Yulu- Business Case Study

About Yulu

- Yulu is India's leading micro-mobility service provider, which offers unique vehicles for the daily commute.
- The bikes have a unisex frame and come in a standard bright blue shade for easy identification.
- Starting off as a mission to eliminate traffic congestion in India, Yulu provides the safest commute solution through a user-friendly mobile app to enable shared, solo and sustainable commuting.
- Yulu zones are located at all the appropriate locations at metro stations, bus stands, office spaces, residential areas, corporate offices, etc) to make those first and last miles smooth, affordable, and convenient!

In [108]:

```
# Importing essential libraries
import pandas as pd
import numpy as np
import seaborn as sns
sns.set_theme(style="whitegrid")
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
#--------------------------------
from scipy.stats import ttest_1samp,ttest_ind
from scipy.stats import chisquare, chi2_contingency
from scipy.stats import f_oneway, kruskal, shapiro, levene
from statsmodels.graphics.gofplots import qqplot
```

In [109]:
```
df = pd.read_csv('C:/Users/hp/OneDrive/Desktop/Yulu/yulu.csv')
df.head()
```

Out[109]:

	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	reg
0	2011-01-01 00:00:00	1	0	0	1	9.84	14.395	81	0.0	3	
1	2011-01-01 01:00:00	1	0	0	1	9.02	13.635	80	0.0	8	
2	2011-01-01 02:00:00	1	0	0	1	9.02	13.635	80	0.0	5	
3	2011-01-01 03:00:00	1	0	0	1	9.84	14.395	75	0.0	3	
4	2011-01-01 04:00:00	1	0	0	1	9.84	14.395	75	0.0	0	

Problem Statement and Analysing basic metrics

1. Yulu has recently suffered considerable dips in its revenues.
2. Company wants to understand the factors on which the demand for these shared electric cycles depends.
3. Specifically, they want to understand the factors affecting the demand for these shared electric cycles in the Indian market.
4. Which variables are significant in predicting the demand for shared electric cycles in the Indian market?

In [110]:
```
df.shape
```

Out[110]:
```
(10886, 12)
```

- Data given contains 12 features(columns) and 10886 rows of observations

In [111]:
```
# List of all columns in given data
df.columns
```

Out[111]:
```
Index(['datetime', 'season', 'holiday', 'workingday', 'weather', 'temp',
       'atemp', 'humidity', 'windspeed', 'casual', 'registered', 'count'],
      dtype='object')
```

In [112]:
```
# Statistical summary
df.dtypes
```

Out[112]:
```
datetime       object
season          int64
holiday         int64
workingday      int64
weather         int64
temp          float64
atemp         float64
humidity        int64
windspeed     float64
casual          int64
registered      int64
count           int64
dtype: object
```

In [113]:
```
df.describe()
```

Out[113]:

	season	holiday	workingday	weather	temp	atemp	hu
count	10886.000000	10886.000000	10886.000000	10886.000000	10886.00000	10886.000000	10886.0
mean	2.506614	0.028569	0.680875	1.418427	20.23086	23.655084	61.8
std	1.116174	0.166599	0.466159	0.633839	7.79159	8.474601	19.2
min	1.000000	0.000000	0.000000	1.000000	0.82000	0.760000	0.0
25%	2.000000	0.000000	0.000000	1.000000	13.94000	16.665000	47.0
50%	3.000000	0.000000	1.000000	1.000000	20.50000	24.240000	62.0
75%	4.000000	0.000000	1.000000	2.000000	26.24000	31.060000	77.0
max	4.000000	1.000000	1.000000	4.000000	41.00000	45.455000	100.0

In [114]:
```
df.info()
```

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

In [115]:
```
df[df.duplicated()]
```

Out[115]:

datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	regi

- There are no duplicates available in given data.

In [116]:
```
df.isna().sum()
```

Out[116]:
```
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
```

In [117]:
```
df['count'].count()
```

Out[117]:
```
10886
```

In [118]:
```
print('Registered users are: ',round(df["registered"].sum()/df["count"].sum()*100
print('Casual users are: ',round(df["casual"].sum()/df["count"].sum()*100),'%')
```

```
Registered users are:  81 %
Casual users are:  19 %
```

In [119]:
```
print(df['datetime'].min(),'-----' , df['datetime'].max())
```

```
2011-01-01 00:00:00 ----- 2012-12-19 23:00:00
```

Observations

1. There are no missing values in dataset. So no need to handle missing values.
2. Except datetime all other columns are having integer or float data type.
3. There are no duplicates in given data.
4. Mean and 50 percentile value of temp, atemp, humidity, windspeed columns are not having much differnce
5. There is large differene in mean and 50 percentile value of casual, registered and count column
6. 81% of users are registered while 19% are casual users

Converting Season, Weather, Workingday and holiday column to categories

In [120]:
```
df.loc[df['season'] == 1,'Season']='spring'
df.loc[df['season'] == 2,'Season']='summer'
df.loc[df['season'] == 3,'Season']='fall'
df.loc[df['season'] == 4,'Season']='winter'
#----------------------------------------
df.loc[df['weather'] == 1,'Weather']='Clear'
df.loc[df['weather'] == 2,'Weather']='Mist-Cloudy'
df.loc[df['weather'] == 3,'Weather']='Light Rain '
df.loc[df['weather'] == 4,'Weather']='Heavy Rain'
#----------------------------------------
df.loc[df['workingday'] == 1,'Day_Status']='Working'
df.loc[df['workingday'] == 0,'Day_Status']='Holiday\Weekend'
#----------------------------------------
df.loc[df['holiday'] == 1,'Holiday']='Yes'
df.loc[df['holiday'] == 0,'Holiday']='No'
```

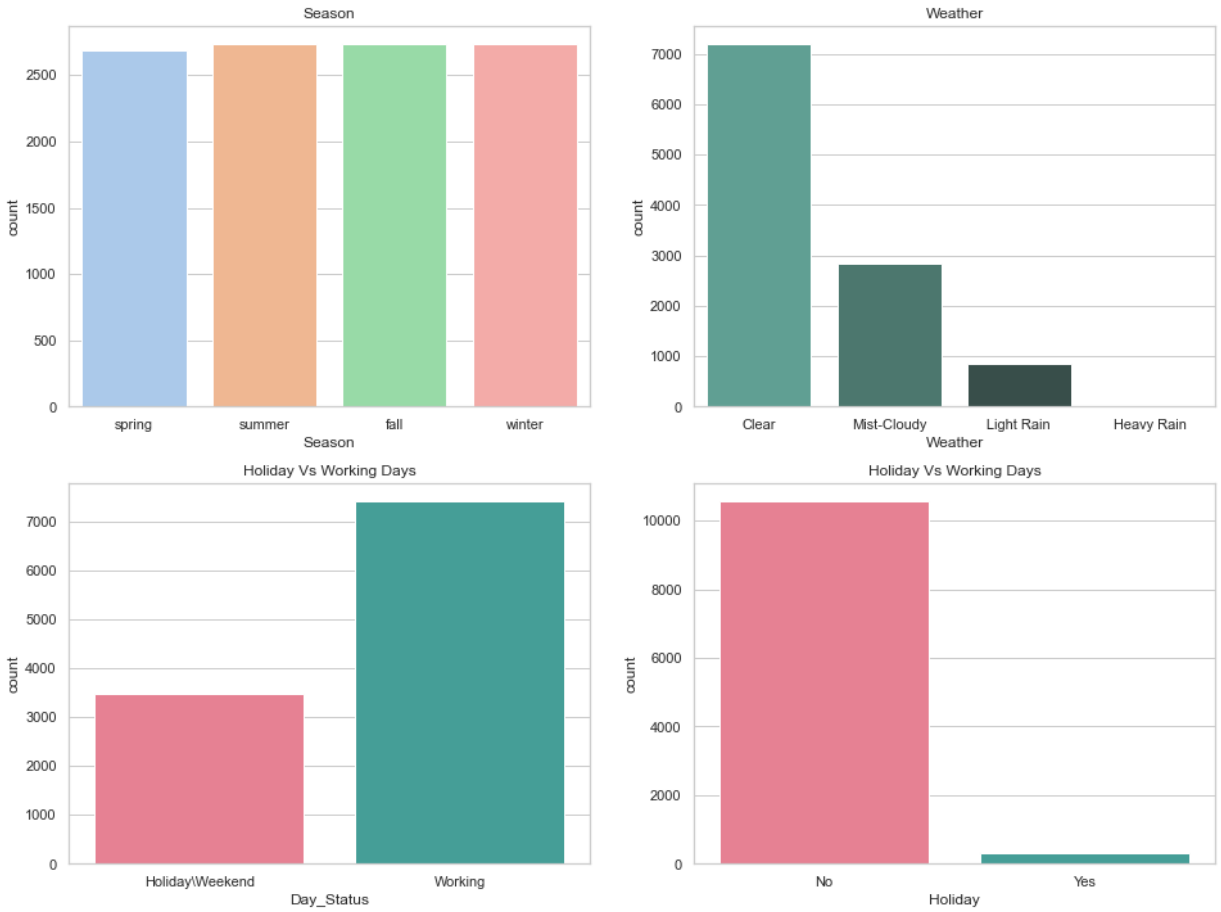
In [121]:
```
df.drop(columns = ['season','weather','workingday','holiday'],inplace=True)
```

In [122]:
```
df_time = df[['datetime','count','casual','registered']]
df_time['datetime'] = pd.to_datetime(df_time['datetime'])
df_time["day"]=df_time["datetime"].dt.day_name()
df_time["hour"] = df_time["datetime"].dt.hour
df_time["Month"] = df_time["datetime"].dt.month
df_time["year"] = df_time["datetime"].dt.year
```

Visual Analysis - Univariate & Bivariate

Univariate Analysis

In [123]:
```
fig, axs = plt.subplots(nrows=2, ncols=2, figsize=(16,12))
sns.countplot(data=df, x='Season', palette='pastel',ax=axs[0][0]).set(title='Seas
sns.countplot(data=df, x='Weather',palette='dark:#5A9_r',ax=axs[0][1]).set(title=
sns.countplot(data=df, x='Day_Status',palette='husl',ax=axs[1][0]).set(title='Ho
sns.countplot(data=df, x='Holiday',palette='husl',ax=axs[1][1]).set(title='Holida
plt.show()
```



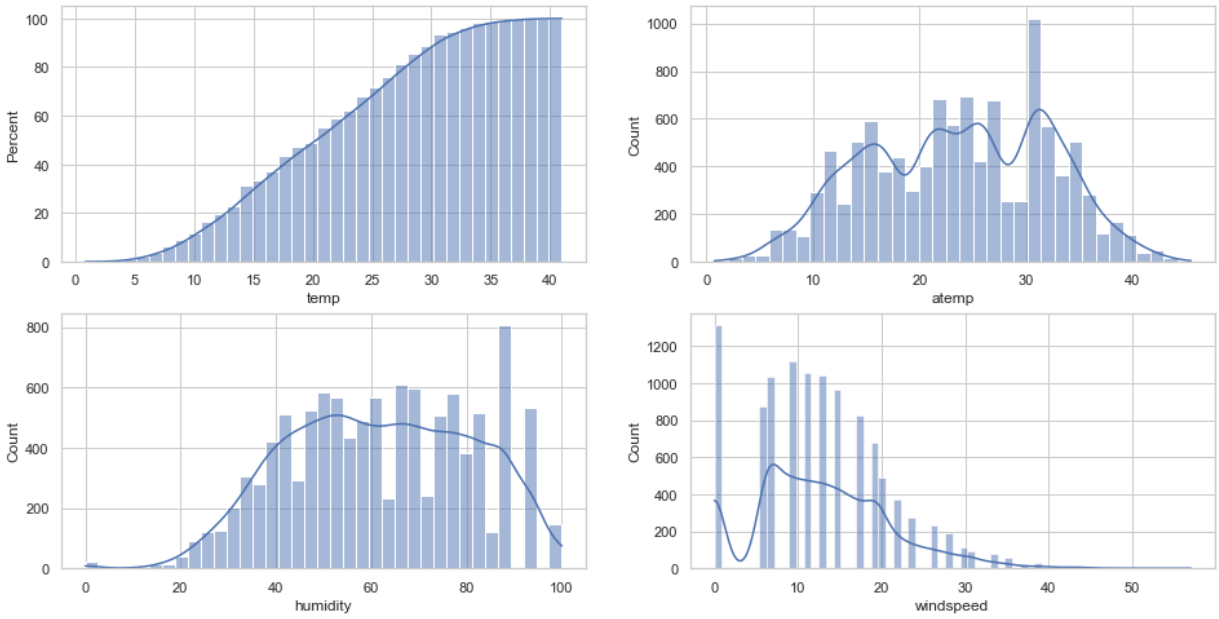
Observations

1. Seasonal data is almost same for all given seasons
2. Weather looks mostly clear followed by cloudy, light rain. Only one data point avaiable for high rain
3. As expected working days are more as compared to holidays

In [ ]:
```
# Analysis on numerical columns
```

In [124]:
```
fig, axs = plt.subplots(nrows=2, ncols=2, figsize=(16,8))
sns.histplot(data=df,x='temp',ax=axs[0][0], kde=True,cumulative = True, stat = '
sns.histplot(data=df,x='atemp',ax=axs[0][1],kde=True)
sns.histplot(data=df,x='humidity',ax=axs[1][0],kde=True)
sns.histplot(data=df,x='windspeed',ax=axs[1][1],kde=True)
```

Out[124]: <AxesSubplot:xlabel='windspeed', ylabel='Count'>



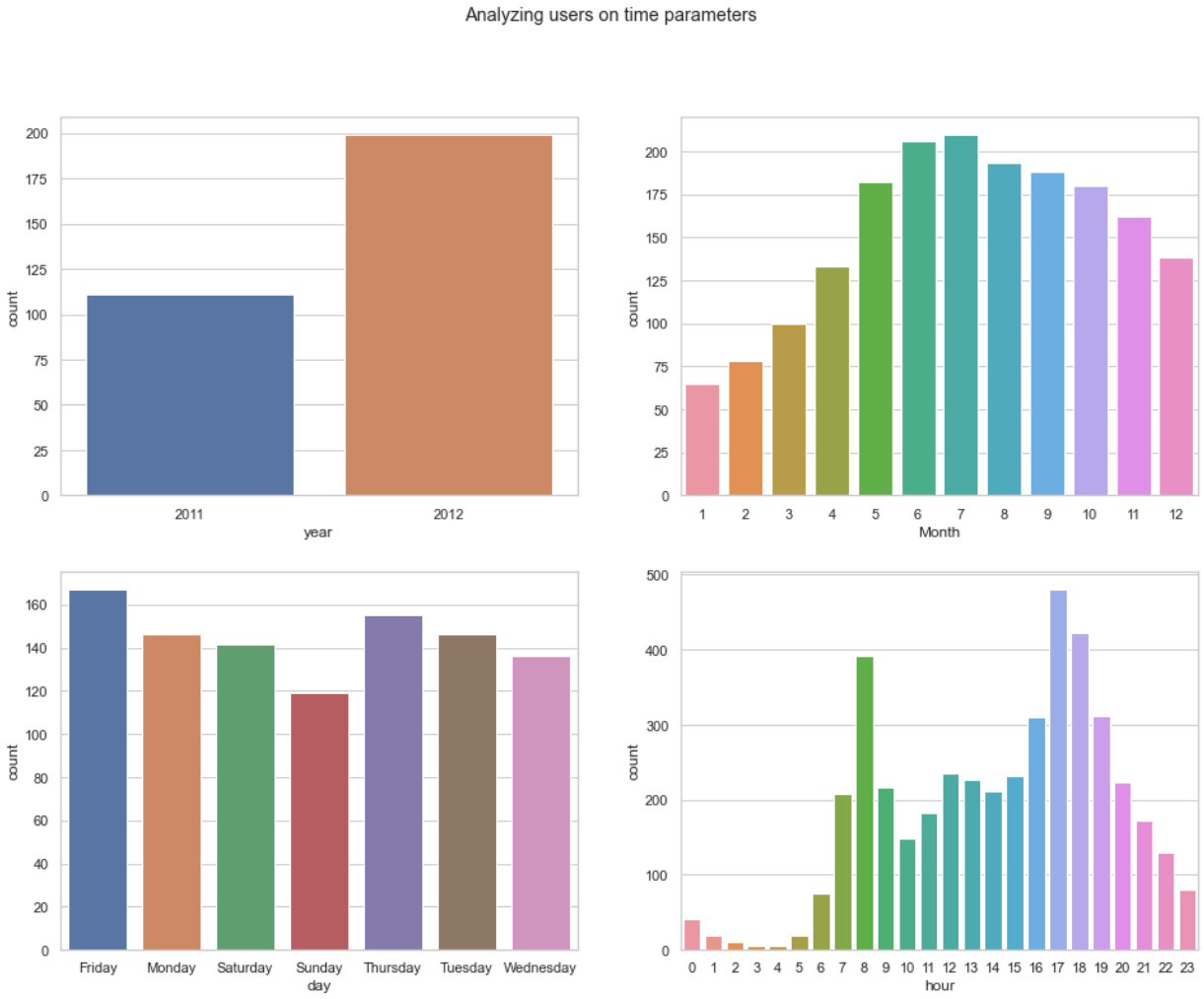
Observations related to graphical distributions

1. temp, atemp and humidity looks like they follows the Normal Distribution
2. windspeed follows the binomial distribution
3. As seen from above plot casual, registered and count are having right skewed plots
4. 80 % of the time, the temperature is less than 28 degrees celcius.s

In [125]:
```
# VISUALIZING TIMEING FEATURES
```

In [126]:
```
fig, axs = plt.subplots(nrows=2, ncols=2, figsize=(16,12))
plt.suptitle('Analyzing users on time parameters')
sns.barplot(y = df_time.groupby('year')['count'].median(), x = df_time.groupby('y
sns.barplot(y = df_time.groupby('Month')['count'].median(), x = df_time.groupby(
sns.barplot(y = df_time.groupby('day')['count'].median(), x = df_time.groupby('da
sns.barplot(y = df_time.groupby('hour')['count'].median(), x = df_time.groupby('h
```

Out[126]: <AxesSubplot:xlabel='hour', ylabel='count'>



Observations:

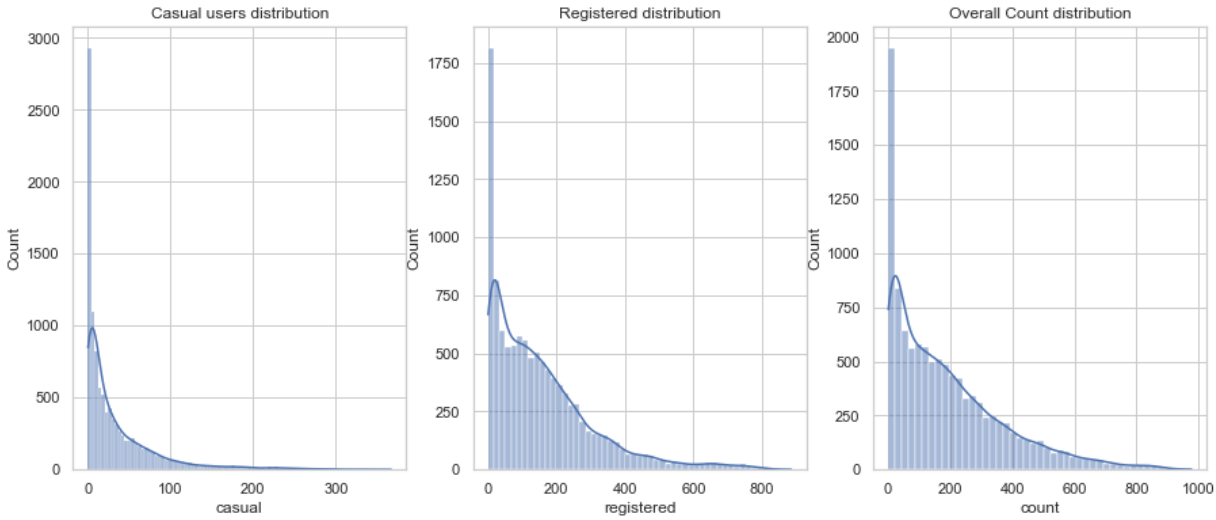
- In year 2012 number of user increased as compared to 2011.
- Median of number of cycles rented are higher during morning 7 to 9 am to evening 4 to 8pm .
- On friday majority of people are using cycles. Week off days (Saturday and Sunday) shows less count.
- Second part of year i.e after month of May, users are increased.

In [ ]:
```
# Analysis on Casual vs Registered customers
```

In [127]:
```
fig, axs = plt.subplots(nrows=1, ncols=3, figsize=(15,6))

sns.histplot(data=df,x='casual',ax=axs[0],kde=True).set_title("Casual users distr
sns.histplot(data=df,x='registered',ax=axs[1],kde=True).set_title("Registered dis
sns.histplot(data=df,x='count',ax=axs[2],kde=True).set_title("Overall Count distr
```

Out[127]: Text(0.5, 1.0, 'Overall Count distribution')

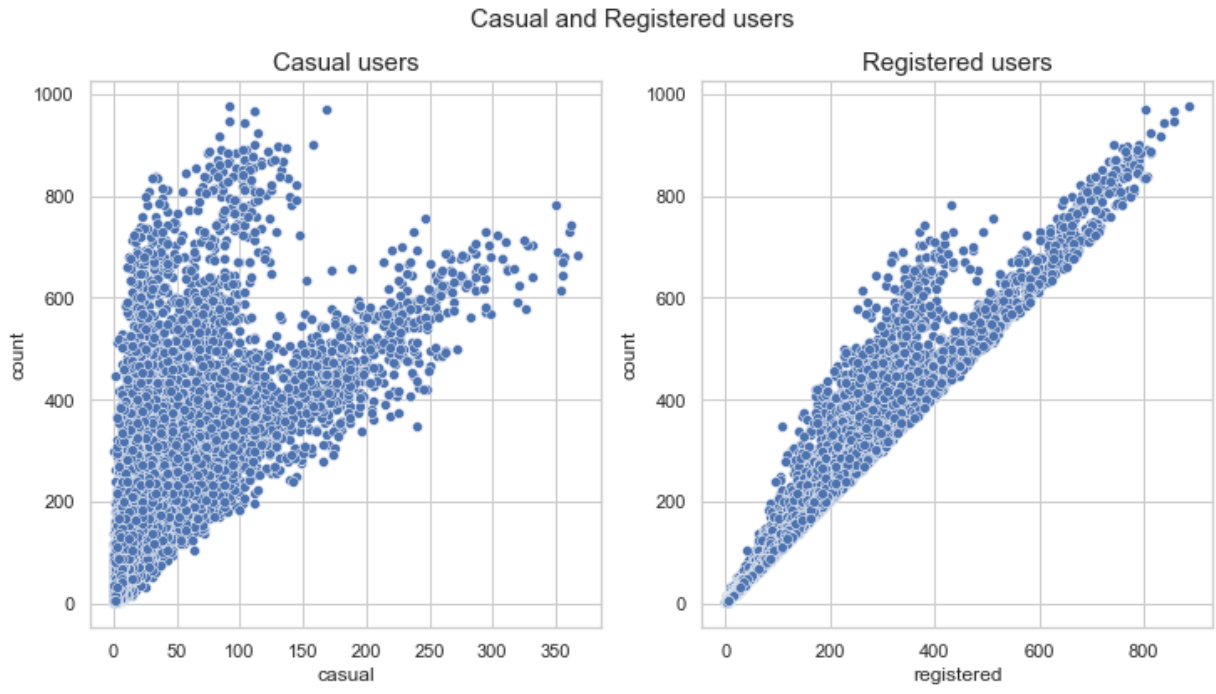


In [128]:
```
plt.figure(figsize=(12,6))
plt.suptitle('Casual and Registered users', fontsize=15)

plt.subplot(1,2,1)
sns.scatterplot(x=df['casual'], y=df['count'],data=df)
plt.title('Casual users', fontsize=15)

plt.subplot(1,2,2)
sns.scatterplot(x=df['registered'], y=df['count'],data=df)
plt.title('Registered users', fontsize=15)
plt.plot()
```

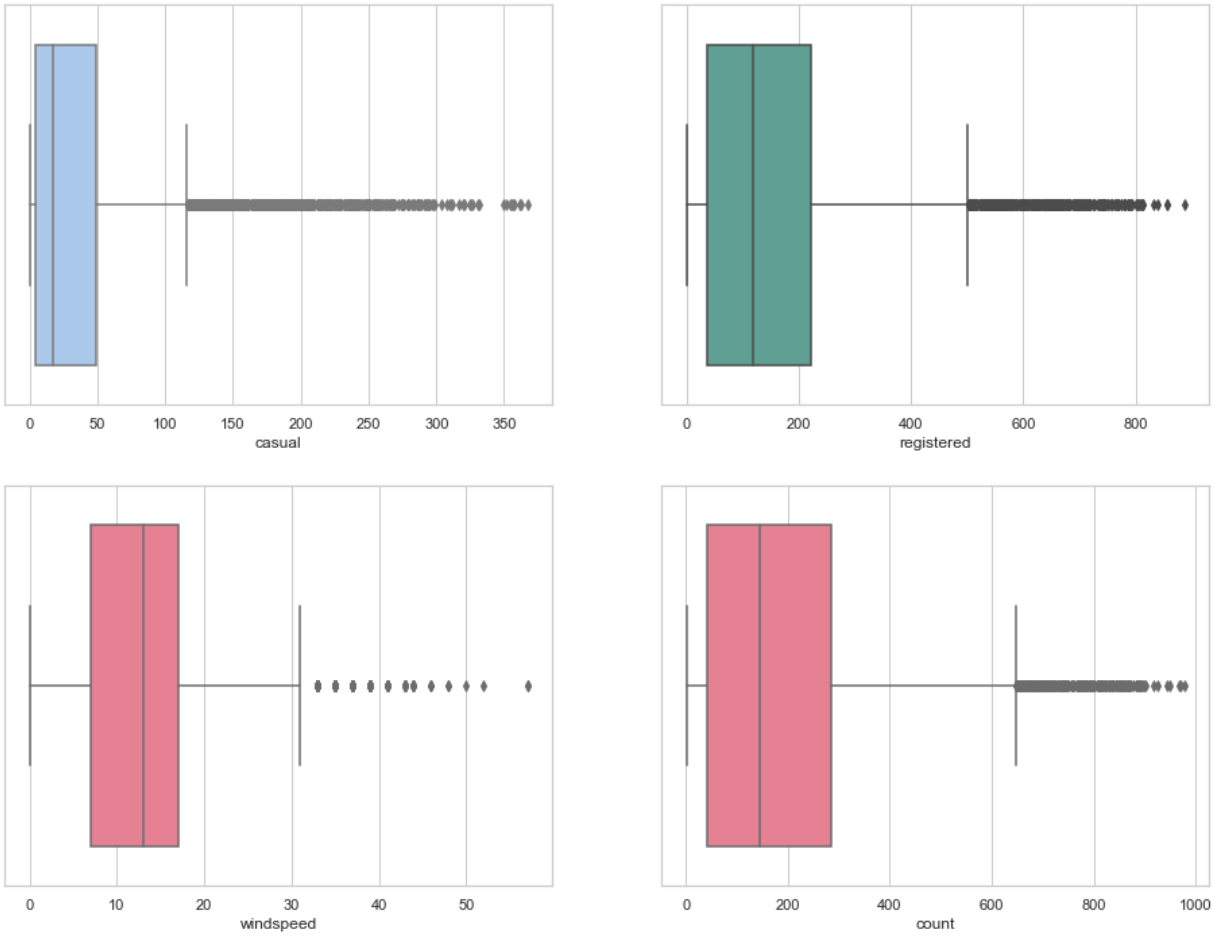
Out[128]: []



Registered cycle counts seems to be much higher than the casual customers.

In [129]:
```
# Detecting outliers using box plot
```

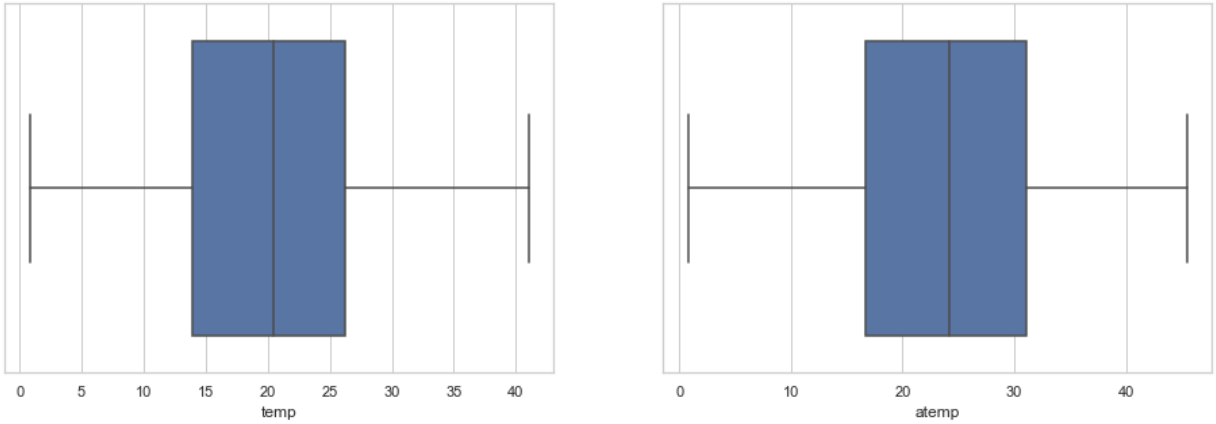
In [130]:
```
fig, axs = plt.subplots(nrows=2, ncols=2, figsize=(16,12))
sns.boxplot(data=df, x='casual', palette='pastel',ax=axs[0][0])
sns.boxplot(data=df, x='registered',palette='dark:#5A9_r',ax=axs[0][1])
sns.boxplot(data=df, x='windspeed',palette='husl',ax=axs[1][0])
sns.boxplot(data=df, x='count',palette='husl',ax=axs[1][1])
plt.show()
```



In [131]:
```
plt.figure(figsize=(16,5))
plt.subplot(121)
sns.boxplot(df['temp']);

plt.subplot(122)
sns.boxplot(df['atemp'])

plt.show()
```



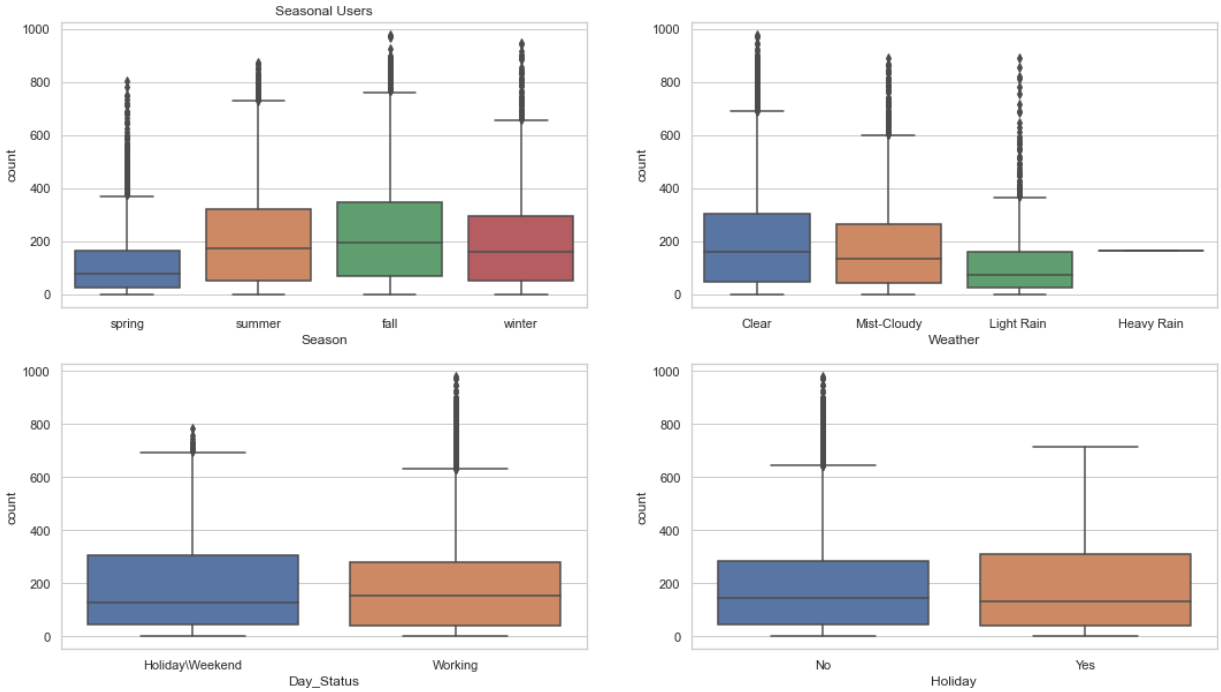
Observations Related to box plots outliers

1. As seen from above plot there are outliers present in casual, registered, windspeed and count features
2. There are outliers present in temp and atemp columns.

Bivariate Analysis

In [132]:
```
fig, axs = plt.subplots(nrows=2, ncols=2, figsize=(18, 10))
sns.boxplot(data=df,x='Season', y='count',ax=axs[0,0]).set_title('Seasonal Users
sns.boxplot(data=df,x='Weather', y='count',ax=axs[0,1])
sns.boxplot(data=df,x='Day_Status', y='count',ax=axs[1,0])
sns.boxplot(data=df,x='Holiday', y='count',ax=axs[1,1])
```

Out[132]: <AxesSubplot:xlabel='Holiday', ylabel='count'>

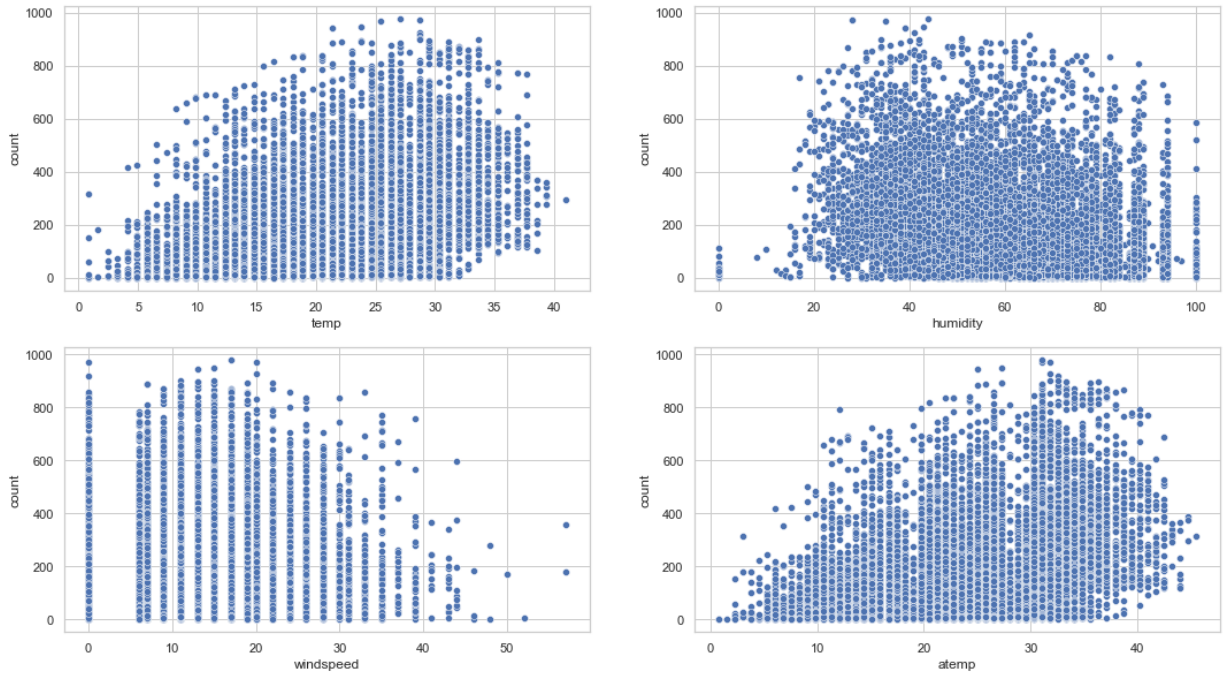


Observations Related to box plots outliers

1. In summer and fall season more bikes are rented followed by winter and spring
2. Clear wheather attracts more people to take bike than cloudy and rainy days.
3. On weekends and holidays slightly more bikes are rented than that of working days.
4. On holidays more bikes are rented.

In [133]:
```
fig, axs = plt.subplots(nrows=2, ncols=2, figsize=(18, 10))
sns.scatterplot(data=df,x='temp', y='count',ax=axs[0][0])
sns.scatterplot(data=df,x='humidity', y='count',ax=axs[0][1])
sns.scatterplot(data=df,x='windspeed', y='count',ax=axs[1][0])
sns.scatterplot(data=df,x='atemp', y='count',ax=axs[1][1])
```

Out[133]: <AxesSubplot:xlabel='atemp', ylabel='count'>

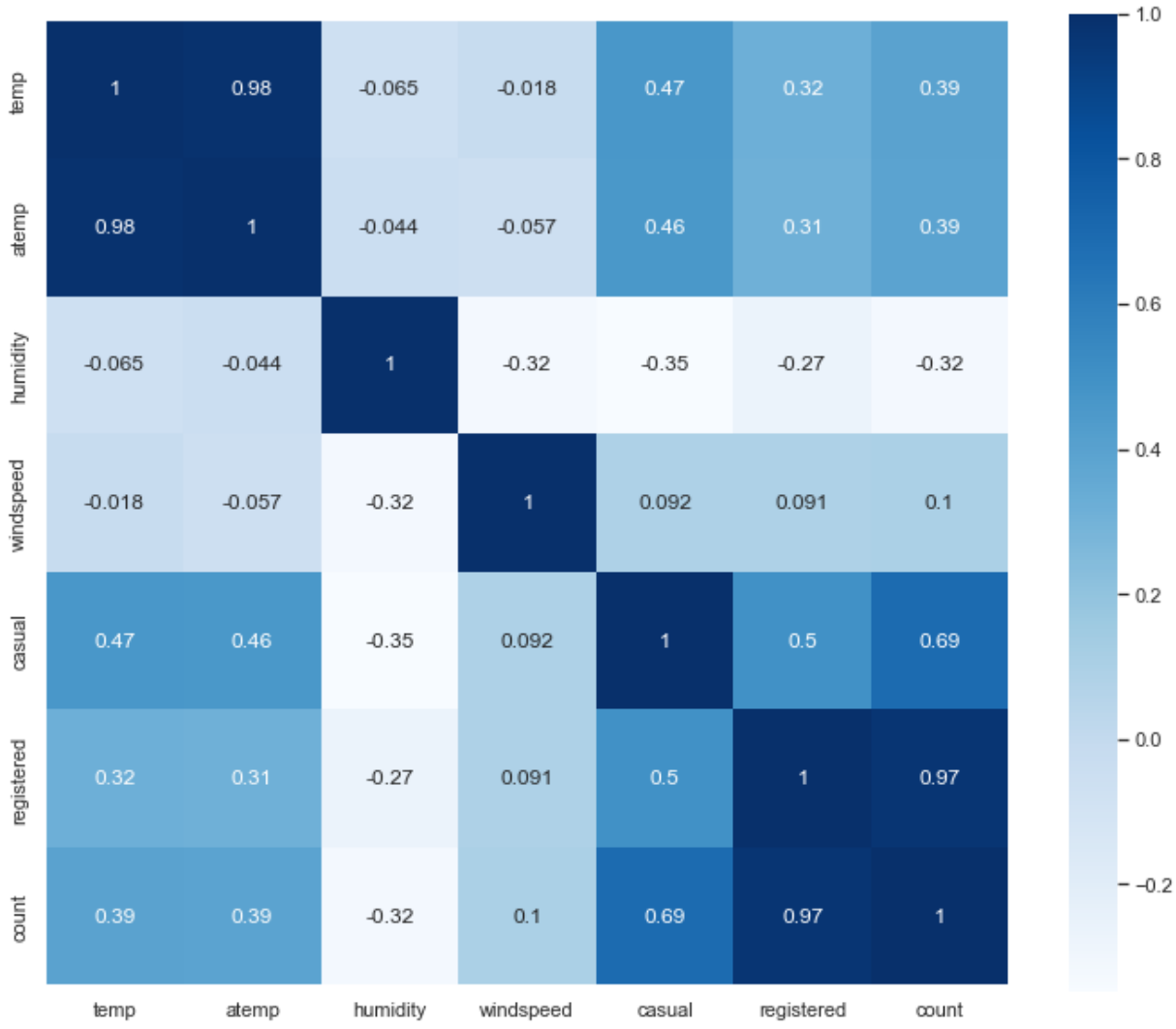


Observations from scatterplots

- Customers are low when humidity is below 20
- When windspeed is more than 30 bikes rented are less
- If temperature lies between 15 to 35 degree more bykes are rented

In [134]:
```
plt.figure(figsize=(12,10))
sns.heatmap(df.corr(method="pearson"),square=True,annot=True,cmap="Blues")
```

Out[134]: <AxesSubplot:>



Observations from correlation matrix

- There is high correlation between temp and atemp featues: 0.98
- Humidity is negatively correlated with all coustomers: -0.32
- Registered user have high correlation than casual users.
- Windspeed doen't seem to have large effect on number of cycles rented for all customers

Hypothesis Testing

Weather & Season dependency test

As we are comparing two categorical variable Chiquared test is selected

Null Hypothesis H0: Weather and Season are independet.

Alternate Hypothesis Ha: Weather and Season are dependent

In [135]:
```
weather_season = pd.crosstab(index=df['Season'],columns=df['Weather'])
weather_season
```

Out[135]:

Season	Weather	Clear	Heavy Rain	Light Rain	Mist-Cloudy
fall		1930	0	199	604
spring		1759	1	211	715
summer		1801	0	224	708
winter		1702	0	225	807

In [136]:
```
chi_stat, p_value, dof, expeced_Values =chi2_contingency(weather_season)
alpha = 0.05
print('Chi Statistics: ',chi_stat)
print('P Value: ',p_value)
print('Degree of freedom: ',dof)
print()
print("Expected value matrix: ",expeced_Values)
print('*'*50)
if p_value < alpha:
    print('As p values is less than aplha we reject Null hypothesis')
    print('Weather and Season are dependent.')
else:
    print('As p values is more than aplha we accept Null hypothesis')
    print('Weather and Season are independent.')
```

```
Chi Statistics:  49.158655596893624
P Value:  1.549925073686492e-07
Degree of freedom:  9

Expected value matrix:  [[1.80559765e+03 2.51056403e-01 2.15657450e+02 7.114938
45e+02]
 [1.77454639e+03 2.46738931e-01 2.11948742e+02 6.99258130e+02]
 [1.80559765e+03 2.51056403e-01 2.15657450e+02 7.11493845e+02]
 [1.80625831e+03 2.51148264e-01 2.15736359e+02 7.11754180e+02]]
**************************************************
As p values is less than aplha we reject Null hypothesis
Weather and Season are dependent.
```

In [ ]:
```
```

Checking working Day has an effect on the number of electric cycles rented or not

As we are comparing categorical variable and count of cycles we are selecting 2 sample T test.

Null Hypothesis H0: Working day has effect on the number of cycles being rented.

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

In [169]:
```
holiday     = df.loc[df['Day_Status']=='Holiday\\Weekend']['count']
working_day = df.loc[df['Day_Status']=='Working']['count']
```

In [170]:
```
t_stat, p_t_value = ttest_ind(a=holiday, b=working_day)
```

In [171]:
```
alpha = 0.05
print('T Statistics: ',t_stat)
print('P Value: ',p_t_value)
print()
print('*'*50)
if p_t_value < alpha:
    print('As p values is less than alpha we reject Null hypothesis')
    print('Working day has effect on the number of cycles being rented.')
else:
    print('As p values is more than alpha we accept Null hypothesis')
    print('Working day has no effect on the number of cycles being rented.')
```

```
T Statistics:  -1.2096277376026694
P Value:  0.22644804226361348

**************************************************
As p values is more than alpha we accept Null hypothesis
Working day has no effect on the number of cycles being rented.
```

In [ ]:
```

```

Season has an effect on the number of cycles rented or not

We will initially perform Annova test and if assumption of anova fails we will check Kruskal Test

Null Hypothesis H0: Number of cycles rented is similar in different season.

Alternate Hypothesis Ha: Number of cycles rented is not similar in different season.

In [140]:
```
# Seperating season and weather with respect to values
S1 = df.loc[df['Season'] == 'winter']['count']
S2 = df.loc[df['Season'] == 'summer']['count']
S3 = df.loc[df['Season'] == 'fall']['count']
S4 = df.loc[df['Season'] == 'spring']['count']
```

In [141]:

```
f_ratio, p_anova_season_value = f_oneway(S1,S2,S3,S4)

alpha = 0.05
print('F_ratio: ',f_ratio)
print('P Value: ',p_anova_season_value)
print()
print('*'*50)
if p_anova_season_value < alpha:
    print('As p values is less than alpha we reject Null hypothesis')
    print('Number of cycles rented is not similar in different weather and seasor
else:
    print('As p values is more than alpha we accept Null hypothesis')
    print('Number of cycles rented is similar in different weather and season')
```

```
F_ratio:  236.94671081032104
P Value:  6.164843386499654e-149

**************************************************
As p values is less than alpha we reject Null hypothesis
Number of cycles rented is not similar in different weather and season
```

Checking if data is normally distributed

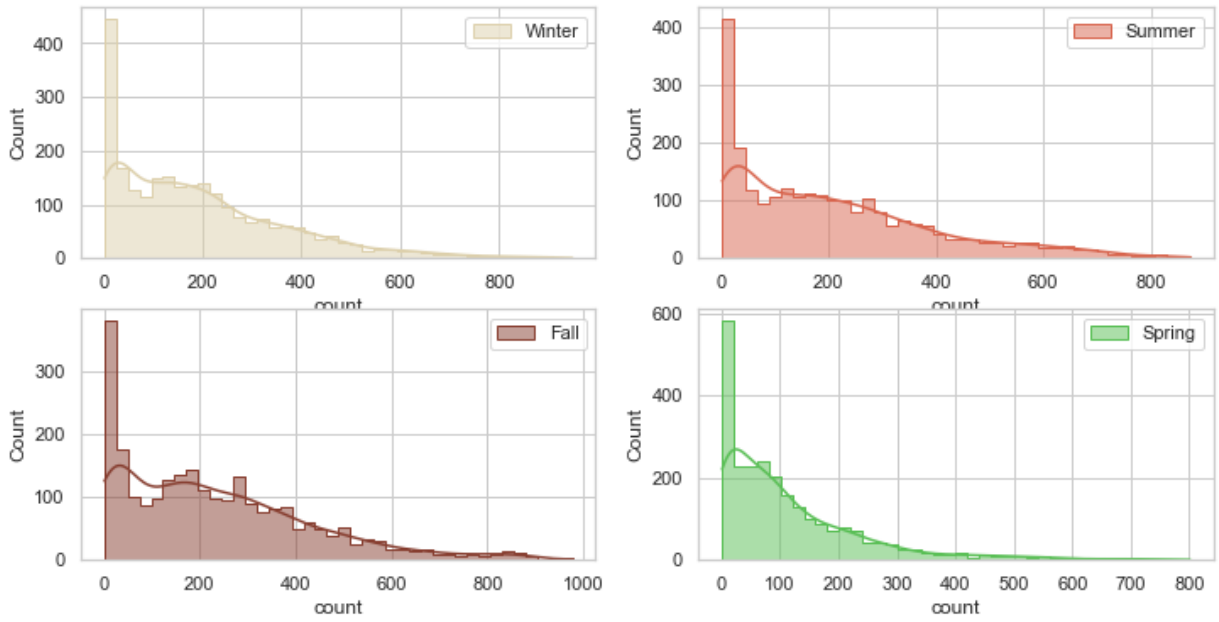
In [142]:
```
plt.figure(figsize = (12, 6))
plt.subplot(2, 2, 1)
sns.histplot(S1.sample(2500), bins = 40, element = 'step', color = '#DDD0AC', kde
plt.legend()


plt.subplot(2, 2, 2)
sns.histplot(S2.sample(2500), bins = 40, element = 'step', color = '#D8654E', kde
plt.legend()


plt.subplot(2, 2, 3)
sns.histplot(S3.sample(2500), bins = 40, element = 'step', color = '#873E30', kde
plt.legend()


plt.subplot(2, 2, 4)
sns.histplot(S4.sample(2500), bins = 40, element = 'step', color = '#59BF55', kde
plt.legend()
plt.plot()
```

Out[142]: []



Distribution of Season do not follow normal distribution

In [143]:

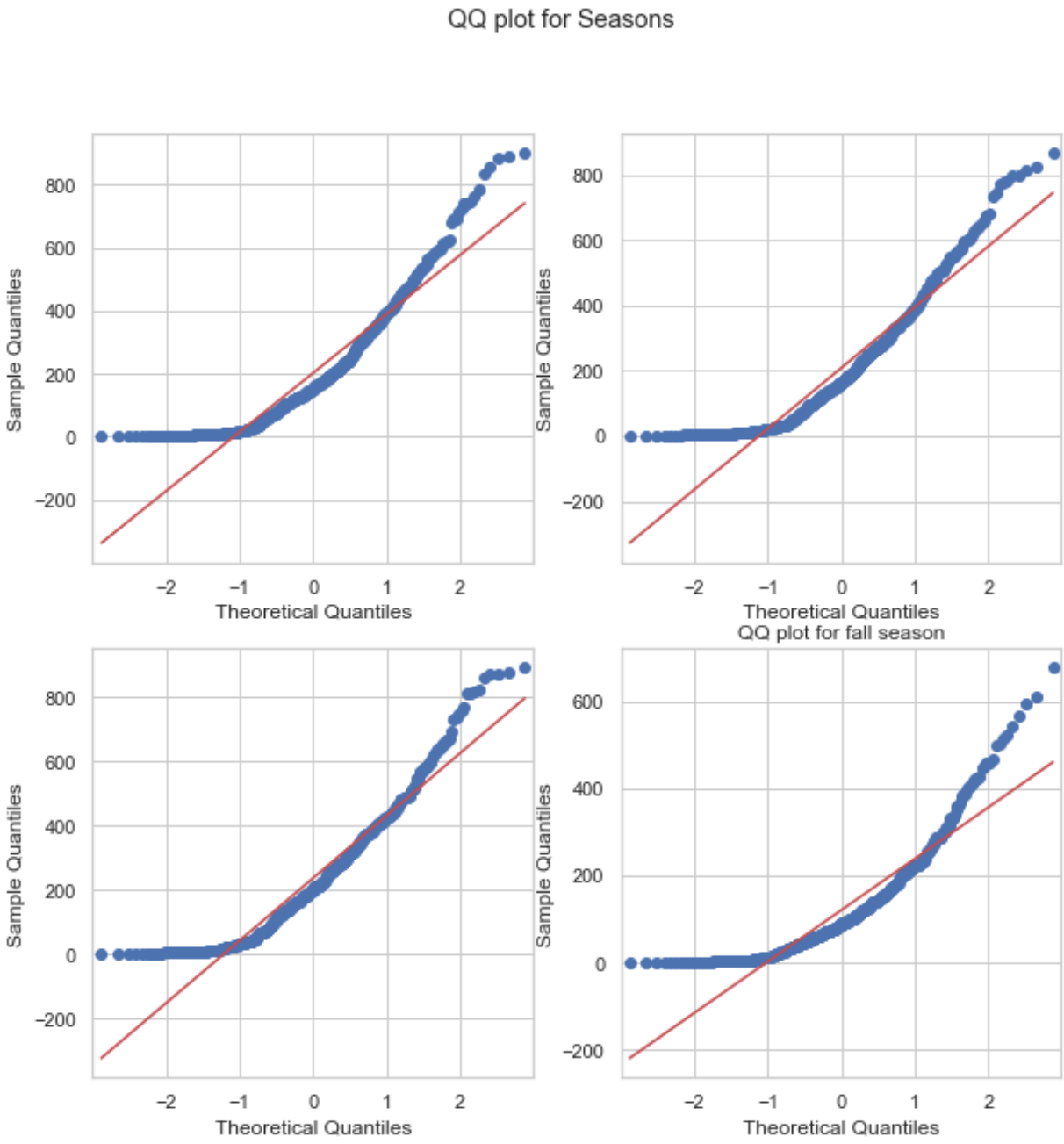
```
fig, axs = plt.subplots(nrows=2, ncols=2, figsize=(10,10))
plt.suptitle('QQ plot for Seasons')

qqplot(S1.sample(500), line="s",ax=axs[0][0])
plt.title('QQ plot for winter season')

qqplot(S2.sample(500), line="s",ax=axs[0][1])
plt.title('QQ plot for summer season')

qqplot(S3.sample(500), line="s",ax=axs[1][0])
plt.title('QQ plot for fall season')

qqplot(S4.sample(500), line="s",ax=axs[1][1])
plt.show()
```



All 4 QQ plots shows that the samples do not follow normal distribution

To check normality of data we will perform Shapiro-Wilk test

In [144]:
```
S1_subset= S1.sample(500)
test_stat, p_value = shapiro(S1_subset)
print(p_value)
if p_value < 0.05:
    print('The Winter season sample is not normally distributed')
else:
    print('The Winter season sample follows normal distribution')
```

```
8.254610074571065e-19
The Winter season sample is not normally distributed
```

In [145]:
```
S2_subset= S2.sample(500)
test_stat, p_value = shapiro(S2_subset)
print(p_value)
if p_value < 0.05:
    print('The Summer season sample is not normally distributed')
else:
    print('The Summer season sample follows normal distribution')
```

```
1.307031704810219e-18
The Summer season sample is not normally distributed
```

In [146]:
```
S3_subset= S3.sample(500)
test_stat, p_value = shapiro(S3_subset)
print(p_value)
if p_value < 0.05:
    print('The Fall season sample is not normally distributed')
else:
    print('The Fall season sample follows normal distribution')
```

```
2.540009610332291e-16
The Fall season sample is not normally distributed
```

In [147]:
```
S4_subset= S4.sample(500)
test_stat, p_value = shapiro(S4_subset)
print(p_value)
if p_value < 0.05:
    print('The Spring season sample is not normally distributed')
else:
    print('The Spring season sample follows normal distribution')
```

```
1.4613718951156351e-22
The Spring season sample is not normally distributed
```

3rd assumption verfication (Variance) : levene test

In [148]:
```
levene_stat, p_value_seasonal = levene(S1.sample(2500), S2.sample(2500),S3.sample
print(levene_stat, p_value_seasonal)
if p_value_seasonal < 0.05:
    print("Variances of all weather are not equal")
else:
    print('Variances of all weather are equal')
```

```
174.7693978635108 1.9267209295709706e-110
Variances of all weather are not equal
```

As Seasonal data is not normally distributed and do not have same variance we cannot perform f_oneway test. We will check f ratio and p value using Kruskal-Wallis test

In [149]:
```
t_stat, p_kruskal_season_Value = kruskal(S1,S2,S3,S4)

alpha = 0.05
print('Test Statistic =', t_stat)
print('p value =', p_kruskal_season_Value)
print()
print('*'*50)
if p_kruskal_season_Value < alpha:
    print('As p values is less than alpha we reject Null hypothesis')
    print('Number of cycles rented is not same in different seasons')
else:
    print('As p values is more than alpha we accept Null hypothesis')
    print('Number of cycles rented is same in different Sasons')
```

```
Test Statistic = 699.6668548181915
p value = 2.4790083726176776e-151

**************************************************
As p values is less than alpha we reject Null hypothesis
Number of cycles rented is not same in different seasons
```

This concludes Season has positive impact on number of cycles rented

In [ ]:
```
```

Similarly we can check for Weather has an effect on the number of cycles rented

We will initially perform Annova test and if assumption of anova fails we will check Kruskal Test

Null Hypothesis H0: Number of cycles rented is similar in different weather.

Alternate Hypothesis Ha: Number of cycles rented is not similar in different weather.

In [150]:
```
W1 = df.loc[df['Weather'] == 'Clear']['count']
W2 = df.loc[df['Weather'] == 'Mist-Cloudy']['count']
W3 = df.loc[df['Weather'] == 'Light Rain ']['count']
W4 = df.loc[df['Weather'] == 'Heavy Rain']['count']
print(len(W1),len(W2),len(W3),len(W4))
```

```
7192 2834 859 1
```

Skipping **'Heavy rain'** data point as there in only 1 data point and we cannot perform a ANOVA test with a single data point.

In [151]:
```
f_ratio, p_anova_weather = f_oneway(W1,W2,W3)
```

In [152]:
```
alpha = 0.05
print('F_ratio: ',f_ratio)
print('P Value: ',p_anova_weather)
print()
print('*'*50)
if p_anova_weather < alpha:
    print('As p values is less than alpha we reject Null hypothesis')
    print('Number of cycles rented is not similar in different weather')
else:
    print('As p values is more than alpha we accept Null hypothesis')
    print('Number of cycles rented is similar in different weather')
```

```
F_ratio:  98.28356881946706
P Value:  4.976448509904196e-43

**************************************************
As p values is less than alpha we reject Null hypothesis
Number of cycles rented is not similar in different weather
```

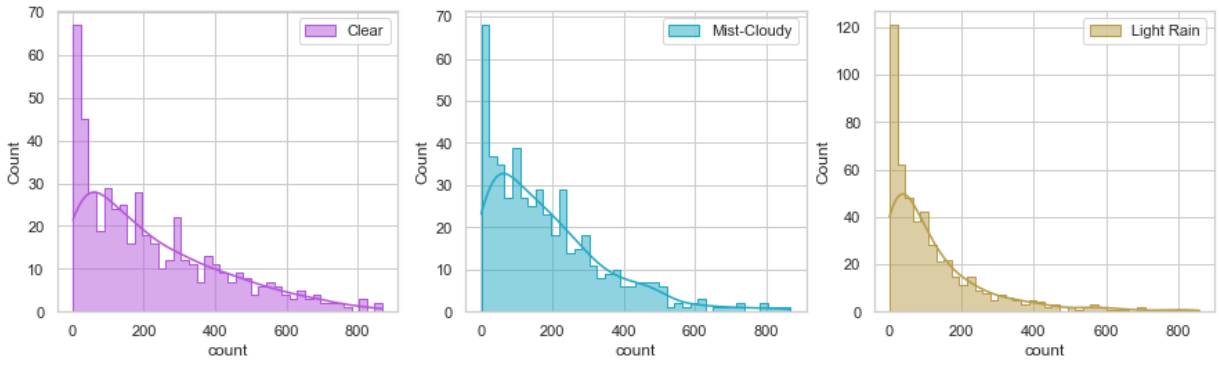
Checking if data is normally distributed

In [153]:
```
plt.figure(figsize = (15, 4))
plt.subplot(1, 3, 1)
sns.histplot(W1.sample(500), bins = 40, element = 'step', color = '#B357D8', kde
plt.legend()

plt.subplot(1, 3, 2)
sns.histplot(W2.sample(500), bins = 40, element = 'step', color = '#21A9C4', kde
plt.legend()

plt.subplot(1, 3, 3)
sns.histplot(W3.sample(500), bins = 40, element = 'step', color = '#B89C4A', kde
plt.legend()
plt.plot()
```

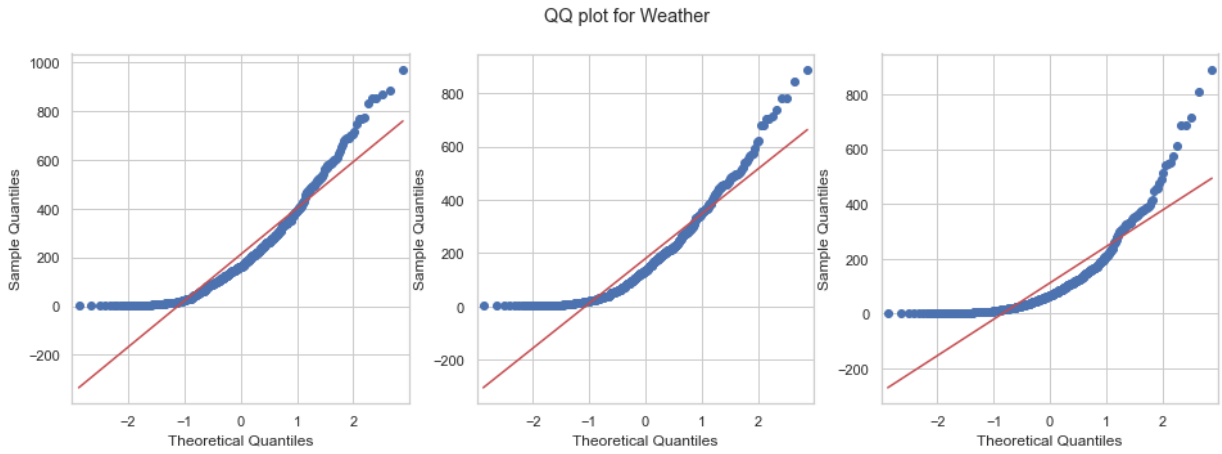
Out[153]: []



Distribution of Weather do not follow normal distribution

In [154]:
```
fig, axs = plt.subplots(nrows=1, ncols=3, figsize=(16,5))
plt.suptitle('QQ plot for Weather')

qqplot(W1.sample(500), line="s",ax=axs[0])
qqplot(W2.sample(500), line="s",ax=axs[1])
qqplot(W3.sample(500), line="s",ax=axs[2])
plt.show()
```



All 3 QQ plots shows that the samples do not follow normal distribution

To check normality of data we will perform Shapiro-Wilk test

In [155]:
```
W1_subset= W1.sample(500)
test_stat, p_value = shapiro(W1_subset)
print(p_value)
if p_value < 0.05:
    print('The CLEAR weather sample is not normally distributed')
else:
    print('The CLEAR weather sample follows normal distribution')
```

```
6.8239592257445166e-18
The CLEAR weather sample is not normally distributed
```

In [156]:
```
W2_subset= W2.sample(500)
test_stat, p_value = shapiro(W2_subset)
print(p_value)

if p_value < 0.05:
    print('The MIST cloud weather sample is not normally distributed')
else:
    print('The MIST cloud weather sample is normally distributed')
```

```
2.220534850449492e-21
The MIST cloud weather sample is not normally distributed
```

In [157]:
```
W3_subset= W3.sample(500)
test_stat, p_value = shapiro(W3_subset)
print(p_value)

if p_value < 0.05:
    print('The Light rain weather sample is not normally distributed')
else:
    print('The Light rain weather sample is normally distributed')
```

```
5.573571001132289e-26
The Light rain weather sample is not normally distributed
```

3rd assumption verfication (Variance) : levene test

In [158]:
```
levene_stat, p_value = levene(W1.sample(500), W2.sample(500),W3.sample(500))
print(levene_stat, p_value)
if p_value < 0.05:
    print("Variances of all weather are not equal")
else:
    print('Variances of all weather are equal')
```

```
19.675777910620162 3.6753613976913414e-09
Variances of all weather are not equal
```

As the data of weather is not normally distributed and do not have same variance we cannot perform f_oneway test. We will check f ratio and p value using Kruskal-Wallis test

In [159]:
```
test_stat, p_value_kruskal_weather = kruskal(W1, W2, W3)

alpha = 0.05
print('T_Statistic =', test_stat)
print('p value =', p_value_kruskal_weather)
print()
print('*'*50)
if p_value_kruskal_weather < alpha:
    print('As p values is less than alpha we reject Null hypothesis')
    print('Number of cycles rented is not same in different weather')
else:
    print('As p values is more than alpha we accept Null hypothesis')
    print('Number of cycles rented is same in different weather')
```

```
T_Statistic = 204.95566833068537
p value = 3.122066178659941e-45

**************************************************
As p values is less than alpha we reject Null hypothesis
Number of cycles rented is not same in different weather
```

This concludes Weather has positive impact on number of cycles rented

Insights

- The time period of given data is from 1st January 2011 to 19th December 2012 i.e 718 days.
- In 2012 demand is increased as compared to 2011.
- Registered users are more than casual users.
- In spring and summer months that is spring (March, April, May), summer (June, July, August) demand increases.
- Clear and less cloudy Weather has more count of total rental bikes, followed by the misty weather and rainy weather.
- While demand is slightly decreasing in fall and winter seasons.
- Less data has been given for high rain situdations. As this cycles are two wheelers it can be understood people will not choose this option in rainy seasons.
- Inconsistency can be seen in hourly demand throughout the day.
- Cycles are getting more use during 7-9 AM and 4-8 PM which are office hours.
- Features atemp and temp are highly correalted.

- More than 80 % of the time, the temperature is less than 28 degrees celcius.
- Windspeed increases , the number of bike rented are decreases.

Hypothesis Testing Results

1. As we performed chi square test on weather and season columns we found this two features are dependent on each other.
2. Mean of number of cycles rented on working days are equal as the cycles rented on offdays.
3. It is concluded from statistical tests, that demand on weekdays and off-days are similar
4. Using anova test we concluded that the number of cycles rented across different weather and season are not same.

In [ ]:

Recommendations

1. It has been seen from insights there is hourly pattern which affects count of rental bikes, company can adjust price rating accordingly. During office or rush hours it can offer less prices or discounts to attract more customers.

2. Impact of weather on bike rentals can be understood as people avoid taking 2 wheelers during rainy seasons.Yulu can offer weather-specific discounts to attract more customers during these favorable weather conditions.

3. To convert casual users to registered company can offer free trails or offer on credit cards discounts. To avoid retention rating increase Yulu can offer exclusive offers, or personalized recommendations for registered users.

4. During rush hours and comfortable weathers ensure having sufficient bikes available to meet the higher demand.
5. More data should be collected for exterme whether conditions to understand customer behavior. There can be situation where people prefer 2 wheelrs rather than 4 wheelers to avoid traffic jams once wheather clears. Here Yulu bikes can considered as options by people.

6. For shorter distance travels company can offer less pricing so that instead of private two wheelers people will prefer these bikes. Company can target bike stations at metro, railway stations so that people can take bikes from this stations to reach destinations such as company quarters or home.

7. As Yulu promotes less polution statergy they can offer special schemes or discounts on enviormental day and earh day. This can target future customers also.

8. Social media platforms can promote the electric bikes and target customers through interactive posts. Advertising campaigns to reach specific customer bookings.

9. As demand on weekend is also high, Yulu can target evening time as people go out for dinner outings or nearby locations.

10. Company can have tie ups with corporate businesses as well as nowdays companies are promoting enviormental friendly travels with addtional employee benefits.