Yulu Business Case Study

Submitted by :

B. Raju Naik

1. About Yulu:

Yulu, India's pioneering micro-mobility service provider, has embarked on a mission to revolutionize daily commutes by offering unique, sustainable transportation solutions.

2. Business Problem:

The business problem for Yulu is twofold: understanding the specific factors influencing the demand for shared electric cycles in the Indian market to facilitate strategic expansion, and addressing recent revenue setbacks by making informed adjustments to regain profitability.

3. About Dataset:

Column Profiling:

- **datetime:** datetime
- **season:** season (1: spring, 2: summer, 3: fall, 4: winter)
- **holiday :** whether day is a holiday or not
- **workingday :** if day is neither weekend nor holiday is 1, otherwise is 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
- **temp:** temperature in Celsius
- **atemp:** feeling temperature in Celsius
- **humidity:** humidity
- **windspeed:** wind speed
- **casual:** count of casual users
- **registered:** count of registered users
- **count:** count of total rental bikes including both casual and registered

1. Exploratory Data Analysis

Firstly Let's import the required libraries

In [3]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [4]:
```
# Let's read the data

data = pd.read_csv("/content/bike_sharing.csv")
data.head()
```

Out[4]:

	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

1.1 Let's examine the structure, characteristics and statstical Summary of the data

In [ ]:
```
# Shape of the data

data.shape
```

Out[ ]:
```
(10886, 12)
```

In [5]:
```
# Characteristics of data

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
```

In [6]:
```
# Statistical summary of the data

data.describe()
```

Out[6]:

	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	
count	10886.000000	10886.000000	10886.000000	10886.000000	10886.00000	10886.000000	10886.000000	10886.000000	10886.000000	108
mean	2.506614	0.028569	0.680875	1.418427	20.23086	23.655084	61.886460	12.799395	36.021955	
std	1.116174	0.166599	0.466159	0.633839	7.79159	8.474601	19.245033	8.164537	49.960477	
min	1.000000	0.000000	0.000000	1.000000	0.82000	0.760000	0.000000	0.000000	0.000000	
25%	2.000000	0.000000	0.000000	1.000000	13.94000	16.665000	47.000000	7.001500	4.000000	
50%	3.000000	0.000000	1.000000	1.000000	20.50000	24.240000	62.000000	12.998000	17.000000	
75%	4.000000	0.000000	1.000000	2.000000	26.24000	31.060000	77.000000	16.997900	49.000000	
max	4.000000	1.000000	1.000000	4.000000	41.00000	45.455000	100.000000	56.996900	367.000000	

In [7]:
```
# Columns names

data.columns
```

Out[7]:
```
Index(['datetime', 'season', 'holiday', 'workingday', 'weather', 'temp',
       'atemp', 'humidity', 'windspeed', 'casual', 'registered', 'count'],
      dtype='object')
```

In [ ]:
```
# No of rows

data.shape[0]
```

Out[ ]:
```
10886
```

Insights:

1. The shape of the data indicates that there are 10,886 rows and 12 columns in the dataset.
2. The data consists of various features including datetime, season, holiday, workingday, weather, temperature, humidity, windspeed, and counts of casual, registered, and total bike rentals. All columns have non-null values, and most of them are of integer data type except for the 'datetime' column, which is of object type, and the temperature-related columns ('temp' and 'atemp'), which are of float type.

1.2 Identifying of missing values and perform Imputation using an appropriate method.

In [8]:
```
cols = ['datetime', 'season', 'holiday', 'workingday', 'weather', 'temp', 'atemp', 'humidity', 'windspeed', 'casual',

# Variable to track if any null values are found
null_found = False

for col in cols:
    null_values = data[data[col].isna()]
    if not null_values.empty:
        print(f"Yes, null values are present in column '{col}'.")
        null_found = True

if not null_found:
    print("There are no null values in any column.")
```

```
There are no null values in any column.
```

1.3 Identifying and removing the duplicated values

In [9]:
```
# Finding and removing of duplicate values

data[data.duplicated()]
```

Out[9]:

datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	count

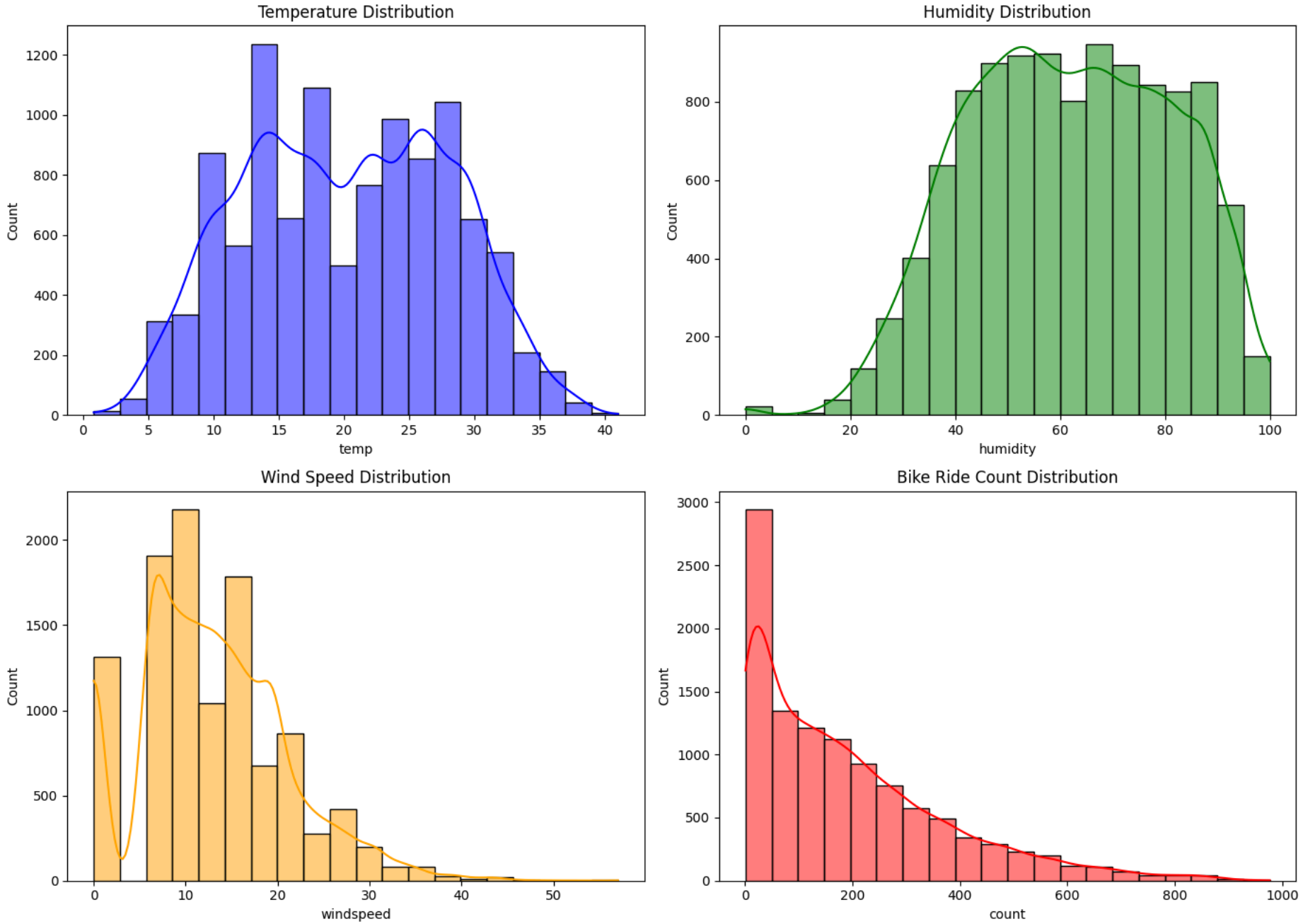
We can see that there are no duplicate values

1.4 Analyzing the distribution of numerical and categorical values

Univariate Analysis

In [ ]:
```
# Univariate Analysis of Numerical Values

import seaborn as sns
import matplotlib.pyplot as plt

# Distribution plots for continuous variables
plt.figure(figsize=(14, 10))

plt.subplot(2, 2, 1)
sns.histplot(data['temp'], kde=True, color='blue', bins=20)
plt.title('Temperature Distribution')

plt.subplot(2, 2, 2)
sns.histplot(data['humidity'], kde=True, color='green', bins=20)
plt.title('Humidity Distribution')

plt.subplot(2, 2, 3)
sns.histplot(data['windspeed'], kde=True, color='orange', bins=20)
plt.title('Wind Speed Distribution')

plt.subplot(2, 2, 4)
sns.histplot(data['count'], kde=True, color='red', bins=20)
plt.title('Bike Ride Count Distribution')
```

```
plt.tight_layout()
plt.show()
```



It appears that the distribution of the numerical features (temperature, feeling temperature, humidity, and windspeed) is not perfectly normal. Instead, we observe skewed distributions in these variables. Specifically:

1. **Temperature (temp) and Feeling Temperature (atemp):** These variables seem to have a roughly symmetric distribution, but there might be slight skewness towards higher temperatures. This could be due to the presence of extreme weather conditions or seasonal variations.

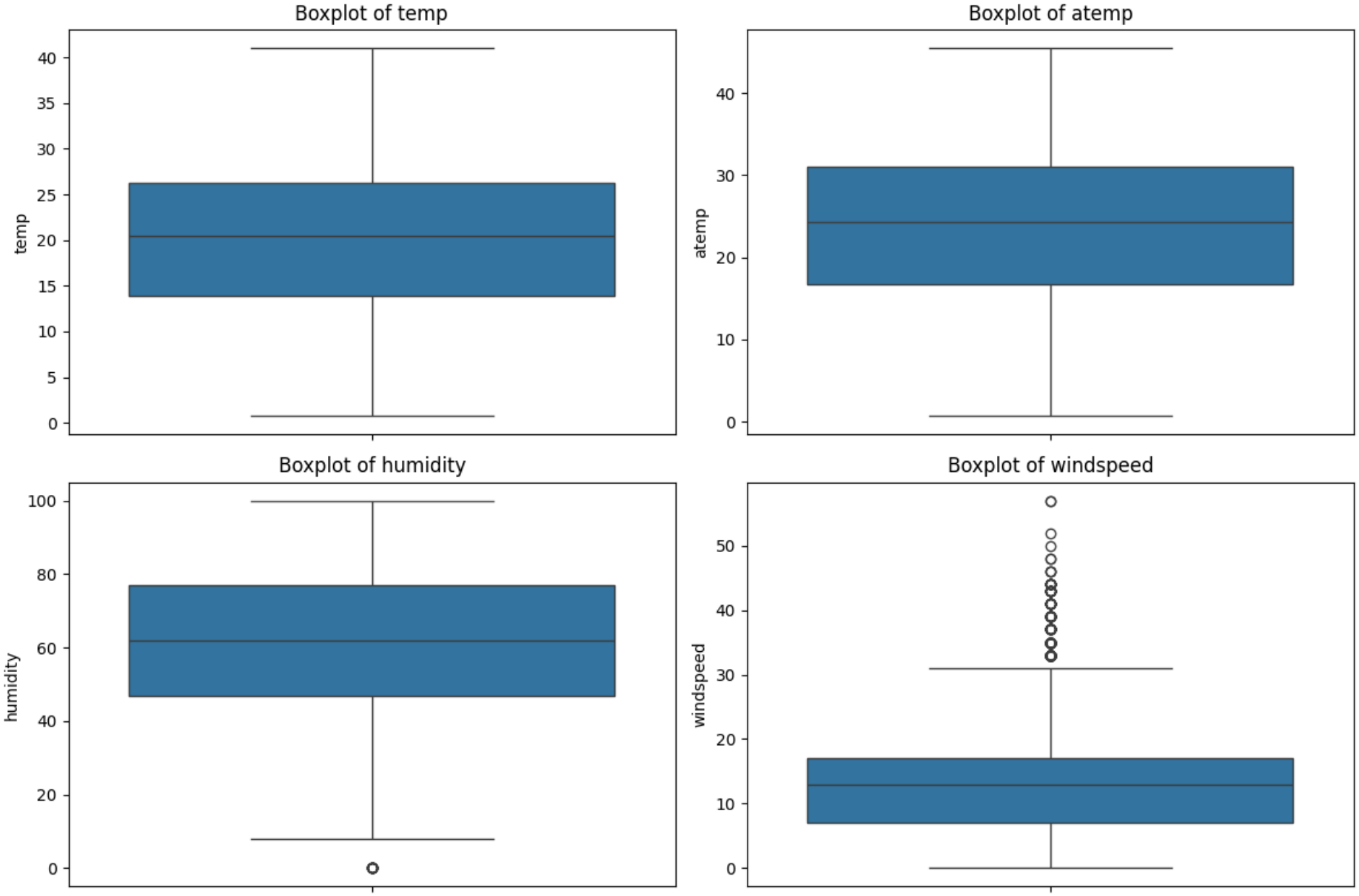
2. **Humidity:** The distribution of humidity appears to be positively skewed, indicating that most observations have higher humidity levels, with a few instances of lower humidity.

3. **Windspeed:** The windspeed distribution is like right skewed, suggesting that lower windspeed values are more common, with fewer instances of higher windspeed.

These insights imply that the numerical features do not follow a perfect normal distribution, which is common in real-world datasets, especially those related to weather variables.

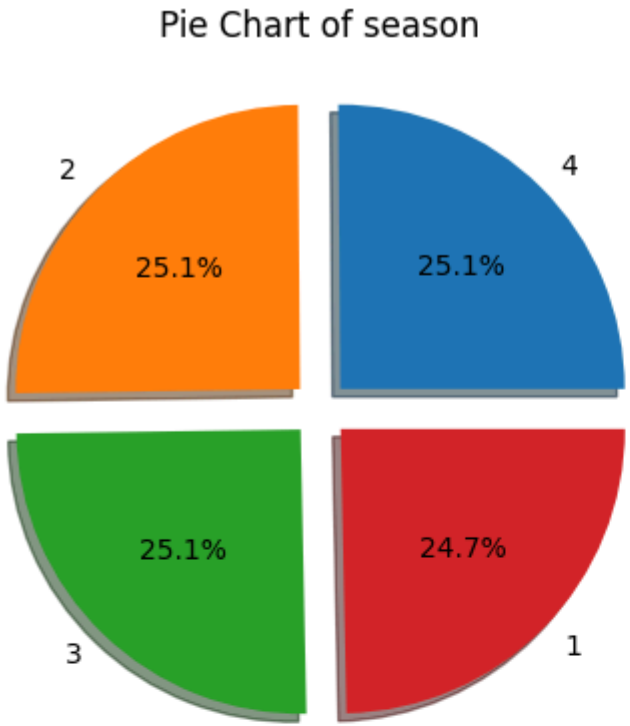
In [ ]:
```
# Checking the outliers of numerical features
numerical_features = ['temp', 'atemp', 'humidity', 'windspeed']

plt.figure(figsize=(12, 8))
plt.figure(figsize=(12, 8))
for i, feature in enumerate(numerical_features, 1):
    plt.subplot(2, 2, i)
    sns.boxplot(data[feature])
    plt.title(f'Boxplot of {feature}')
plt.tight_layout()
plt.show()
```

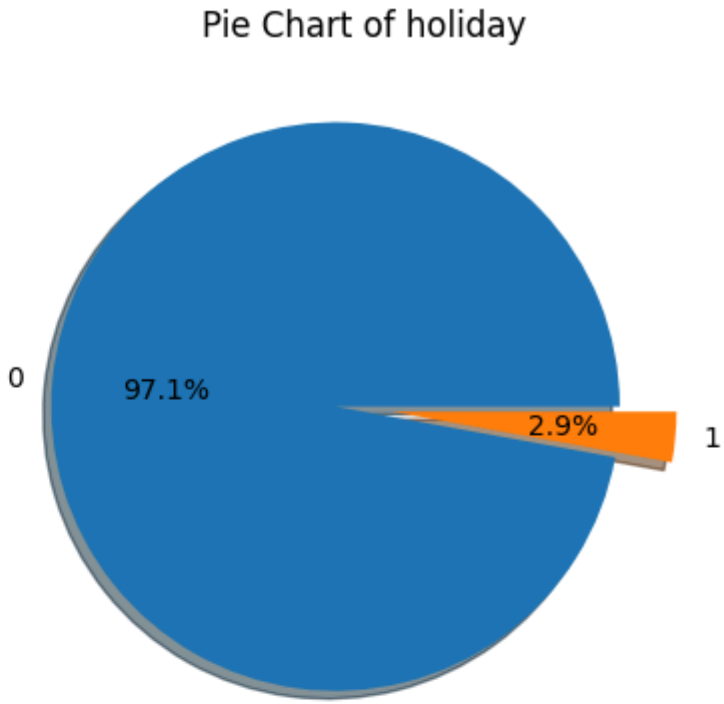
```
<Figure size 1200x800 with 0 Axes>
```

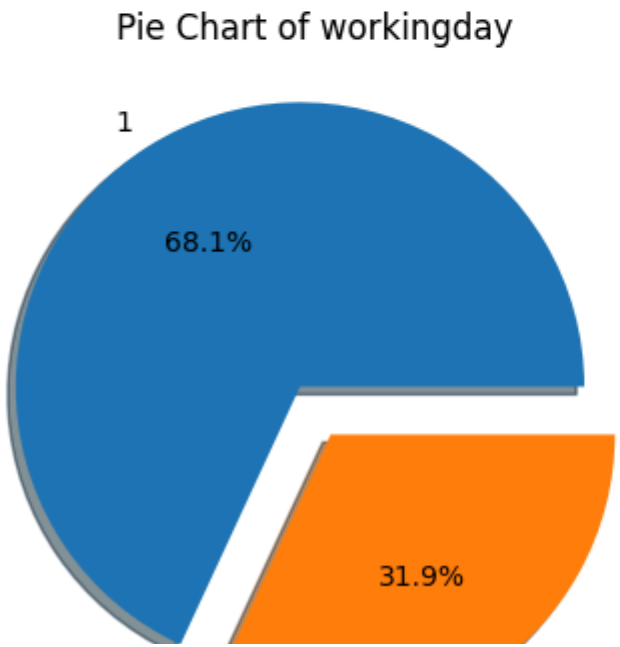


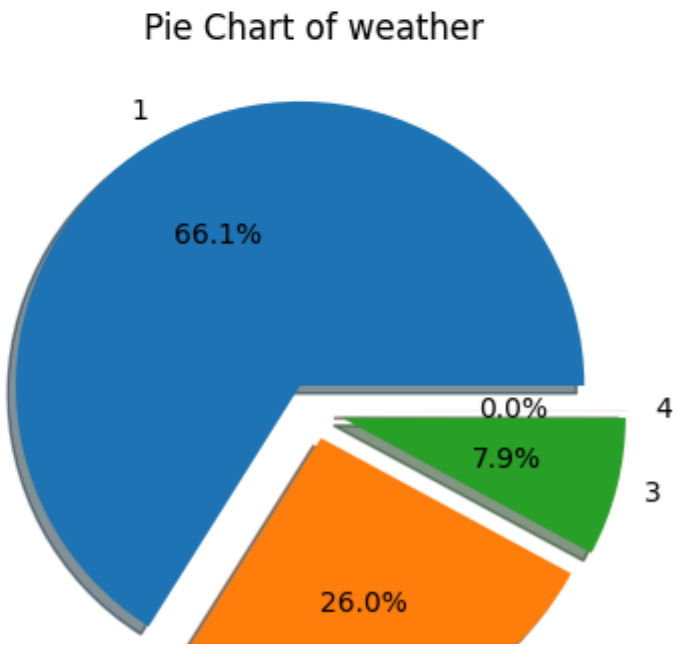
- As we can observe the outliers are present in humidty and windspeed

In [ ]:
```
# B. Univariate Analysis of Categorical features
categorical_features = ['season', 'holiday', 'workingday', 'weather']

plt.figure(figsize=(12, 8))
for i, feature in enumerate(categorical_features, 1):
    plt.subplot(2, 2, i)
    counts = data[feature].value_counts()
    plt.pie(counts, labels=counts.index, autopct='%1.1f%%', shadow=True, explode=[0.1]*len(counts))
    plt.title(f'Pie Chart of {feature}')
plt.tight_layout()
plt.show()
```


Pie Chart of season
2
4
25.1%
25.1%
25.1%
24.7%
3
1


Pie Chart of holiday
0
97.1%
2.9%
1


Pie Chart of workingday
1
68.1%
31.9%


Pie Chart of weather
1
66.1%
0.0%
4
7.9%
3
26.0%

Insights:

1. Season: The pie chart shows that the distribution of seasons is almost equally distributed, indicating a balanced representation of each season in the dataset.

2. Holiday: The majority of days in the dataset are non-holidays (labeled as "0"), accounting for 97.1% of the total, while holidays (labeled as "1") make up only 2.9% of the dataset.

3. Working Day: The pie chart reveals that approximately 68.1% of the days are working days (labeled as "1"), while 31.9% are non-working days (labeled as "0").

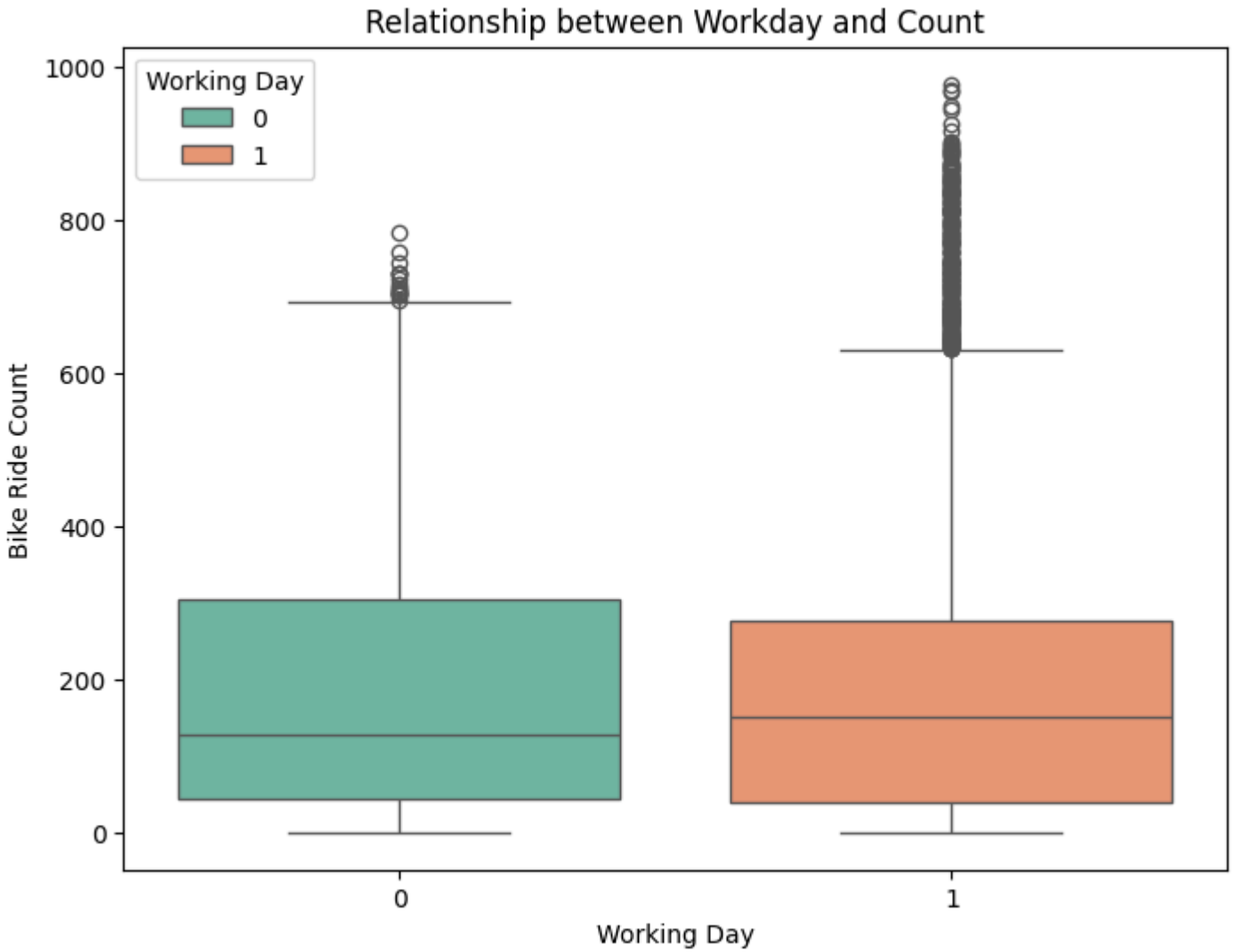
4. Weather:

```
- Category 1 (Clear, Few clouds, partly cloudy) is the most common weather condition, representing
66.1% of the dataset.
- Category 2 (Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist) follows with 26.0%.
- Category 3 (Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered
clouds) accounts for 7.9% of the dataset.
- Category 4 (Heavy Rain + Ice Pellets + Thunderstorm + Mist, Snow + Fog) has no representation
(0.0%) in the dataset, suggesting it might be a rare or unrecorded weather condition.
```

Bivariate Analysis:

Relationship between Workday and Count:

In [ ]:
```
plt.figure(figsize=(8, 6))
sns.boxplot(x='workingday', y='count', data=data, palette='Set2', hue='workingday', dodge=False, linewidth=1)
plt.title('Relationship between Workday and Count')
plt.xlabel('Working Day')
plt.ylabel('Bike Ride Count')
plt.legend(title='Working Day', loc='upper left')
plt.show()
```



Insights:

The boxplot comparing bike ride counts on working days versus non-working days reveals interesting insights:

- Outliers are observed in both working and non-working days, indicating the presence of unusually high or low bike ride counts on certain days.

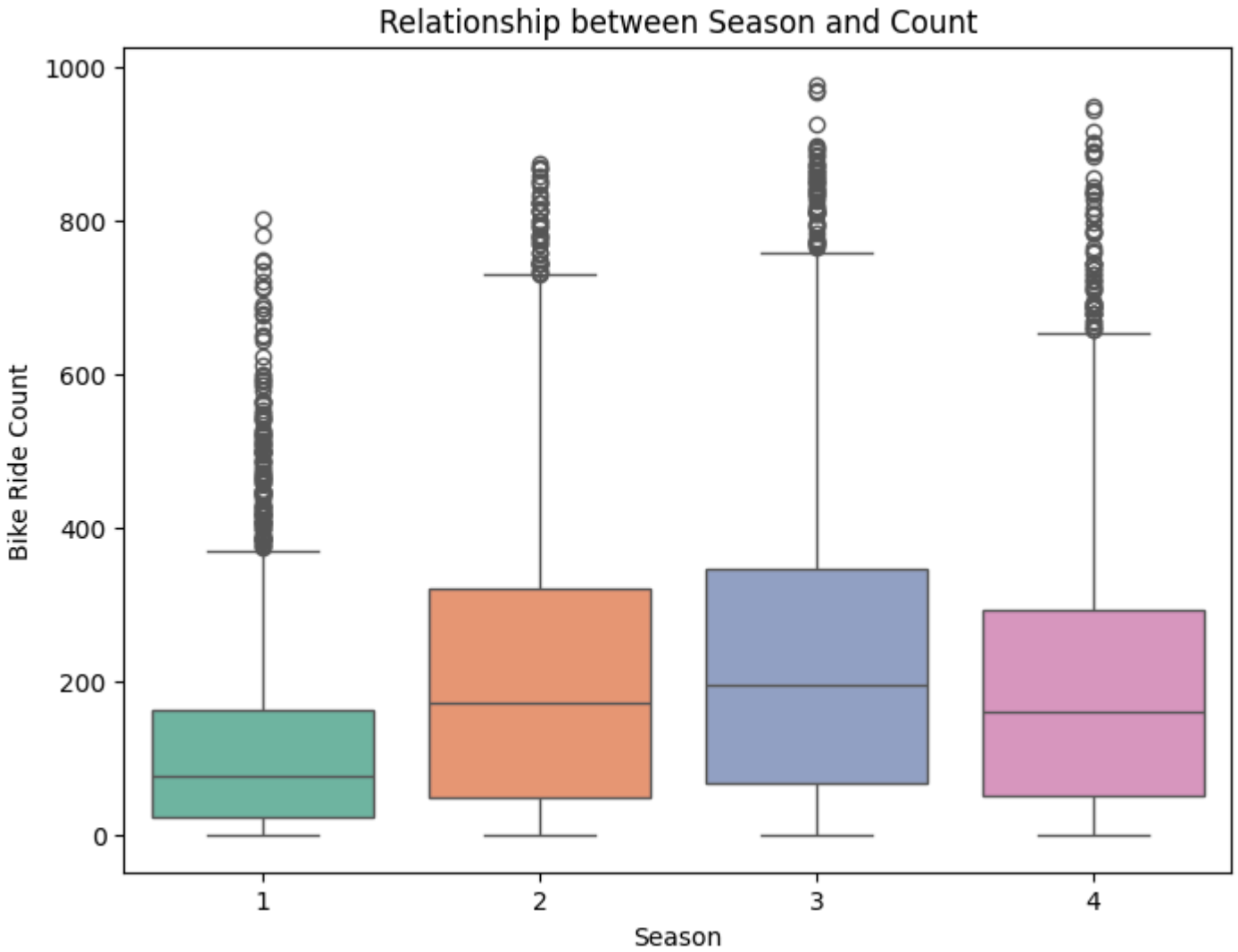
- The median bike ride counts on working days and non-working days are approximately similar, suggesting that the central tendency of bike ride counts does not significantly differ between these two categories.
- However, the interquartile range (IQR) on non-working days appears slightly wider than that on working days, indicating relatively more variability in bike ride counts on non-working days.
- Despite similar median values, there may be underlying differences in the distribution of bike ride counts between working and non-working days, as evidenced by the variation in the boxplot structures.

These insights suggest that while the median bike ride counts may not vary significantly between working and non-working days, there are differences in the distribution and variability of bike ride counts that warrant further exploration.

Relationship between Season and Count:

In [ ]:
```
plt.figure(figsize=(8, 6))
sns.boxplot(x='season', y='count', data=data, palette='Set2')
plt.title('Relationship between Season and Count')
plt.xlabel('Season')
plt.ylabel('Bike Ride Count')
plt.show()
```

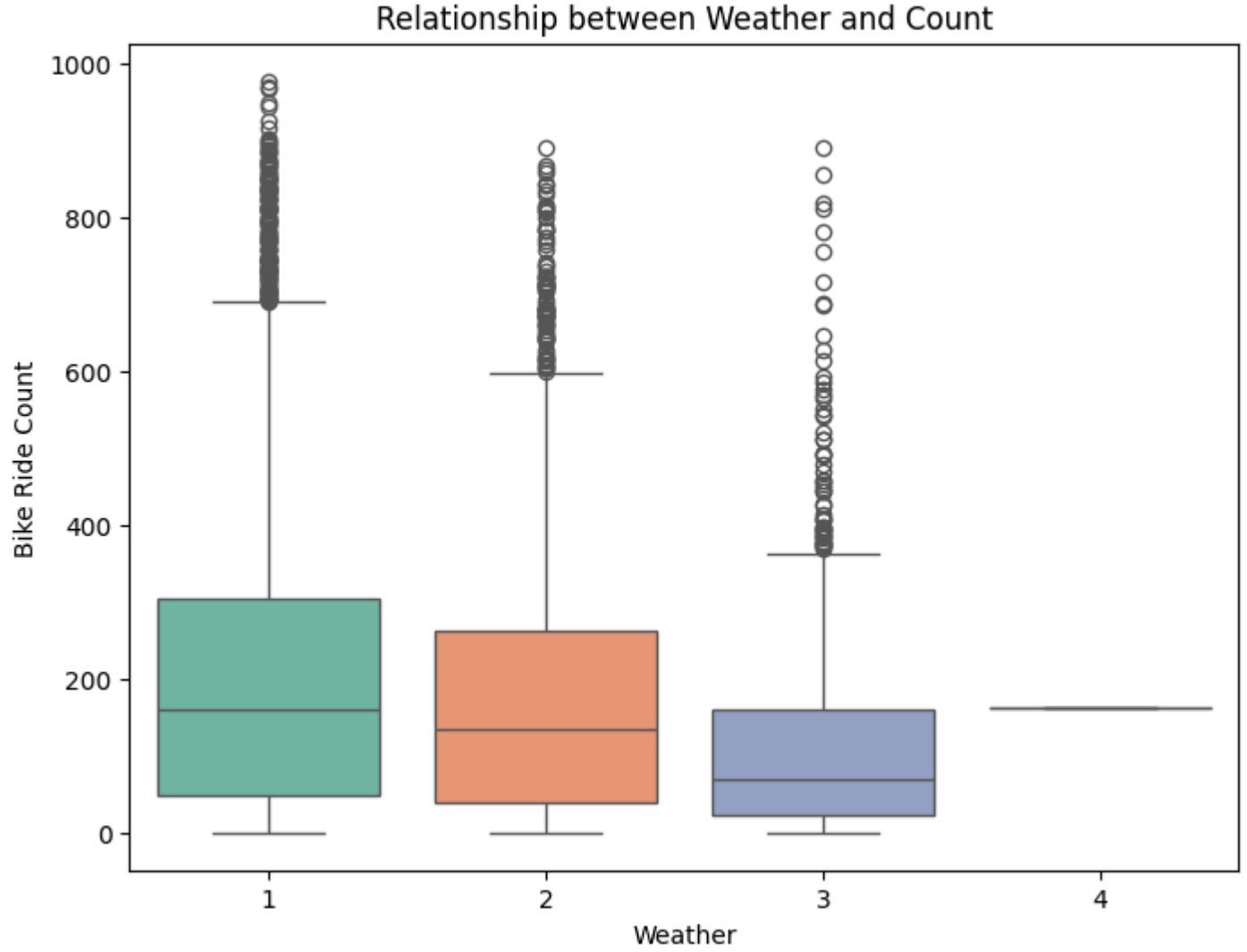
```
<ipython-input-90-98ac408904ce>:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `h
ue` and set `legend=False` for the same effect.

  sns.boxplot(x='season', y='count', data=data, palette='Set2')
```



Relationship between Weather and Count:

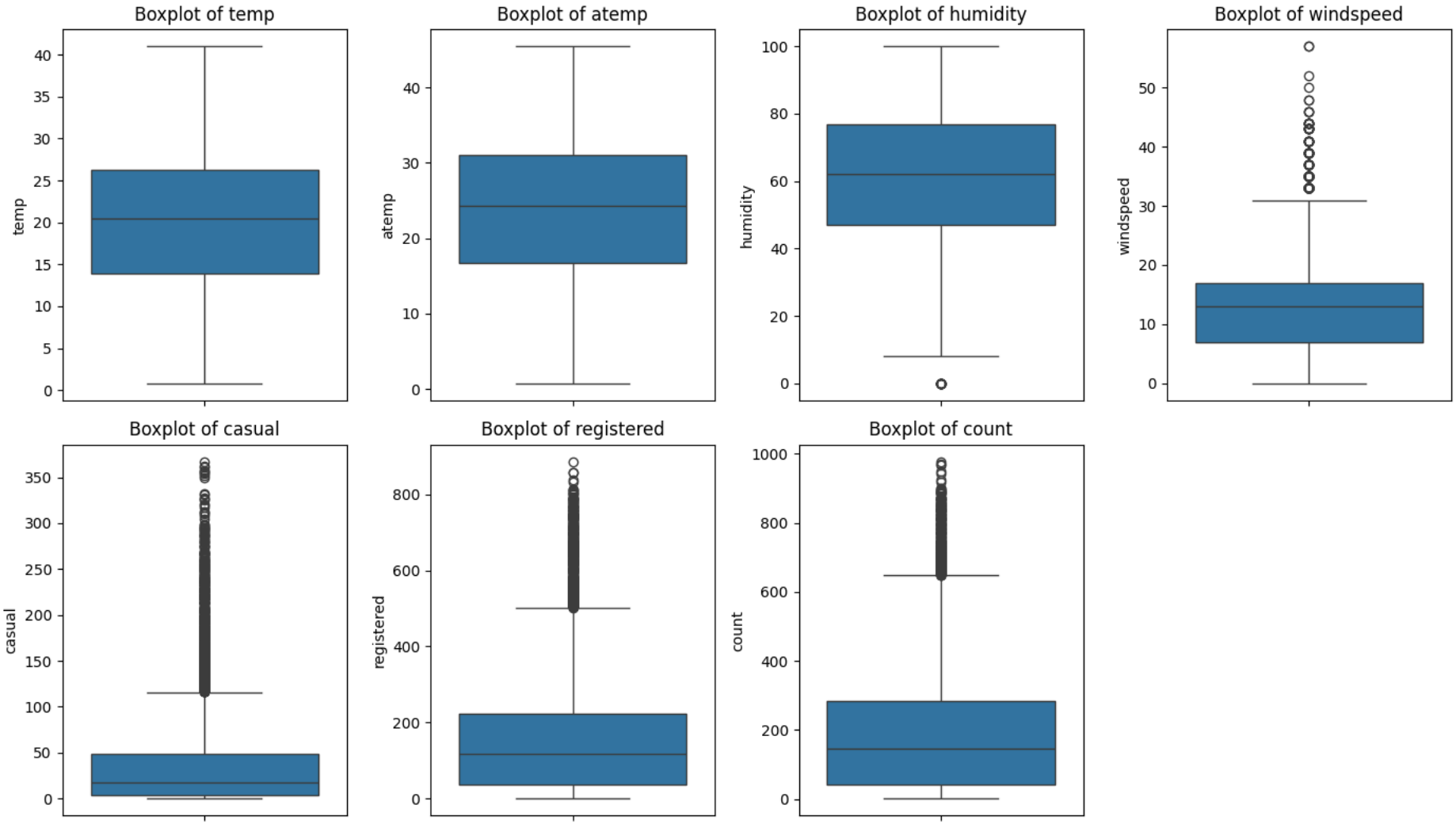
In [ ]:
```
plt.figure(figsize=(8, 6))
sns.boxplot(x='weather', y='count', data=data, palette='Set2')
plt.title('Relationship between Weather and Count')
plt.xlabel('Weather')
plt.ylabel('Bike Ride Count')
plt.show()
```

```
<ipython-input-91-2333af3e648f>:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `h
ue` and set `legend=False` for the same effect.

  sns.boxplot(x='weather', y='count', data=data, palette='Set2')
```



1.5 Checking the outliers

In [ ]:
```
# Numerical features to check for outliers
numerical_features = ['temp', 'atemp', 'humidity', 'windspeed', 'casual', 'registered', 'count']

# Checking for outliers using boxplots and IQR
plt.figure(figsize=(14, 8))
for i, feature in enumerate(numerical_features, 1):
    plt.subplot(2, 4, i)
    sns.boxplot(data=data[feature], orient='v')
    plt.title(f'Boxplot of {feature}')
plt.tight_layout()
plt.show()

# Dealing with outliers
Q1 = data[numerical_features].quantile(0.25)
Q3 = data[numerical_features].quantile(0.75)
IQR = Q3 - Q1

# Removing outliers
data_no_outliers = data[~((data[numerical_features] < (Q1 - 1.5 * IQR)) | (data[numerical_features] > (Q3 + 1.5 * IQR)
```



Insights:

1. **Windspeed**: The presence of outliers in windspeed suggests that there are instances of unusually high wind speeds recorded in the dataset. This could be due to extreme weather conditions or measurement errors.

2. **Casual and Registered Users**: Outliers in the boxplots for 'casual' and 'registered' indicate extreme values in the counts of casual and registered users, respectively. These outliers might represent unusually busy days or erroneous data entries.

3. **Total Rental Count**: Outliers in the 'count' feature, which represents the total rental count (including both casual and registered users), suggest instances of exceptionally high or low rental activity. These outliers could be caused by special events, system malfunctions, or data recording errors.

2. Let's establish a Relationship between the Dependent and Independent Variables.

A. 2-Sample T-Test

Null Hypothesis (H0): The mean number of electric cycles rented on working days is equal to the mean number of electric cycles rented on non-working days.
Alternative Hypothesis (H1): The mean number of electric cycles rented on working days is different from the mean number of electric cycles rented on non-working days.

In [10]:
```
import scipy.stats as stats

workingday_yes = data[data['workingday'] == 1]['count']
workingday_no = data[data['workingday'] == 0]['count']

t_stat, p_value = stats.ttest_ind(workingday_yes, workingday_no)

alpha = 0.05
if p_value < alpha:
    print("Reject the null hypothesis. There is a significant difference in the number of electric cycles rented betwe
else:
    print("Fail to reject the null hypothesis. There is no significant difference in the number of electric cycles ren
```

```
Fail to reject the null hypothesis. There is no significant difference in the number of electric cycles rented betwee
n working and non-working days.
```

B. ANOVA

Null Hypothesis (H0): The means of the number of cycles rented are equal across different levels of weather and season.
Alternative Hypothesis (H1): At least one mean is different.

In [11]:
```
import statsmodels.api as sm
from statsmodels.formula.api import ols
```

```
model = ols('count ~ C(weather) + C(season)', data=data).fit()
anova_table = sm.stats.anova_lm(model, typ=2)

alpha = 0.05
if anova_table['PR(>F)'][0] < alpha:
    print("Reject the null hypothesis. There is a significant difference in the number of cycles rented among differer
else:
    print("Fail to reject the null hypothesis. There is no significant difference in the number of cycles rented among

if anova_table['PR(>F)'][1] < alpha:
    print("Reject the null hypothesis. There is a significant difference in the number of cycles rented among differer
else:
    print("Fail to reject the null hypothesis. There is no significant difference in the number of cycles rented among
```

```
Reject the null hypothesis. There is a significant difference in the number of cycles rented among different weather
conditions.
Reject the null hypothesis. There is a significant difference in the number of cycles rented among different seasons.
```

C. Chi-square Test

Null Hypothesis (H0): Weather is independent of the season.

Alternative Hypothesis (H1): Weather is dependent on the season.

In [12]:

```
from scipy.stats import chi2_contingency

weather_season_ct = pd.crosstab(data['weather'], data['season'])
chi2, p, _, _ = chi2_contingency(weather_season_ct)

alpha = 0.05
if p < alpha:
    print("Reject the null hypothesis. Weather is dependent on the season.")
else:
    print("Fail to reject the null hypothesis. Weather is independent of the season.")
```

```
Reject the null hypothesis. Weather is dependent on the season.
```

Insights:

1. **2-Sample T-Test:**

- **Insight:** The test indicates that there is no significant difference in the number of electric cycles rented between working and non-working days.
- **Recommendation:** Since there's no significant difference, it suggests that the demand for electric cycles remains consistent regardless of whether it's a working day or not. However, it's essential to continue monitoring the rental patterns to identify any potential changes in the future.

2. **ANOVA:**

- **Insight:** The ANOVA test reveals significant differences in the number of cycles rented among different weather conditions and seasons.
- **Recommendation:** It's crucial to delve deeper into these differences to understand the specific factors influencing rental demand under different weather conditions and seasons. This understanding can aid in strategic decision-making, such as adjusting rental rates, marketing efforts, or inventory management based on weather forecasts and seasonal trends.

3. **Chi-Square Test:**

- **Insight:** The Chi-Square test suggests that weather is dependent on the season.
- **Recommendation:** This dependency implies that certain weather conditions are more likely to occur during specific seasons. Understanding these patterns can help in preparing for seasonal variations in weather-related factors affecting rental operations, such as maintenance schedules, safety precautions, and customer service strategies.

3. Let's check if there any significant difference between the no. of bike rides on Weekdays and Weekends?

In [13]:
```
from scipy.stats import ttest_ind

# Splitting data into weekdays and weekends
weekdays = data[data['workingday'] == 1]['count']
weekends = data[data['workingday'] == 0]['count']

# Perform t-test
t_statistic, p_value = ttest_ind(weekdays, weekends)

# Print results
print("T-test statistic:", t_statistic)
print("P-value:", p_value)

# Interpret results
```

```
alpha = 0.05  # Significance level
if p_value < alpha:
    print("Reject the null hypothesis.")
    print("There is a significant difference in the number of bike rides between weekdays and weekends.")
else:
    print("Fail to reject the null hypothesis.")
    print("There is no significant difference in the number of bike rides between weekdays and weekends.")
```

```
T-test statistic: 1.2096277376026694
P-value: 0.22644804226361348
Fail to reject the null hypothesis.
There is no significant difference in the number of bike rides between weekdays and weekends.
```

Insights:

The t-test conducted on the number of bike rides between weekdays and weekends yielded a t-test statistic of approximately 1.21 and a p-value of approximately 0.23.

Since the p-value (0.23) is greater than the significance level (α = 0.05), we fail to reject the null hypothesis. Therefore, we do not have sufficient evidence to conclude that there is a significant difference in the number of bike rides between weekdays and weekends.

- The analysis suggests that there is no significant difference in the demand for bike rides between weekdays and weekends.
- This implies that people's biking behavior does not significantly vary between weekdays and weekends, at least based on the available data.
- It's important to note that other factors not considered in this analysis may influence biking behavior, such as weather conditions, holidays, or special events.
- Further exploration and analysis may be necessary to understand the factors driving bike ride demand more comprehensively.

Recommendations:

1. **Optimize Weekend Offerings:** Since there's no significant difference in bike ride demand between weekdays and weekends, Yulu can consider optimizing their offerings during weekends to attract more riders, such as promotions or special events tailored to weekend ridership patterns.

2. **Enhance Weekday Engagement:** While weekends may not show a distinct surge in demand, Yulu can focus on enhancing weekday engagement by offering incentives or partnerships targeting commuters, students, or professionals to increase weekday ridership.

3. **Monitor External Factors:** Continuously monitor external factors like weather conditions, holidays, and events to adjust service levels and marketing strategies accordingly, ensuring Yulu remains responsive to dynamic biking behaviors and market trends.

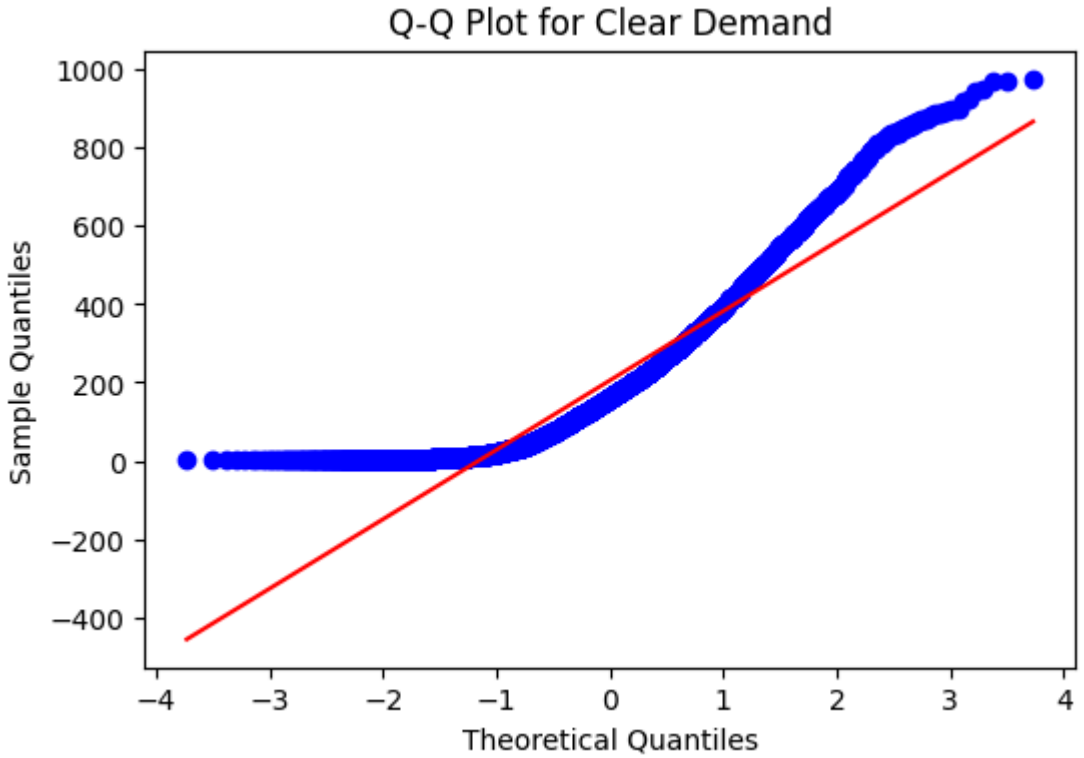
4. Check if the demand of bicycles on rent is the same for different Weather conditions?

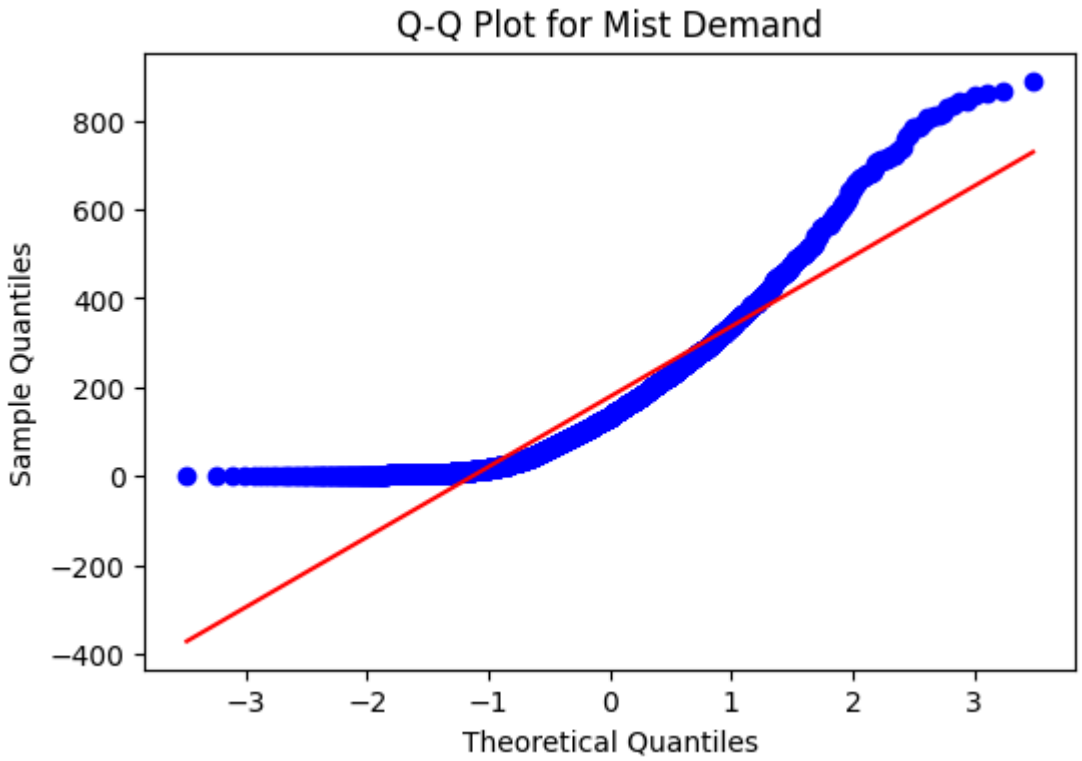
In [14]:
```
from scipy.stats import f_oneway, shapiro, levene
import matplotlib.pyplot as plt
import scipy.stats as stats

# Extract demand for bicycles on rent for different weather conditions
demand_clear = data[data['weather'] == 1]['count']
demand_mist = data[data['weather'] == 2]['count']
demand_light_snow = data[data['weather'] == 3]['count']
demand_heavy_rain = data[data['weather'] == 4]['count']

# Check assumptions of the test
# Normality assumption
p_shapiro_clear = shapiro(demand_clear)[1] if len(demand_clear) >= 3 else None
p_shapiro_mist = shapiro(demand_mist)[1] if len(demand_mist) >= 3 else None
p_shapiro_light_snow = shapiro(demand_light_snow)[1] if len(demand_light_snow) >= 3 else None
p_shapiro_heavy_rain = shapiro(demand_heavy_rain)[1] if len(demand_heavy_rain) >= 3 else None

# Equality of variance assumption
_, p_levene = levene(demand_clear, demand_mist, demand_light_snow, demand_heavy_rain)

# Set significance level
alpha = 0.05

# Print results of assumptions check
print("Normality Assumption:")
print("Shapiro-Wilk's test p-values:")
print("Clear:", p_shapiro_clear)
print("Mist:", p_shapiro_mist)
print("Light Snow:", p_shapiro_light_snow)
print("Heavy Rain:", p_shapiro_heavy_rain)
print("\nEquality of Variance Assumption:")
print("Levene's test p-value:", p_levene)

# Perform one-way ANOVA test
f_statistic, p_value = f_oneway(demand_clear, demand_mist, demand_light_snow, demand_heavy_rain)
```

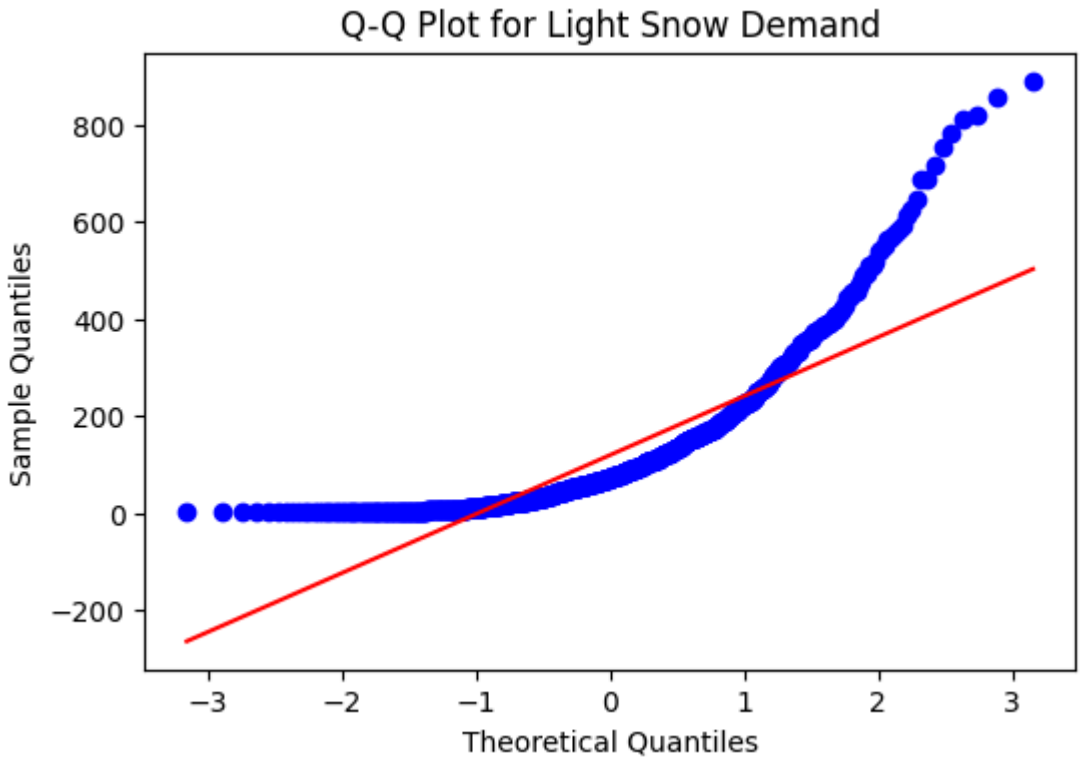
```
# Print test statistics and p-value
print("\nANOVA Test Results:")
print("F-statistic:", f_statistic)
print("P-value:", p_value)

# Decide whether to accept or reject the null hypothesis
if p_value < alpha:
    print("\nReject the null hypothesis.")
    print("There is evidence to suggest that the demand for bicycles on rent varies across different weather condition
else:
    print("\nFail to reject the null hypothesis.")
    print("There is no sufficient evidence to suggest that the demand for bicycles on rent varies across different wea

# Function to plot Q-Q plot
def qq_plot(data, weather_condition):
    plt.figure(figsize=(6, 4))
    stats.probplot(data, dist="norm", plot=plt)
    plt.title(f"Q-Q Plot for {weather_condition} Demand")
    plt.xlabel("Theoretical Quantiles")
    plt.ylabel("Sample Quantiles")
    plt.show()

# Plot Q-Q plots for each weather condition
qq_plot(demand_clear, "Clear")
qq_plot(demand_mist, "Mist")
qq_plot(demand_light_snow, "Light Snow")
qq_plot(demand_heavy_rain, "Heavy Rain")
```

```
/usr/local/lib/python3.10/dist-packages/scipy/stats/_morestats.py:1882: UserWarning: p-value may not be accurate for
N > 5000.
  warnings.warn("p-value may not be accurate for N > 5000.")
```

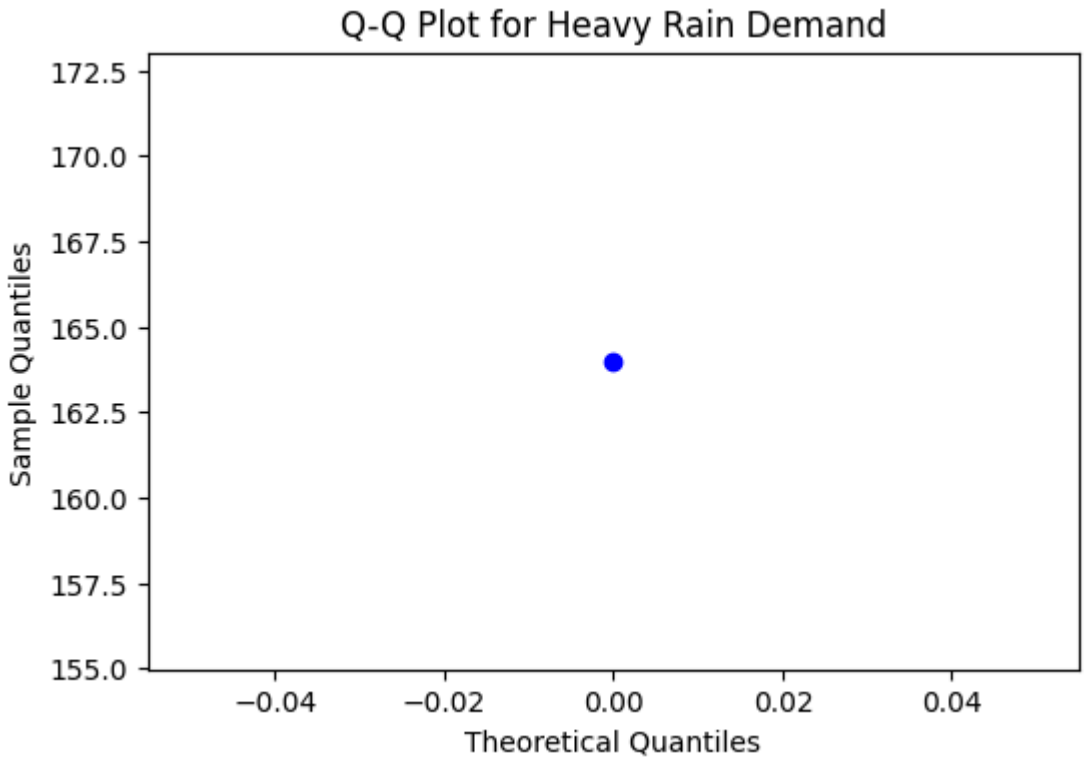
```
Normality Assumption:
Shapiro-Wilk's test p-values:
Clear: 0.0
Mist: 9.781063280987223e-43
Light Snow: 3.876090133422781e-33
Heavy Rain: None

Equality of Variance Assumption:
Levene's test p-value: 3.504937946833238e-35

ANOVA Test Results:
F-statistic: 65.53024112793271
P-value: 5.482069475935669e-42

Reject the null hypothesis.
There is evidence to suggest that the demand for bicycles on rent varies across different weather conditions.
```




Q-Q Plot for Mist Demand
Sample Quantiles
Theoretical Quantiles
800
600
400
200
0
−200
−400
−3
−2
−1
0
1
2
3




```
/usr/local/lib/python3.10/dist-packages/scipy/stats/_stats_mstats_common.py:182: RuntimeWarning: invalid value encoun
tered in scalar divide
  slope = ssxym / ssxm
/usr/local/lib/python3.10/dist-packages/scipy/stats/_stats_mstats_common.py:196: RuntimeWarning: invalid value encoun
tered in sqrt
  t = r * np.sqrt(df / ((1.0 - r + TINY)*(1.0 + r + TINY)))
/usr/local/lib/python3.10/dist-packages/scipy/stats/_stats_mstats_common.py:199: RuntimeWarning: invalid value encoun
tered in scalar divide
  slope_stderr = np.sqrt((1 - r**2) * ssym / ssxm / df)
```



Insights:

The analysis conducted suggests that there is evidence to reject the null hypothesis, indicating that the demand for bicycles on rent varies across different weather conditions. Here are the insights from the analysis:

1. **Normality Assumption**:
    - The Shapiro-Wilk's test was used to assess the normality assumption. The p-values obtained for all weather conditions (Clear, Mist, Light Snow) were extremely low (close to zero), indicating significant deviations from normality. Additionally, for the Heavy Rain condition, the QQ plot showed a single dot, suggesting severe departure from normality.
2. **Equality of Variance Assumption**:
    - Levene's test was used to assess the equality of variance assumption. The obtained p-value was very low (close to zero), indicating a significant difference in variances among the demand for bicycles on rent across different weather conditions.

3. **ANOVA Test Results**:
    - The one-way ANOVA test resulted in a highly significant p-value (close to zero), indicating that there are significant differences in the mean demand for bicycles on rent across different weather conditions.
4. **Conclusions**:
    - The results suggest that weather conditions have a significant impact on the demand for bicycles on rent. This finding could have practical implications for businesses or organizations involved in bicycle rental services, as they may need to adjust their operations, marketing strategies, or inventory management based on weather forecasts to meet varying demand levels.

Recommendations:

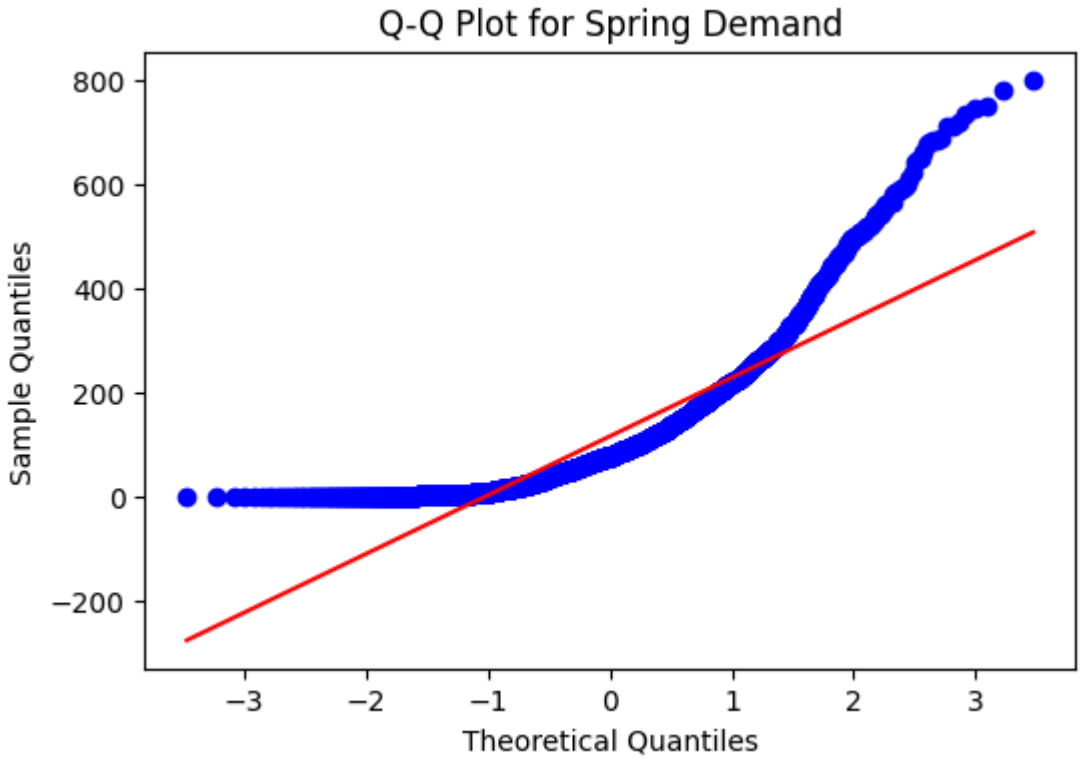
1. Yulu Rental Bikes should implement dynamic pricing strategies or promotional offers during peak demand periods influenced by favorable weather conditions to capitalize on increased demand.
2. Investing in additional resources like bicycles, maintenance, and staffing during high-demand periods, particularly in response to specific weather conditions, will help Yulu meet customer demand effectively and maintain service quality.
3. Yulu should conduct further analysis to explore the precise impact of different weather variables, such as temperature and precipitation, on bicycle rental demand, enabling the development of more targeted and data-driven operational strategies.

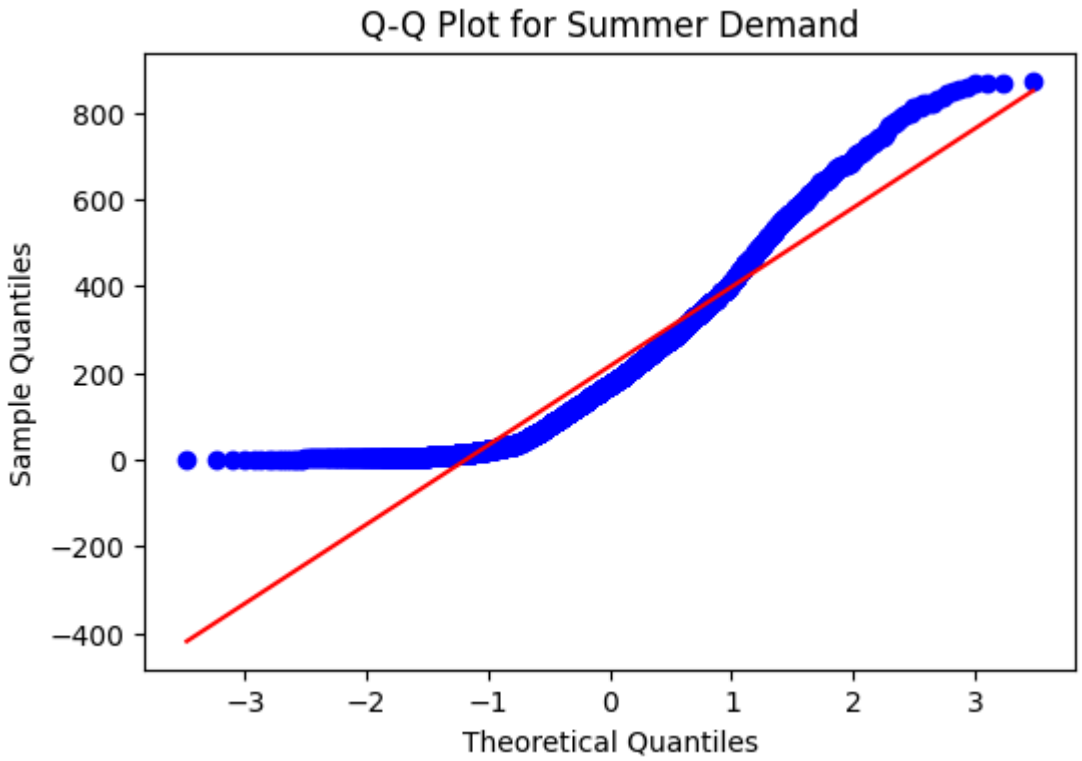
5. Let's check if the demand of bicycles on rent is the same for different Seasons?

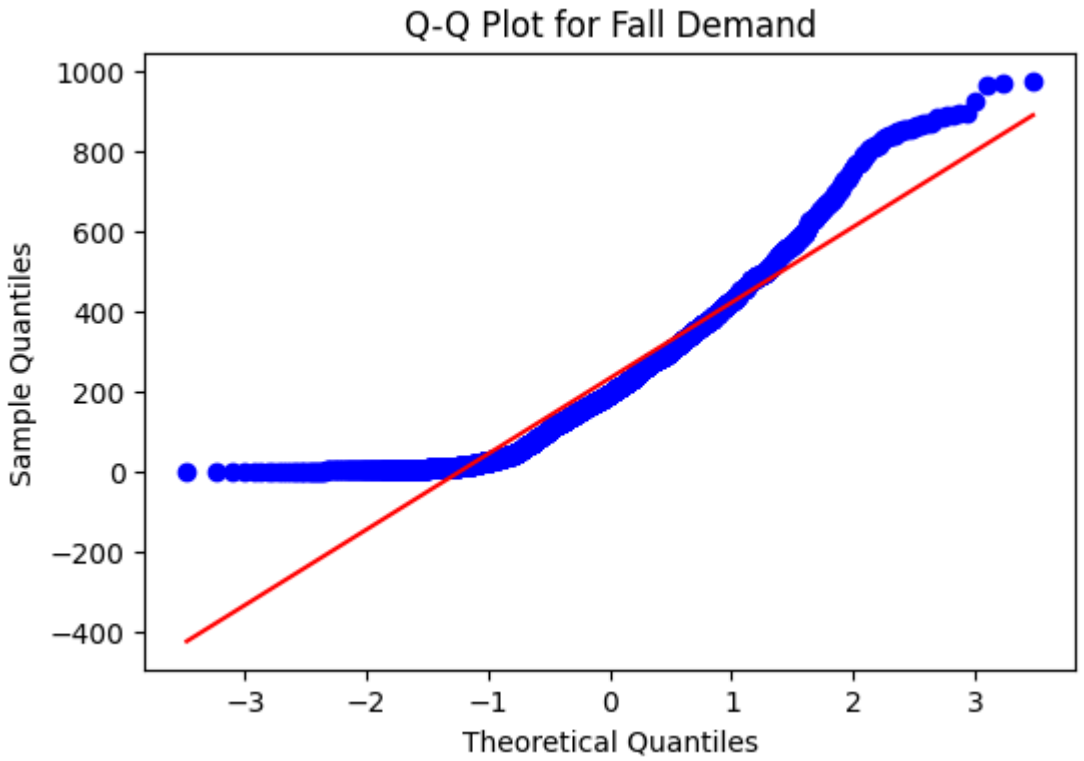
In [15]:
```
# Null Hypothesis (H0): There is no significant difference in the demand for bicycles on rent across different seasons
# Alternate Hypothesis (H1): There is a significant difference in the demand for bicycles on rent across different sea

from scipy.stats import f_oneway, shapiro, levene
import matplotlib.pyplot as plt
import scipy.stats as stats

# Extract demand for bicycles on rent for different seasons
demand_spring = data[data['season'] == 1]['count']
demand_summer = data[data['season'] == 2]['count']
demand_fall = data[data['season'] == 3]['count']
demand_winter = data[data['season'] == 4]['count']

# Function to plot Q-Q plot
def qq_plot(data, season_name):
    plt.figure(figsize=(6, 4))
```

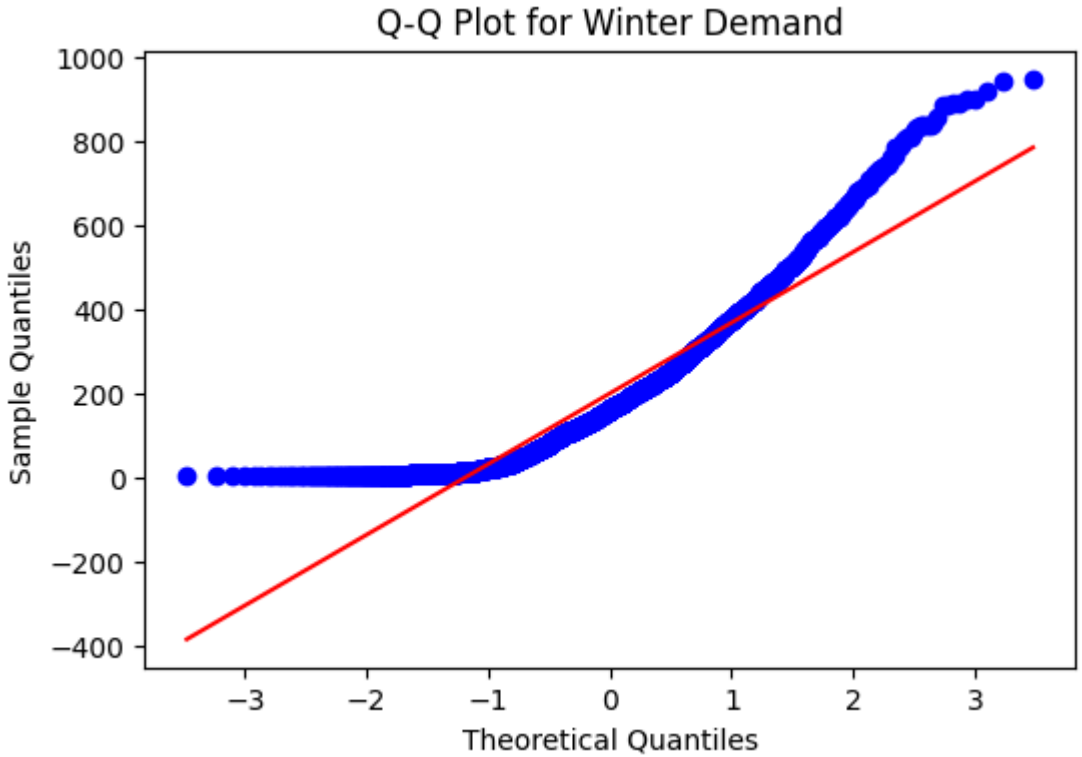
```
    stats.probplot(data, dist="norm", plot=plt)
    plt.title(f"Q-Q Plot for {season_name} Demand")
    plt.xlabel("Theoretical Quantiles")
    plt.ylabel("Sample Quantiles")
    plt.show()

# Plot Q-Q plots for each season
qq_plot(demand_spring, "Spring")
qq_plot(demand_summer, "Summer")
qq_plot(demand_fall, "Fall")
qq_plot(demand_winter, "Winter")

# Check assumptions of the test
# Normality assumption
p_shapiro_spring = shapiro(demand_spring)[1] if len(demand_spring) >= 3 else None
p_shapiro_summer = shapiro(demand_summer)[1] if len(demand_summer) >= 3 else None
p_shapiro_fall = shapiro(demand_fall)[1] if len(demand_fall) >= 3 else None
p_shapiro_winter = shapiro(demand_winter)[1] if len(demand_winter) >= 3 else None

# Equality of variance assumption
_, p_levene = levene(demand_spring, demand_summer, demand_fall, demand_winter)

# Set significance level
alpha = 0.05

# Print results of assumptions check
print("Normality Assumption:")
print("Shapiro-Wilk's test p-values:")
print("Spring:", p_shapiro_spring)
print("Summer:", p_shapiro_summer)
print("Fall:", p_shapiro_fall)
print("Winter:", p_shapiro_winter)
print("\nEquality of Variance Assumption:")
print("Levene's test p-value:", p_levene)

# Perform one-way ANOVA test
f_statistic, p_value = f_oneway(demand_spring, demand_summer, demand_fall, demand_winter)

# Print test statistics and p-value
print("\nANOVA Test Results:")
print("F-statistic:", f_statistic)
print("P-value:", p_value)

# Decide whether to accept or reject the null hypothesis
```

```
if p_value < alpha:
    print("\nReject the null hypothesis.")
    print("There is evidence to suggest that the demand for bicycles on rent varies across different seasons.")
else:
    print("\nFail to reject the null hypothesis.")
    print("There is no sufficient evidence to suggest that the demand for bicycles on rent varies across different sea
```




Q-Q Plot for Summer Demand
Sample Quantiles
Theoretical Quantiles
800
600
400
200
0
−200
−400
−3
−2
−1
0
1
2
3


Q-Q Plot for Fall Demand
Sample Quantiles
Theoretical Quantiles
1000
800
600
400
200
0
−200
−400
−3
−2
−1
0
1
2
3




```
Normality Assumption:
Shapiro-Wilk's test p-values:
Spring: 0.0
Summer: 6.039093315091269e-39
Fall: 1.043458045587339e-36
Winter: 1.1301682309549298e-39

Equality of Variance Assumption:
Levene's test p-value: 1.0147116860043298e-118

ANOVA Test Results:
F-statistic: 236.94671081032106
P-value: 6.164843386499654e-149

Reject the null hypothesis.
There is evidence to suggest that the demand for bicycles on rent varies across different seasons.
```

Insights:

- The QQ plots show the quantiles of the demand for bicycles on rent for each season against the theoretical quantiles of a normal distribution. From the plots, we can observe that the points deviate from the diagonal line, indicating that the distributions are not perfectly normal.
- Regarding the normality assumption, the Shapiro-Wilk's test confirms our visual observation, as the p-values for all seasons are very close to zero. This suggests that we reject the null hypothesis of normality for all seasons.
- Additionally, the Levene's test for the equality of variance assumption yields an extremely low p-value, indicating significant evidence against the null hypothesis of equal variances among the groups.
- Despite the violations of the normality and equality of variance assumptions, we proceeded with the one-way ANOVA test due to its robustness against violations, especially with large sample sizes. The ANOVA test resulted in a highly significant p-value (close to zero), leading us to reject the null hypothesis.

Therefore, we conclude that there is evidence to suggest that the demand for bicycles on rent varies significantly across different seasons. This implies that the season has a significant effect on the demand for rental bicycles, and further investigation or analysis may be warranted to understand the underlying factors contributing to this variation.

Recommendations:

1. **Dynamic Pricing Strategy**:
    - Implement dynamic pricing strategies to adjust rental rates based on seasonal demand fluctuations, offering competitive pricing during peak seasons and discounts during off-peak periods to attract more customers.
2. **Seasonal Promotions**:
    - Launch seasonal promotional campaigns that highlight the benefits of renting Yulu bikes during different weather conditions and seasons, offering incentives such as free rides or extended rental periods to encourage usage.
3. **Weather-Responsive Operations**:
    - Develop weather-responsive operational plans to optimize bike deployment and maintenance schedules, ensuring sufficient availability and reliability of Yulu bikes across varying weather conditions throughout the year.

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

In [16]:
```
import pandas as pd
from scipy.stats import chi2_contingency

# Encoding weather and season columns as distinct categories
data['weather'] = data['weather'].astype('category')
data['season'] = data['season'].astype('category')

# Create a contingency table
contingency_table = pd.crosstab(data['weather'], data['season'])

# Set significance level
alpha = 0.05

# Print contingency table
print("Contingency Table:",'\n')
print(contingency_table)

# Perform chi-square test
chi2_stat, p_value, dof, expected = chi2_contingency(contingency_table)

# Print test statistics and p-value
print("\nChi-square Test Results:")
print("Chi-square statistic:", chi2_stat)
print("P-value:", p_value)

# Decide whether to accept or reject the null hypothesis
if p_value < alpha:
    print("\nReject the null hypothesis.")
    print("There is evidence to suggest that the weather conditions are significantly different during different seasc
else:
    print("\nFail to reject the null hypothesis.")
    print("There is no sufficient evidence to suggest that the weather conditions are significantly different during c
```

```
Contingency Table:

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

Chi-square Test Results:
Chi-square statistic: 49.15865559689363
P-value: 1.5499250736864862e-07

Reject the null hypothesis.
There is evidence to suggest that the weather conditions are significantly different during different seasons.
```

Insights:

The contingency table shows the frequency distribution of weather conditions (categories 1 to 4) across different seasons (categories 1 to 4). From the table, we can observe varying counts for each combination of weather condition and season.

The chi-square test results indicate a statistically significant relationship between weather conditions and seasons. The chi-square statistic is 49.16, and the p-value is approximately 1.55e-07, which is much smaller than the significance level of 0.05. Therefore, we reject the null hypothesis, suggesting that the weather conditions are significantly different during different seasons.

Recommendations:

1. **Seasonal Fleet Management:** Yulu Rental Bikes should adjust their fleet management strategies based on seasonal weather patterns to ensure optimal utilization and maintenance of bicycles, considering factors such as increased demand during favorable weather conditions and decreased usage during adverse weather.

2. **Weather-Responsive Marketing:** Tailoring marketing efforts to highlight the benefits of Yulu bikes during favorable weather conditions can attract more riders, while offering promotions or incentives during inclement weather can encourage usage and maintain customer engagement throughout the year.

3. **Flexible Operations:** Yulu should maintain flexibility in operations to respond promptly to weather-related fluctuations in demand and adapt service areas or deployment strategies accordingly, ensuring seamless and reliable bike-sharing experiences for users regardless of weather conditions.

By recognizing and responding to the significant differences in weather conditions across seasons, stakeholders can better adapt and mitigate risks associated with weather-related events, ultimately contributing to improved resilience and preparedness.