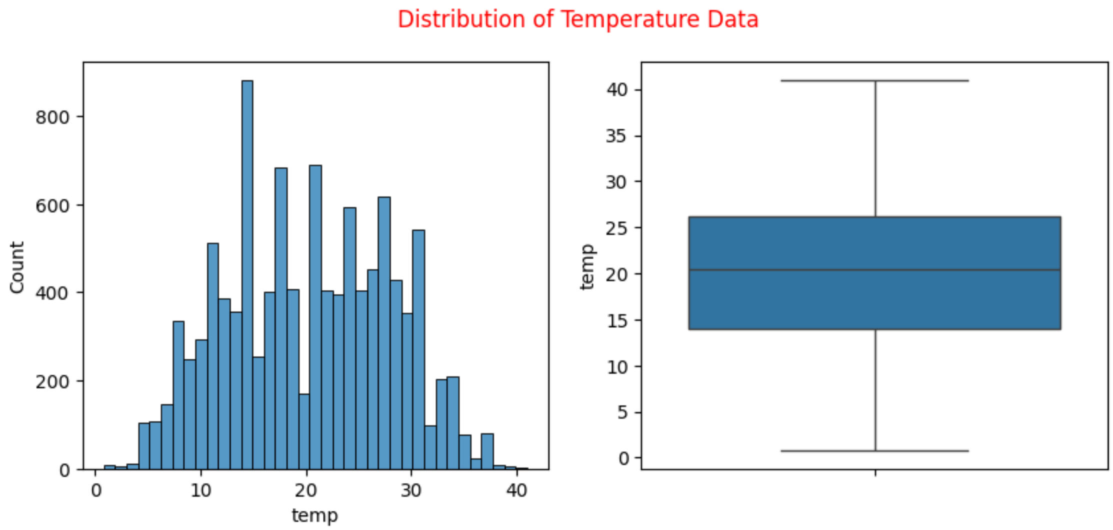
The columns season and weather each have 4 unique values, while the columns holiday and working days each have 2 unique values. These columns will be treated as categorical variables, and the remaining columns will be treated as numerical columns.

1.Numerical Variables:

[ ]:
```
# Temperature:

data["temp"].describe()
```

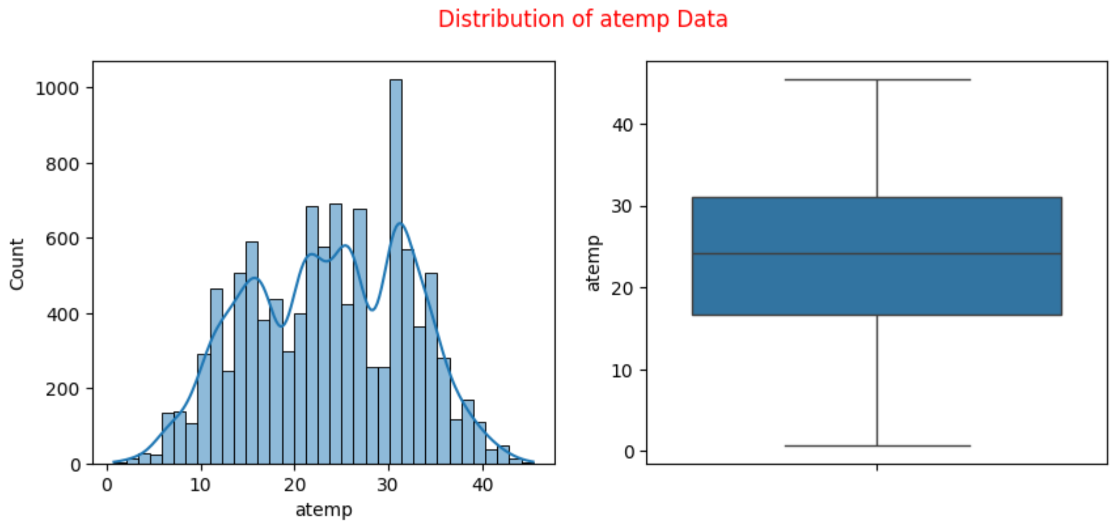
[ ]:
```
count    10886.00000
mean        20.23086
std          7.79159
min          0.82000
25%         13.94000
50%         20.50000
75%         26.24000
max         41.00000
Name: temp, dtype: float64
```

[ ]:
```
plt.figure(figsize=(10,4))

plt.subplot(1,2,1)
sns.histplot(data["temp"])

plt.subplot(1,2,2)
sns.boxplot(data["temp"])

plt.suptitle("Distribution of Temperature Data", color="red")
plt.show()
```



- Temperatue data have approx normal distribution. data does not have outliers.
- The mean temperature is 20 degrees Celsius. The minimum temperature is 0.82 degrees Celsius, and the maximum temperature is 41 degrees Celsius.
- The 25th percentile of the data is 13.94 degrees Celsius.
- The 75th percentile of the data is 26.24 degrees Celsius.

[ ]:
```
# atemp: temperature feel like

plt.figure(figsize=(10,4))
plt.subplot(1,2,1)
sns.histplot(data["atemp"], kde=True)

plt.subplot(1,2,2)
sns.boxplot(data["atemp"])
plt.suptitle("Distribution of atemp Data", color="red")
plt.show()
```



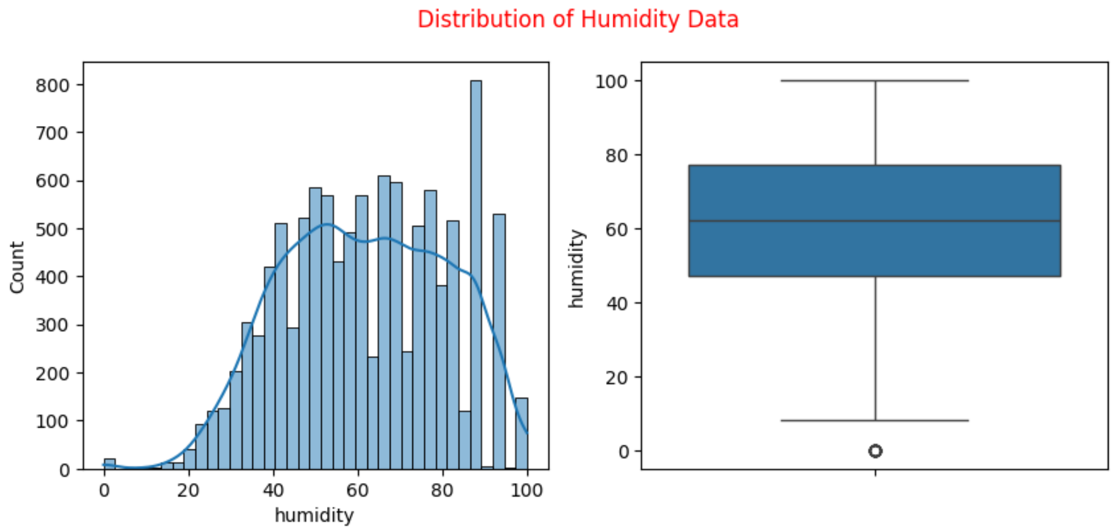
- The plot shows that ‘atemp’ does not follow a normal distribution, and it does not have outliers.
- The mean temperature is 23.65 degrees Celsius.
- The minimum temperature is 0.76 degrees Celsius, and the maximum temperature is 45.45 degrees Celsius.
- The 25th percentile of the data is 16.66 degrees Celsius.
- The 75th percentile of the data is 31 degrees Celsius.

[ ]:

```

```

[ ]:

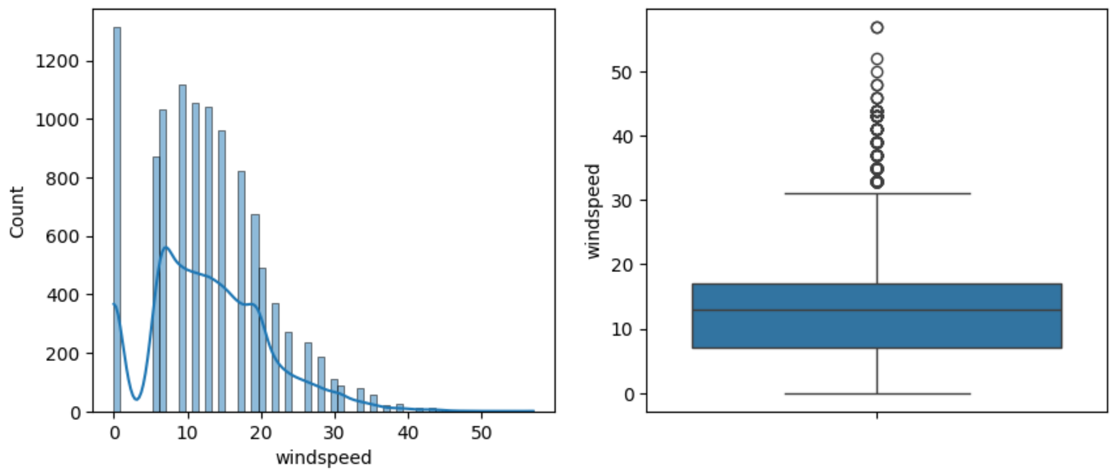
```
# Humidity
plt.figure(figsize=(10,4))
plt.subplot(1,2,1)
sns.histplot(data["humidity"], kde=True)

plt.subplot(1,2,2)
sns.boxplot(data["humidity"])
plt.suptitle("Distribution of Humidity Data", color="red")
plt.show()
```



- We can observe that the data is left-skewed, indicating that it has few values lower than the lower limit.
- The mean value is 61.88.
- The minimum value is 0, and the maximum value is 100.
- The first quartile (Q1) value is 47, and the third quartile (Q3) value is 77.

[ ]:
```
# Windspeed

plt.figure(figsize=(10,4))
plt.subplot(1,2,1)
sns.histplot(data["windspeed"], kde=True)

plt.subplot(1,2,2)
sns.boxplot(data["windspeed"])
```

[ ]:
```
<Axes: ylabel='windspeed'>
```



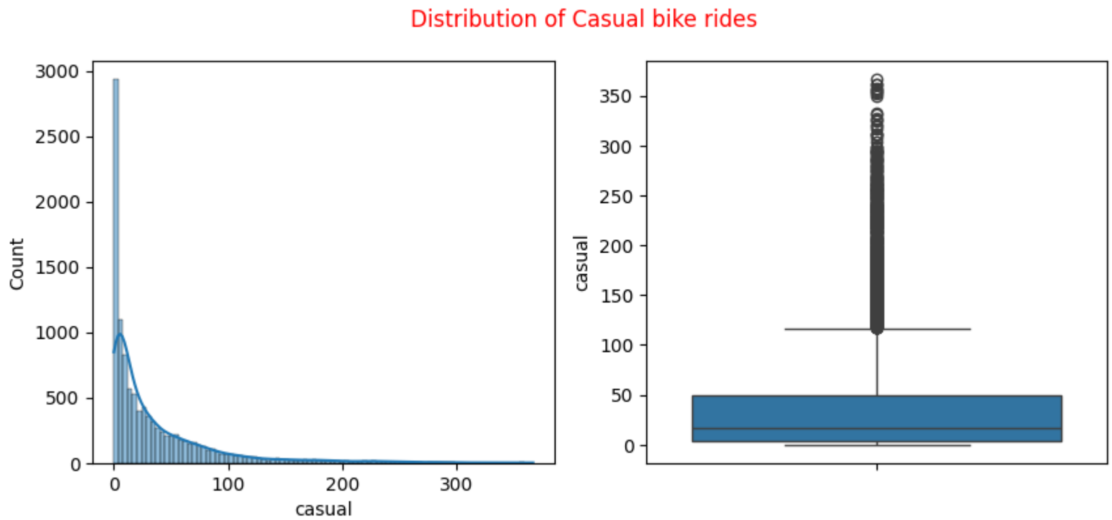

```
[ ]: Q1= data["windspeed"].quantile(0.25)
     Q3= data["windspeed"].quantile(0.75)
     IQR= Q3-Q1
     lower_limit= Q1 - 1.5*IQR
     upper_limit= Q3 + 1.5*IQR
```

```
[ ]: data[data["windspeed"]>upper_limit]["windspeed"].count()
```

```
[ ]: 227
```

- We can observe that the data is right-skewed and contains outliers. Specifically, the column 'windspeed' has 227 outliers.
- The mean value is 12.8.

```
[ ]: # Casual Users
     plt.figure(figsize=(10,4))
     plt.subplot(1,2,1)
     sns.histplot(data["casual"], kde=True)

     plt.subplot(1,2,2)
     sns.boxplot(data["casual"])
     plt.suptitle("Distribution of Casual bike rides", color="red")
     plt.show()
```



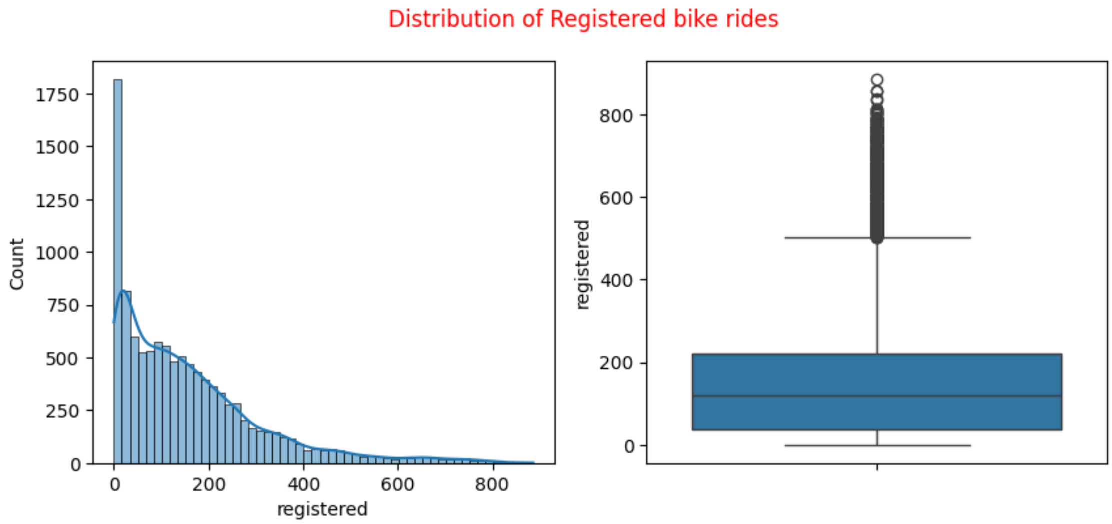

```
[ ]: Q1= data["casual"].quantile(0.25)
     Q3= data["casual"].quantile(0.75)
     IQR= Q3-Q1

     lower_limit= Q1 - 1.5*IQR
     upper_limit= Q3 + 1.5*IQR
```

```
[ ]: data[data["casual"]>upper_limit]["casual"].count()
```

```
[ ]: 749
```

- Upon observing the data, it is evident that the data is right-skewed and contains outliers.
- There are 749 outliers where the values exceed the upper limit.
- The mean value is 36.
- The minimum value is 0, and the maximum value is 376.

```
[ ]: # Registered Users

     plt.figure(figsize=(10,4))
     plt.subplot(1,2,1)
     sns.histplot(data["registered"], kde=True)

     plt.subplot(1,2,2)
     sns.boxplot(data["registered"])
     plt.suptitle("Distribution of Registered bike rides", color="red")
     plt.show()
```




```
[ ]: Q1= data["registered"].quantile(0.25)
     Q3= data["registered"].quantile(0.75)
     IQR= Q3-Q1

     lower_limit= Q1 - 1.5*IQR
     upper_limit= Q3 + 1.5*IQR
```

```
[ ]: data[data["registered"]>upper_limit]["registered"].count()
```

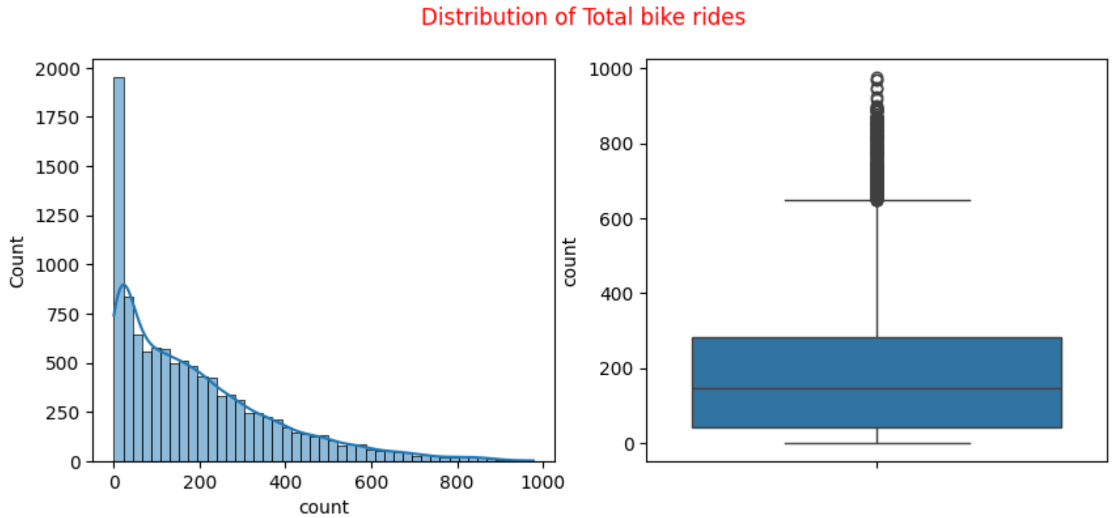
```
[ ]: 423
```

```
[ ]: data["registered"].describe()
```

```
[ ]: count    10886.000000
     mean       155.552177
     std        151.039033
     min          0.000000
     25%         36.000000
     50%        118.000000
     75%        222.000000
     max        886.000000
     Name: registered, dtype: float64
```

- Upon observing the data, it is evident that the data is right-skewed and contains outliers.
- There are 423 outliers where the values exceed the upper limit.
- The mean value is 155.
- The minimum value is 0, and the maximum value is 886.

[ ]:
```
# Total Count
plt.figure(figsize=(10,4))
plt.subplot(1,2,1)
sns.histplot(data["count"], kde=True)

plt.subplot(1,2,2)
sns.boxplot(data["count"])
plt.suptitle("Distribution of Total bike rides", color="red")
plt.show()
```



[ ]:
```
Q1= data["count"].quantile(0.25)
Q3= data["count"].quantile(0.75)
IQR= Q3-Q1

lower_limit= Q1 - 1.5*IQR
upper_limit= Q3 + 1.5*IQR
```

[ ]:
```
data[data["count"]>upper_limit]["count"].count()
```

[ ]: 300

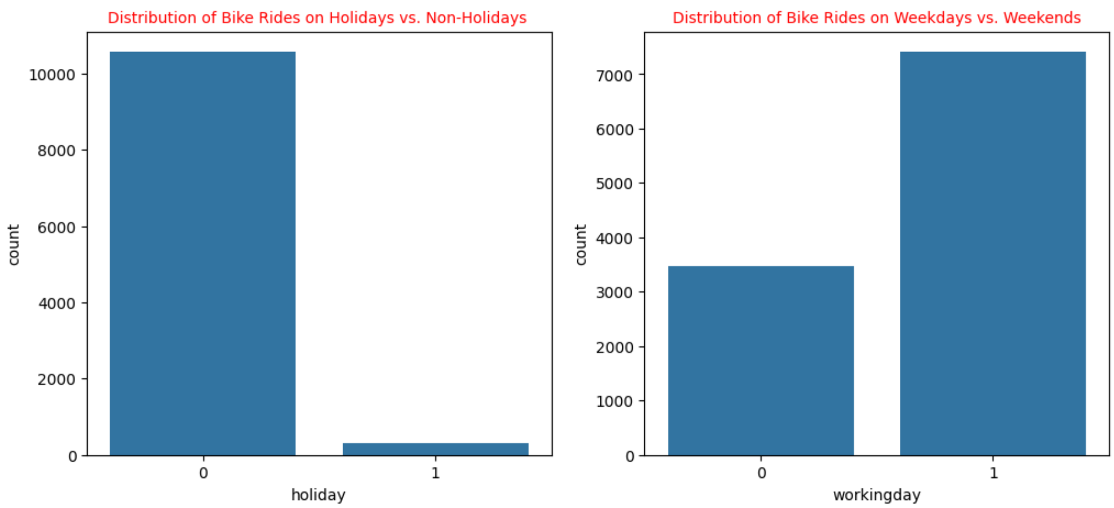
- Upon observing the data, it is evident that the data is right-skewed and contains outliers.
- There are 300 outliers where the values exceed the upper limit.
- The mean value is 191.
- The minimum value is 1, and the maximum value is 977.

2. Categorical variables:

[ ]:
```
labels= data["season"].value_counts().index
labels
```

[ ]: Index([4, 2, 3, 1], dtype='int64', name='season')

[ ]:
```
# Holiday
plt.figure(figsize=(12,5))

plt.subplot(1,2,1)
sns.countplot(data=data, x="holiday")
plt.title("Distribution of Bike Rides on Holidays vs. Non-Holidays ",␣
↪color="red", fontsize=10)

plt.subplot(1,2,2)
sns.countplot(data=data, x="workingday")
plt.title("Distribution of Bike Rides on Weekdays vs. Weekends", color="red",␣
↪fontsize=10)

plt.show()
```



[ ]:
```
# percentage of bike rides
rides_h=(data.groupby("holiday")["count"].sum())/data["count"].sum()*100
rides_h
```

[ ]:
```
holiday
0    97.228067
1     2.771933
Name: count, dtype: float64
```

[ ]:
```
# percentage of bike rides
rides_w= data.groupby("workingday")["count"].sum()/data["count"].sum()*100
rides_w
```

[ ]: workingday
0   31.40156
1   68.59844
Name: count, dtype: float64

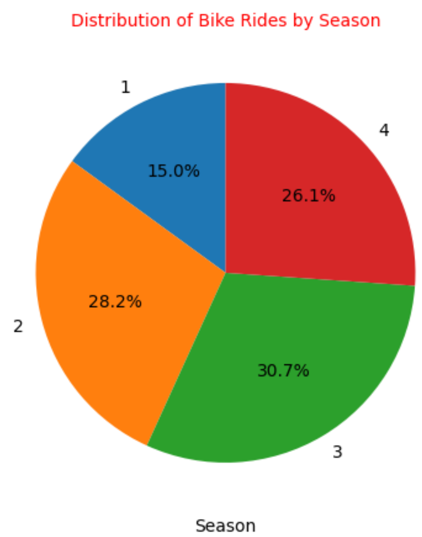
Here, we observe the distribution of bike rides during holidays and working days.

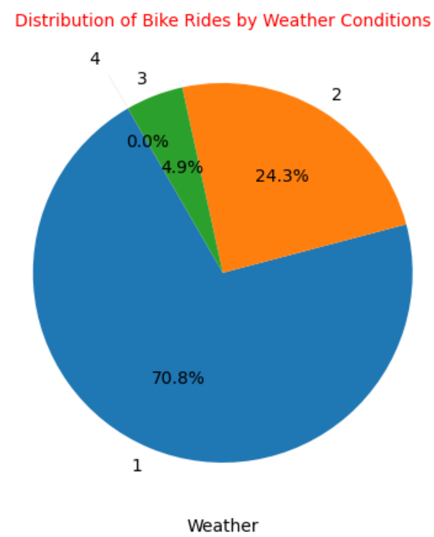
- The majority of bike rides, 97.22% of the total, are observed on non-holidays, whereas only 2.77% of bike rides occur on holidays.
- The majority of bike rides, accounting for 68.6% of the total, are observed on working days. Meanwhile, 31% of bike rides are observed on holidays.

[ ]:
```
season=data.groupby("season")["count"].sum()
size=np.array(season.values)
labels=np.array(season.index)
```

[ ]:
```
weather=data.groupby("weather")["count"].sum()
size_w=np.array(weather.values)
labels_w=np.array(weather.index)
```

[ ]:
```
# pie chart of season
plt.figure(figsize=(12,5))
plt.subplot(1,2,1)
plt.pie(season.values, labels=season.index,  autopct='%1.1f%%', startangle=90, ␣
  ↪explode=(0,0,0,0))
plt.xlabel("Season")
plt.title("Distribution of Bike Rides by Season", color="red", fontsize=10)

#pie chart of weather
plt.subplot(1,2,2)
plt.pie(size_w, labels=labels_w,  autopct='%1.1f%%', startangle=120, ␣
  ↪explode=(0,0,0,0.2))
plt.title("Distribution of Bike Rides by Weather Conditions", color="red",␣
  ↪fontsize=10)
plt.xlabel("Weather")
plt.show()
```





Season: * From the pie chart, we observe that: 1. Season 3 has the highest number of bike riders, accounting for 30.7% of the total bike rides. 2. Season 2 follows closely with 28.2% of the total bike rides. 3. Season 4 and Season 1 account for 26.1% and 15% of the total bike rides, respectively.

weather: 1. Weather type 1 has the majority of bike rides, accounting for 70.8% of the total, followed by weather types 2 and 3, which make up 24.3% and 4.9% of the total bike rides.

2. Weather type 4 has 0% of bike rides

Outlier detation and deletion:

Converting categorical variables from datatype object to the appropriate categorical type.

[ ]:
```
data["season"]=data["season"].astype("object")
data["holiday"]=data["holiday"].astype("object")
data["workingday"]=data["workingday"].astype("object")
data["weather"]=data["weather"].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  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
```

[ ]:
```
df_num=data.select_dtypes(include= np.number)
```

[ ]:
```
df_num.columns
```

[ ]:
```
Index(['temp', 'atemp', 'humidity', 'windspeed', 'casual', 'registered',
       'count'],
      dtype='object')
```

[ ]:
```
df_cat=data.select_dtypes(include="object")
df_cat.columns
```

[ ]:
```
Index(['datetime', 'season', 'holiday', 'workingday', 'weather'],
dtype='object')
```

[ ]:
```
df_cat.shape
```

[ ]:
```
(10886, 5)
```

Outlier treatment:

[ ]:
```
Q1=df_num.quantile(0.25)
```

[ ]:
```
Q3=df_num.quantile(0.75)
```

[ ]:
```
IQR = Q3-Q1
```

[ ]:
```
IQR
```

[ ]:
```
temp           12.3000
atemp          14.3950
humidity       30.0000
windspeed       9.9964
casual         45.0000
registered    186.0000
count         242.0000
dtype: float64
```

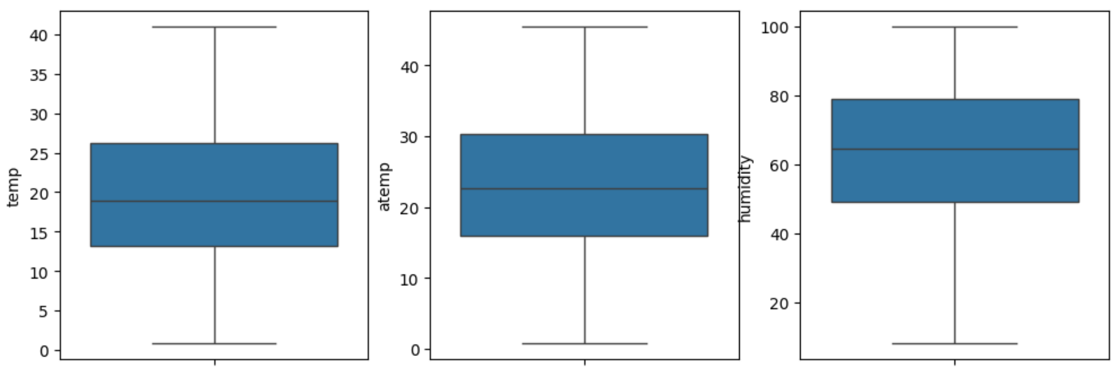
[ ]:
```
# Outlier filter:
df_iqr= data[~((df_num<(Q1 - 1.5 * IQR)) | (df_num > ( Q3 + 1.5 * IQR))).
 ↪any(axis=1)]
```

[ ]:
```
df_iqr.shape
```

[ ]: (9518, 12)

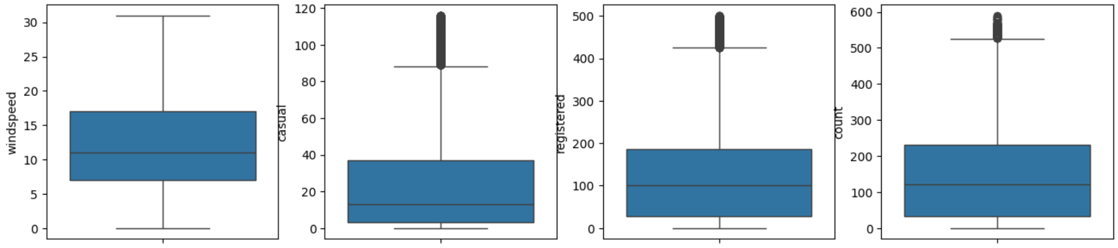
We can observe that initially, we had 1088 rows in the dataframe. After filtering out outliers, we now have 9518 rows.

[ ]:
```
plt.figure(figsize=(12,4))

plt.subplot(1,3,1)
sns.boxplot((df_iqr["temp"]))

plt.subplot(1,3,2)
sns.boxplot(df_iqr["atemp"])

plt.subplot(1,3,3)
sns.boxplot(df_iqr["humidity"])

plt.show()
```



[ ]:
```
plt.figure(figsize=(16,12))

plt.subplot(3,4,1)
sns.boxplot(df_iqr["windspeed"])

plt.subplot(3,4,2)
sns.boxplot(df_iqr["casual"])

plt.subplot(3,4,3)
sns.boxplot(df_iqr["registered"])

plt.subplot(3,4,4)
sns.boxplot(df_iqr["count"])
```

```
plt.show()
```



2 2. Try establishing a Relationship between the Dependent and Independent Variables.

1. Numerical vs numerical variables:

[ ]:
```
df_iqr[["temp","atemp","humidity","windspeed","casual","registered","count"]].
↪corr()
```

[ ]:
```
                temp     atemp  humidity  windspeed    casual  registered  \
temp        1.000000  0.986415 -0.012123  -0.017049  0.515266    0.276560
atemp       0.986415  1.000000  0.006011  -0.058533  0.508335    0.273281
humidity   -0.012123  0.006011  1.000000  -0.302630 -0.335487   -0.262876
windspeed  -0.017049 -0.058533 -0.302630   1.000000  0.112429    0.109959
casual      0.515266  0.508335 -0.335487   0.112429  1.000000    0.579577
registered  0.276560  0.273281 -0.262876   0.109959  0.579577    1.000000
count       0.345398  0.341134 -0.296702   0.118392  0.707486    0.985967

               count
temp        0.345398
atemp       0.341134
humidity   -0.296702
windspeed   0.118392
casual      0.707486
registered  0.985967
count       1.000000
```

Correlation between temp and count

- H0: No correlation
- Ha: There is correlation

[ ]:
```
from scipy.stats import spearmanr
spearmanr(df_iqr["temp"],df_iqr["casual"])
```

[ ]:
```
SignificanceResult(statistic=0.5292099833466296, pvalue=0.0)
```

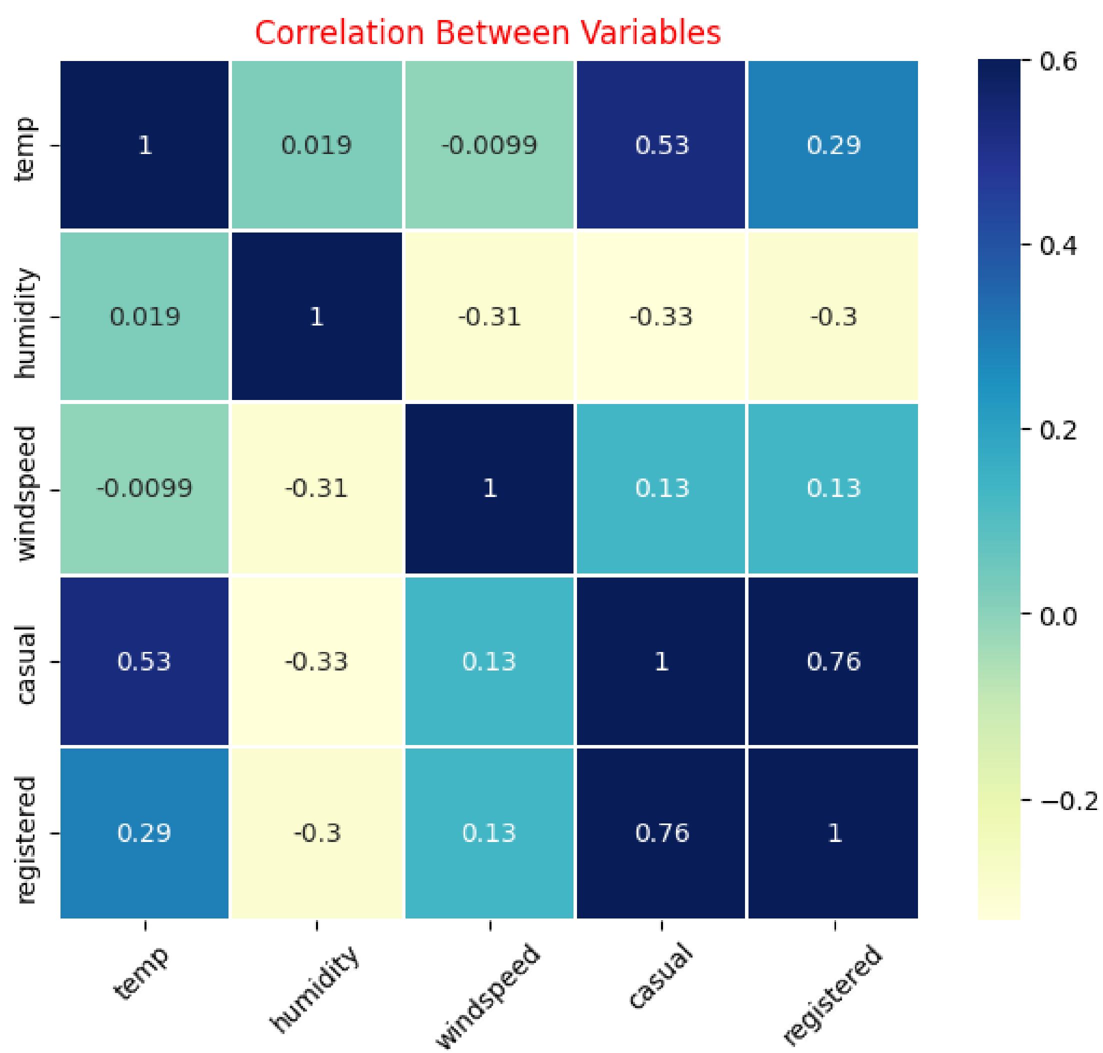
[ ]:
```
spearmanr(df_iqr["temp"],df_iqr["registered"])
```

[ ]:
```
SignificanceResult(statistic=0.29000994215471365, pvalue=8.125458220956345e-184)
```

[ ]:
```
spearmanr(df_iqr["temp"],df_iqr["count"])
```

[ ]:
```
SignificanceResult(statistic=0.34305109699360686,
pvalue=4.5352449165014825e-261)
```

[ ]:
```
plt.figure(figsize=(8,6))

sns.heatmap(df_iqr[["temp","humidity","windspeed","casual","registered"]].
↪corr(method = 'spearman'), annot=True, vmax = .6, linewidths=0.01,␣
↪square=True, cmap="YlGnBu")
plt.title("Correlation Between Variables", color="red", fontsize=12)
plt.xticks(rotation=45)
plt.show()
```



Observation:

The correlations between temperature and casual, registered, and all riders are as follows: 1. The correlation between temperature and casual riders is 0.53. 2. The correlation between temperature and registered riders is 0.29. It can be observed that temperature and casual riders have a moderately strong positive correlation.

Wind speed has the same correlation with both casual and registered rides, which is 0.13. This indicates that wind speed and bike rides have a weak or no relationship.

The correlations between humidity and casual, registered, and all bike riders are as follows: 1. The correlation between humidity and casual riders is -0.33. 2. The correlation between humidity and registered riders is -0.30. It can be observed that humidity has a negative correlation with both casual and registered bike riders.

[ ]:
```
df_iqr.sample()
```

[ ]:
```
                  datetime  season  holiday  workingday  weather   temp   atemp  \
10722  2012-12-13 04:00:00       4        0           1        2  10.66  12.88

       humidity  windspeed  casual  registered  count
10722        56     12.998       0           8      8
```

3 3. Check if there any significant difference between the no. of bike rides on Weekdays and Weekends?

[ ]:
```
df_iqr["workingday"].value_counts()
```

[ ]:
```
workingday
1    6790
0    2728
Name: count, dtype: int64
```

[ ]:
```
df_iqr.groupby("workingday")["count"].describe()
```

[ ]:
```
             count        mean         std  min   25%    50%    75%    max
workingday
0           2728.0  120.681085  106.747811  1.0  31.0   89.0  188.0  551.0
1           6790.0  161.970103  138.588572  1.0  35.0  138.0  249.0  590.0
```

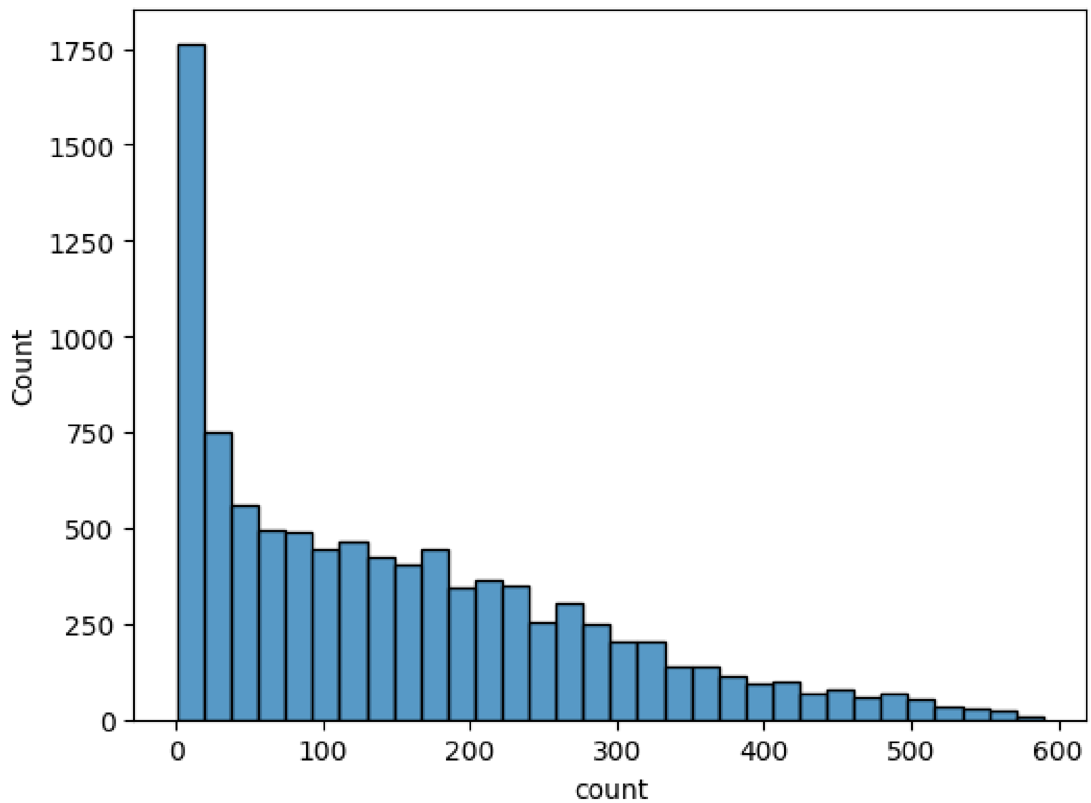
[ ]:
```
df_workingday=df_iqr[["workingday","count"]]
df_workingday.sample().info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1 entries, 4699 to 4699
Data columns (total 2 columns):
```

```
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   workingday  1 non-null      object
 1   count       1 non-null      int64
dtypes: int64(1), object(1)
memory usage: 24.0+ bytes
```

[ ]:
```
sns.histplot(df_workingday["count"])
```

[ ]: <Axes: xlabel='count', ylabel='Count'>



Transforming right skewed data to normali ditributed data using boxcox.

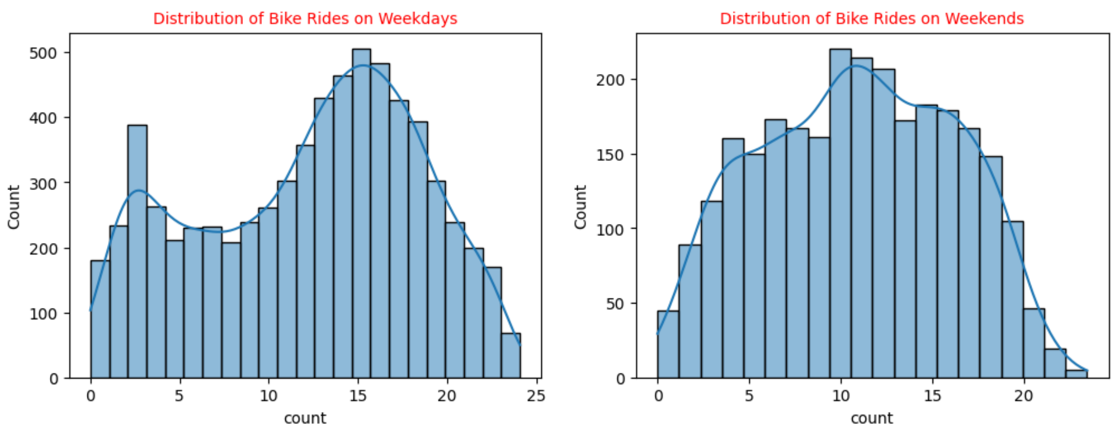
[ ]:
```
from scipy.stats import boxcox
df_workingday["count"], best_lambda= boxcox(df_workingday["count"])
```

```
<ipython-input-58-27dec5c0fe39>:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
```

```
df_workingday["count"], best_lambda= boxcox(df_workingday["count"])
```

[ ]:
```
df_weekday=df_workingday[df_workingday["workingday"]==1]["count"]
df_weekend=df_workingday[df_workingday["workingday"]==0]["count"]
```

[ ]:
```
plt.figure(figsize=(12,4))

plt.subplot(1,2,1)
sns.histplot(df_weekday, kde=True)
plt.title("Distribution of Bike Rides on Weekdays", color="red", fontsize=10)

plt.subplot(1,2,2)
sns.histplot(df_weekend, kde=True)
plt.title("Distribution of Bike Rides on Weekends", color="red", fontsize=10)

plt.show()
```



Null Hypothesis (H0) and Alternate Hypothesis (H1) * H0: There is no significant difference between average bike rides on weekdays and weekends. * H1: There is a significant difference between average bike rides on weekdays and weekends.

Here we can observe that the data approximately follows a Gaussian (normal) distribution, and the dataset is large. Therefore, we can apply a two-sample t-test in this case.

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

t_stat, p_val= ttest_ind(df_weekday, df_weekend, alternative="two-sided")
t_stat, p_val
```

[ ]:
```
(10.00135368319239, 1.9618398183131075e-23)
```

- Significance level (alpha): 5% (0.05)
- Test statistic: 10.00135368319239

- p-value: 1.9618398183131075e-23 The p-value is close to zero and less than the significance level (alpha), therefore we reject the null hypothesis (H0).

Conclusion: There is a significant difference in the average bike rides between weekdays and weekends.

- H0: There is no significant difference between average bike rides on weekdays and weekends.
- H1: Average bike rides on weekdays are greater than average bike rides on weekends.

[ ]:
```
t_stat, p_val= ttest_ind(df_weekday, df_weekend, alternative="greater")
t_stat, p_val
```

[ ]:
```
(10.00135368319239, 9.809199091565538e-24)
```

- Significance level (alpha): 5% (0.05)
- Test statistic: 10.00135368319239
- P-value: 9.809199091565538e-24 The p-value is close to zero and less than the significance level (alpha), we can reject the null hypothesis (H0).

Conclusion: The average bike rides on weekdays are greater than the average bike rides on weekends.

4 4. Check if the demand of bicycles on rent is the same for different Weather conditions?

[ ]:
```
df_iqr.groupby("weather")["count"].describe()
```

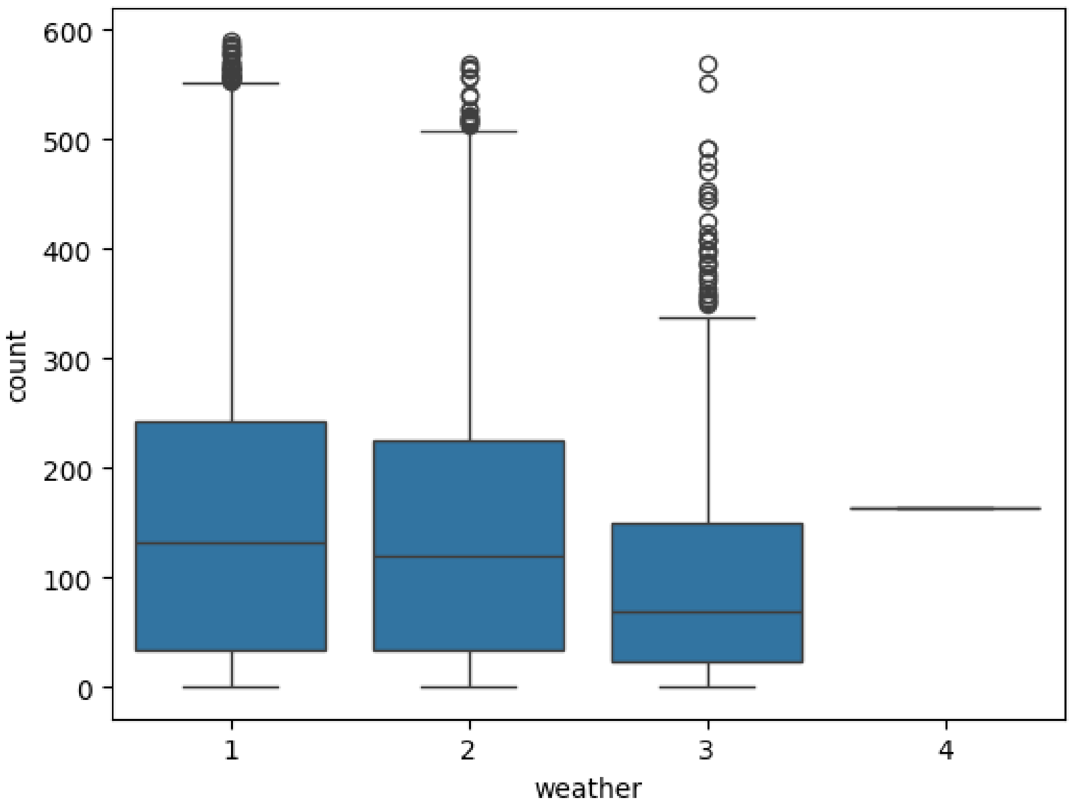
[ ]:

weather	count	mean	std	min	25%	50%	75%	max
1	6176.0	157.522021	135.224946	1.0	35.0	132.0	242.0	590.0
2	2568.0	146.805685	127.190755	1.0	35.0	120.0	226.0	568.0
3	773.0	102.170763	103.066276	1.0	23.0	70.0	149.0	569.0
4	1.0	164.000000	NaN	164.0	164.0	164.0	164.0	164.0

weather: 1. Clear, Few clouds, 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 Pellets + Thunderstorm + Mist, Snow + Fog

[ ]:
```
sns.boxplot(x="weather", y="count", data=df_iqr)
```

[ ]:
```
<Axes: xlabel='weather', ylabel='count'>
```



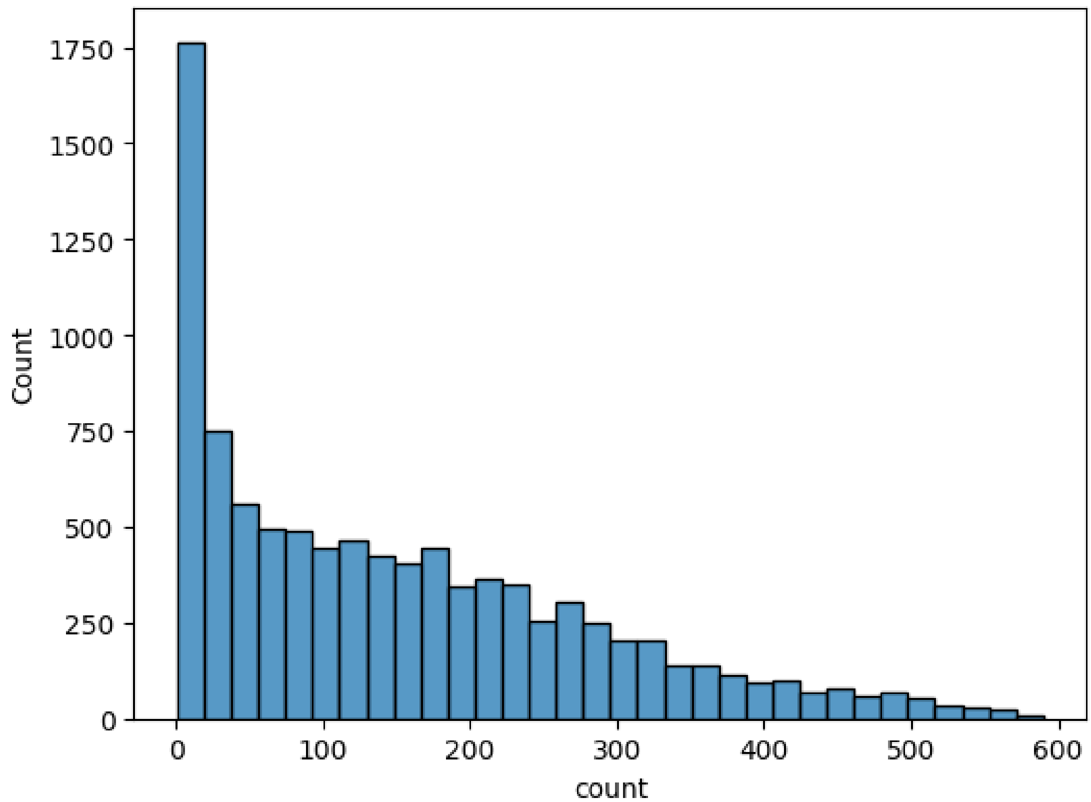
H0: All weather conditions have the same mean of bike rides.

H1: At least one of the weather conditions has a different mean of bike rides.

[ ]:
```
df_weather=df_iqr[["weather","count"]]
```

[ ]:
```
sns.histplot(df_weather["count"])
```

[ ]: <Axes: xlabel='count', ylabel='Count'>



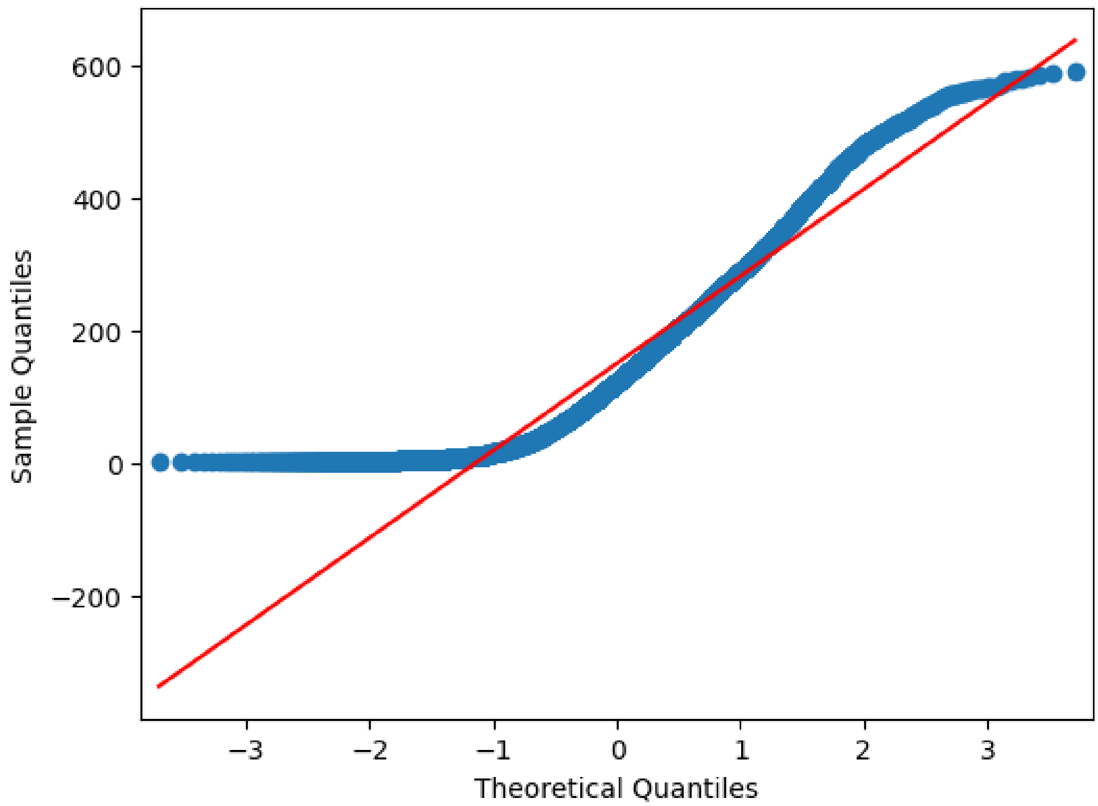
Check assumptions of the test.

```
[ ]: weather_1=df_weather[df_weather["weather"]==1]["count"]
weather_2=df_weather[df_weather["weather"]==2]["count"]
weather_3=df_weather[df_weather["weather"]==3]["count"]
weather_4=df_weather[df_weather["weather"]==4]["count"]
```

i. **Normality Test**

- Using QQ Plot to Check Gaussian Distribution: The QQ plot (Quantile-Quantile plot) is used to assess whether the data follows a Gaussian distribution.
- Inability to Use Shapiro-Wilk Test Due to Large Data Size: The Shapiro-Wilk test, typically recommended for sample sizes of 50-200, cannot be used effectively for large datasets.

```
[ ]: # QQPlot
import statsmodels.api as sm
import matplotlib.pyplot as plt

sm.qqplot(df_weather["count"], line="s")
plt.show()
```



[ ]:
```
df_weather["count"].skew()
```

[ ]: 0.8923415130851888

[ ]:
```
df_weather["count"].kurt()
```

[ ]: 0.11828183897731259

Observation: * We can see that the variable count does not follow a normal distribution. Therefore, to compare the medians of independent groups (category weather), will use *the* Kruskal-Wallis test. * We can see that the variable 'count' has right skewness (0.8923415130851888). * Positive kurtosis (0.11828183897731259) suggests a distribution with heavy tails, indicating the presence of outliers.

ii. Equality Variance: levene Test * H0: Variances are equal accross the groups. * H1: Variances significantly different across the groups.

[ ]:
```
from scipy.stats import levene
levene_stat, p_val=levene(weather_1, weather_2, weather_3, weather_4)
levene_stat, p_val
```

[ ]: (43.7279017358368, 4.67345079664337e-28)

- alpha= 0.05
- pval= 1.1274167026429413e-26 **Conclusion:**
- p-value (1.1274167026429413e-26) is very small and less than the significance level (alpha = 0.05), we reject the null hypothesis (Ho). Therefore,The variances are significantly different across the groups.

Null Hypothesis (H0) and Alternate Hypothesis (H1) * H0: There is no statistically significant difference in the medians of the groups. * H1: There is a statistically significant difference in the medians of at least two groups.

[ ]:
```
from scipy.stats import kruskal
stat, p_value= kruskal(weather_1, weather_2, weather_3, weather_4)
stat, p_value
```

[ ]:
```
(115.95354025348767, 5.738374025114387e-25)
```

Obsevation: * alpha=5% (0.05) * p value: 5.738374025114387e-25 * Conclusion: Since the p-value (5.738374025114387e-25) is very small and less than alpha, we reject the null hypothesis (Ho). we can conclude that there is a statistically significant difference in the medians of at least two groups.

5 5. Check if the demand of bicycles on rent is the same for different Seasons?

Seasons: * 1: spring, 2: summer, 3: fall, 4: winter

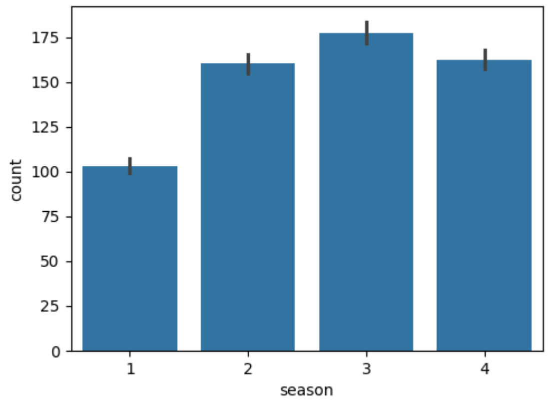
[ ]:
```
df_iqr.groupby("season")["count"].describe()
```

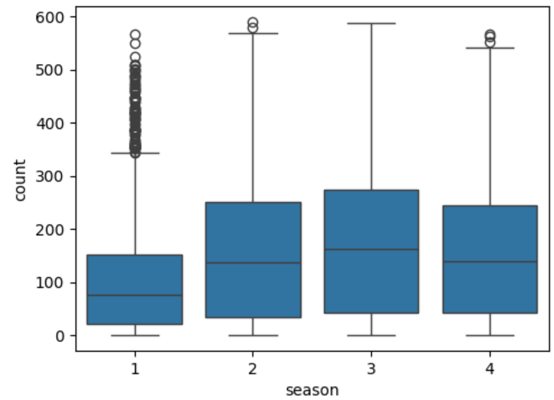
[ ]:

	count	mean	std	min	25%	50%	75%	max
season								
1	2463.0	103.164028	101.351256	1.0	22.5	76.0	151.00	566.0
2	2292.0	160.360820	136.243867	1.0	34.0	136.5	250.25	590.0
3	2288.0	177.151661	141.381497	1.0	43.0	162.0	274.00	587.0
4	2475.0	162.437172	132.658631	1.0	42.0	140.0	245.00	566.0

[ ]:
```
plt.figure(figsize=(12,4))

plt.subplot(1,2,1)
sns.barplot(x="season", y="count", data=df_iqr)

plt.subplot(1,2,2)
sns.boxplot(x="season", y="count", data=df_iqr)
```

[ ]:
```
<Axes: xlabel='season', ylabel='count'>
```

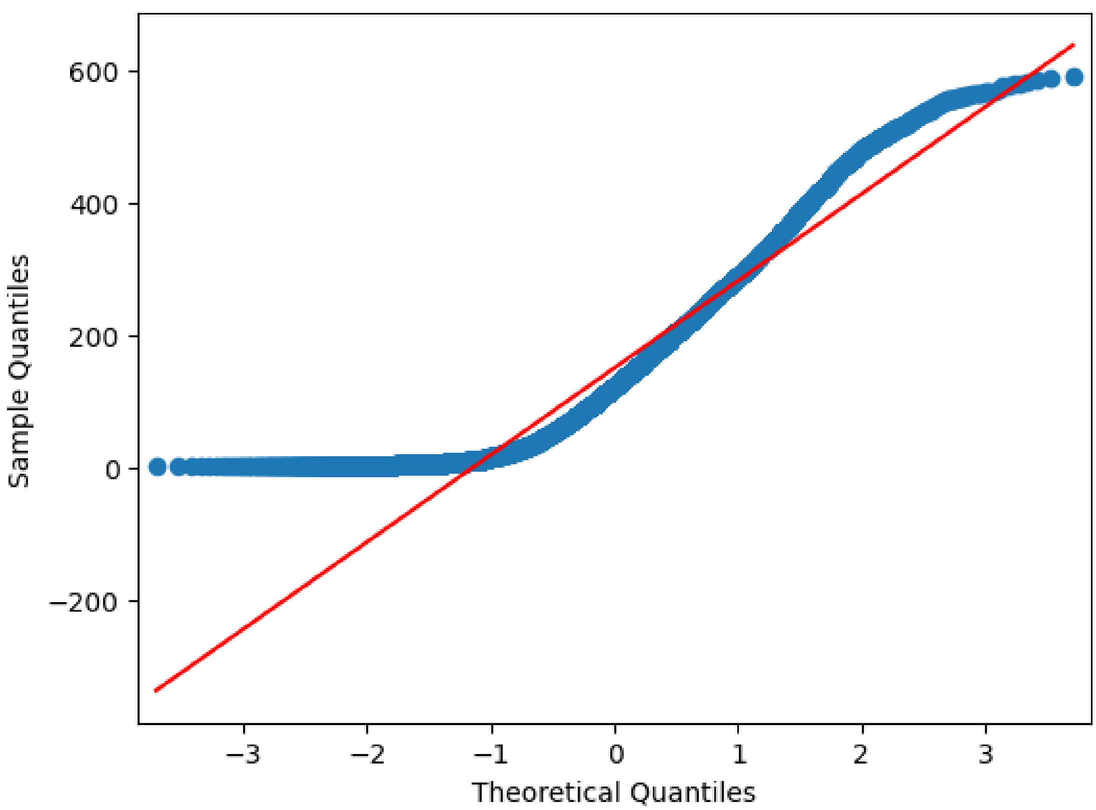




Null Hypothesis and Alternate Hypothesis: * H0: The mean of bike rentals is the same across all seasons. * H1: The mean of bike rentals varies across different seasons.

[ ]:
```
from statsmodels.graphics.gofplots import qqplot
plt.figure(figsize=(8,4))
qqplot(df_iqr["count"], line="s")
plt.show()
```

<Figure size 800x400 with 0 Axes>



[ ]:
```
df_iqr["count"].skew()
```

[ ]: 0.8923415130851888

[ ]:
```
df_iqr["count"].kurt()
```

[ ]: 0.11828183897731259

Observtion: * We can observe from the qqplot that our data does not follow a Gaussian distribution. * We can see that the variable 'count' has right skewness (0.8923415130851888). * Positive kurtosis (0.11828183897731259) suggests a distribution with heavy tails, indicating the presence of outliers.

ii. Equality Variance

[ ]:
```
season_1=df_iqr[df_iqr["season"]==1]
season_2=df_iqr[df_iqr["season"]==2]
season_3=df_iqr[df_iqr["season"]==3]
season_4=df_iqr[df_iqr["season"]==4]
```

levene Test * H0: Variances are equal accross the groups. * H1: Variances are significantly different across the groups.

[ ]:
```
levene_stat, p_val= levene(season_1["count"], season_2["count"],␣
↪season_3["count"], season_4["count"])
levene_stat, p_val
```

[ ]: (137.00312941554566, 6.687186315723853e-87)

- Alpha = 0.05
- P-value = 6.687186315723853e-87
- Since the p-value is very small (much less than alpha), we reject the null hypothesis (Ho). **Conclusion:** The variances significantly differ across the groups.

we obsereved that the data does not follow a normal distribution and the sample size is large, we will use the Kruskal-Wallis test.

[ ]:
```
stats, p_val= kruskal(season_1["count"],␣
↪season_2["count"],season_3["count"],season_4["count"])
stats, p_val
```

[ ]: (429.4814657501883, 9.092946705054743e-93)

Observation: * Alpha = 0.05 * P-value = 9.092946705054743e-93 * Since the p-value is much less than alpha, we reject the null hypothesis (H0). * *Conclusion:* At least one group has a different median. Therefore, the medians of bike rentals differ across all seasons.

6 6. Check if the Weather conditions are significantly different during different Seasons?

Seasons: 1: spring, 2: summer, 3: fall, 4:winter.

Weather: 1. Clear, Few clouds, 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 Pellets + Thunderstorm + Mist, Snow + Fog

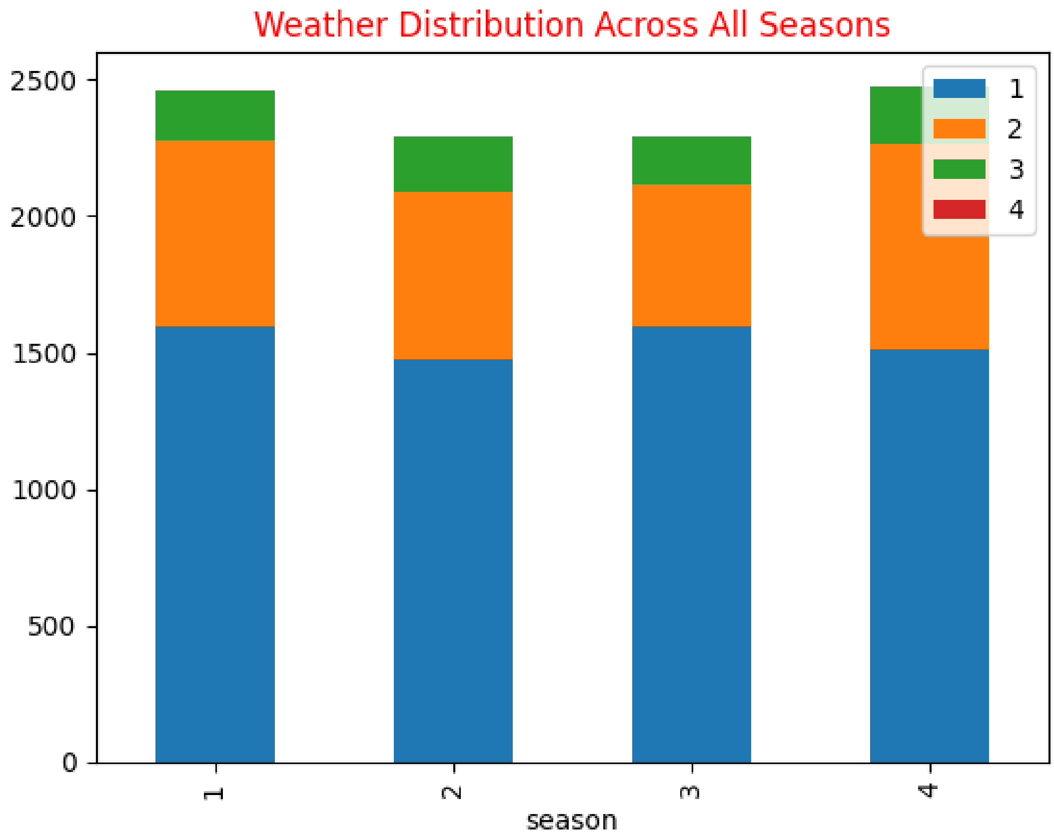
[ ]:
```
# Cross table for weather vs season
cross_tab=pd.crosstab(df_iqr["season"],df_iqr["weather"], margins=True)
# Normalized cross_tab
cross_tab_norm=cross_tab/cross_tab.loc["All", "All"]
cross_tab_norm*100
```

[ ]:

weather	1	2	3	4	All
season					
1	16.757722	7.175877	1.933179	0.010506	25.877285
2	15.475940	6.450935	2.153814	0.000000	24.080689
3	16.789241	5.431813	1.817609	0.000000	24.038664
4	15.864677	7.921832	2.216852	0.000000	26.003362
All	64.887581	26.980458	8.121454	0.010506	100.000000

[ ]:
```
plt.figure(figsize=(8,4))

pd.crosstab(df_iqr["season"],df_iqr["weather"]).plot(kind="bar", stacked=True)
plt.title("Weather Distribution Across All Seasons", color="red", fontsize=12)
plt.legend(loc="upper right")
plt.show()
```

```
<Figure size 800x400 with 0 Axes>
```



Null Hypothesis and Alternate Hypothesis: * H0: Season does not impact the weather. * H1: Season impact the weather.

Distribution: Chi-square distribution.

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

[ ]:
```
chi_stat, p_value, df, exp_freq = chi2_contingency(pd.
    ↪crosstab(df_iqr["season"],df_iqr["weather"]))
print("Chi_stats:",chi_stat)
print("P Value:", p_value)
print("Df:", df)
print("Exp_freq:",exp_freq)
```

```
Chi_stats: 49.95660707768859
P Value: 1.0976664201931232e-07
Df: 9
Exp_freq: [[1.59818113e+03 6.64528682e+02 2.00031414e+02 2.58772851e-01]
 [1.48722337e+03 6.18392099e+02 1.86143728e+02 2.40806892e-01]
 [1.48462786e+03 6.17312881e+02 1.85818870e+02 2.40386636e-01]
 [1.60596764e+03 6.77766337e+02 2.01005989e+02 2.60033621e-01]]
```

- significance level: 5% (0.05)
- P value: 1.0976664201931232e-07
- *Conclusion:* Since the p-value (1.0976664201931232e-07) is less than the significance level (alpha = 0.05), we reject the null hypothesis (H0). we can conclude that season does impact weather.

Observation: We observe from the plot and cross-table that weather 1 is the dominant weather pattern across all seasons, followed by weather 2 and weather 3. * In Season 1 (Spring), weather 1 (mostly clear) accounted for 16.8%, followed by 7.2% for weather 2 (partially cloudy), and 2% for weather 3 (light snow, light rain, and thunderstorm). Only Season 1 had weather 4 (heavy rain + ice pellets + thunderstorm + mist, snow + fog) with a very low percentage of 0.01%.

- Season 2 had 15.5% for weather 1 (mostly clear), followed by 6.5% for weather 2 (partially cloudy), and 2.2% for weather 3 (light snow, light rain, and thunderstorm).
- Season 3 had 16.8% for weather 1 (mostly clear), followed by 5.4% for weather 2 (partially cloudy), and 1.8% for weather 3 (light snow, light rain, and thunderstorm).
- Season 4 had 15.9% for weather 1 (mostly clear), followed by 8% for weather 2 (partially cloudy), and 2.2% for weather 3 (light snow, light rain, and thunderstorm).

7 Insights:

Holiday vs. Working Days: * The majority of bike rides, 97.22%, occur on No Holidays, indicating that weekdays are more popular for bike usage. * Holidays account for only 2.77% of the total bike rides, very less bike usage on these days.

Seasonal: * Season 3 sees the highest bike rides at 30.7%, followed closely by Season 2 with 28.2%. * Season 4 and Season 1 have slightly lower percentages at 26.1% and 15%

Weather Conditions: * Weather type 1 (Mostly Clear) dominates bike rides, constituting 70.8% of the total. * Weather types 2 (Partly Cloudy) and 3 (Light Snow, Light Rain, and Thunderstorm) follow with 24.3% and 4.9%, respectively. * As expected weather type 4 ('Heavy Rain + Ice Pellets + Thunderstorm + Mist, Snow + Fog') has 0% of bike rides

Temperature and Bike Riders: * Casual riders show a stronger positive correlation (0.53) with temperature compared to registered riders (0.29). * This suggests that casual riders, who are likely more influenced by weather conditions, tend to increase their usage as temperatures rise (winter to summer).

Wind Speed and Humidity: * Both casual and registered riders have a weak positive correlation (0.13) with wind speed, indicating minimal or no impact on bike usage. * Humidity negatively correlates with both casual (-0.33) and registered (-0.30) riders, suggesting that higher humidity levels decreases bike usage.

8 Recommendations:

- As we know, workdays and non-workdays are the most influential factors in bike rides. We have observed that working days and non-holidays account for most of the bike rides. This time is the best for business planning. Accordingly, we can plan our inventory, supply chain, and quick customer services for a better customer experience. We can also adjust our prices for higher profitability during peak times. This also a best time to introduce new transportation-related services.

- Holidays and weekends see lower bike ride volumes. Advertising and offering discounts on rides can help improve business during these periods.
- The second most influential factor is weather; we have observed that 71% of bike rides occur on clear days, followed by partially cloudy and light rain days (24%). Weather forecasts are crucial for inventory planning and pricing strategies. We can adjust our prices based on weather conditions, offering minimal surcharges on clear days and discounts on partially cloudy or rainy days.
- We have also identified a strong correlation between temperature increases and casual riders. This suggests that clear seasons are the optimal time to convert casual bike riders into registered riders.
- Both casual (-0.33) and registered (-0.30) riders show a negative correlation with humidity, indicating that higher humidity levels decrease bike usage. offering discounts or promotions during periods of high humidity to incentivize ridership.
- Bike rides are highly influenced by both workdays and holidays, as well as weather conditions. Therefore, monitoring weather conditions and planning business strategy are crucial. Since ridership increases during clear and pleasant days, providing weather information through appropriate advertisements and offering additional discounts via the service app during low business days can further improve business.

[ ]:

```

```