Yulu - Hypothesis Testing.

About Yulu

Yulu is India's leading micro-mobility service provider, which offers unique vehicles for the daily commute. Starting off as a mission to eliminate traffic congestion in India, Yulu provides the safest commute solution through a user-friendly mobile app to enable shared, solo and sustainable commuting.

Yulu zones are located at all the appropriate locations (including metro stations, bus stands, office spaces, residential areas, corporate offices, etc) to make those first and last miles smooth, affordable, and convenient!

Yulu has recently suffered considerable dips in its revenues. They have contracted a consulting company to understand the factors on which the demand for these shared electric cycles depends. Specifically, they want to understand the factors affecting the demand for these shared electric cycles in the Indian market.

```
# from google.colab import drive
# drive.mount('/content/drive')
```

The company wants to know:

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

Library Imports

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

import numpy as np
from scipy.stats import norm
from scipy.stats import ttest_1samp, t,ttest_ind
from scipy.stats import chi2_contingency, chi2,chisquare
from scipy.stats import f, f_oneway, shapiro, ttest_ind, levene, kruskal, probplot



season_val = ["spring","summer","fall","winter"]
weather_val = ["Clear","Mist","Light Snow/Rain","Heavy Rain"]

def season_name(df):
  if df["season"] == 1:
    return "spring"
  elif df["season"] == 2:
    return "summer"
  elif df["season"] == 3:
    return "fall"
  else:
    return "winter"

def wthr_name(df):
  if df["weather"] == 1:
    return "Clear"
  elif df["weather"] == 2:
    return "Mist"
  elif df["weather"] == 3:
    return "Light Snow/Rain"
  else:
    return "Heavy Rain"
```

Import the dataset.

```
df = pd.read_csv("/content/drive/MyDrive/Scaler_Assignments/yulu.csv")
df.head()
```

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

```
df.describe()
```

	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registe
count	10886.000000	10886.000000	10886.000000	10886.000000	10886.00000	10886.000000	10886.000000	10886.000000	10886.000000	10886.000
mean	2.506614	0.028569	0.680875	1.418427	20.23086	23.655084	61.886460	12.799395	36.021955	155.552
std	1.116174	0.166599	0.466159	0.633839	7.79159	8.474601	19.245033	8.164537	49.960477	151.039
min	1.000000	0.000000	0.000000	1.000000	0.82000	0.760000	0.000000	0.000000	0.000000	0.000
25%	2.000000	0.000000	0.000000	1.000000	13.94000	16.665000	47.000000	7.001500	4.000000	36.000
50%	3.000000	0.000000	1.000000	1.000000	20.50000	24.240000	62.000000	12.998000	17.000000	118.000
75%	4.000000	0.000000	1.000000	2.000000	26.24000	31.060000	77.000000	16.997900	49.000000	222.000
max	4.000000	1.000000	1.000000	4.000000	41.00000	45.455000	100.000000	56.996900	367.000000	886.000

```
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
```

```
df.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
```

Looking at this table, we can figure out that there are NO NULL values, so null value treatement would not be needed in this dataset. We are also able to observe that the date values are stored in Object format, so let's change the same to date format.

```
df.loc[:,["datetime"]] = pd.to_datetime(df["datetime"])
```

```
<ipython-input-11-8a23e6fc1015>:1: DeprecationWarning: In a future version, `df.iloc[:, i] = newvals` will attempt to set the values inp
  df.loc[:,["datetime"]] = pd.to_datetime(df["datetime"])
```

Looking at the dataset to understand the columns and the layout of the elements in the same.

```
df.head()
```

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

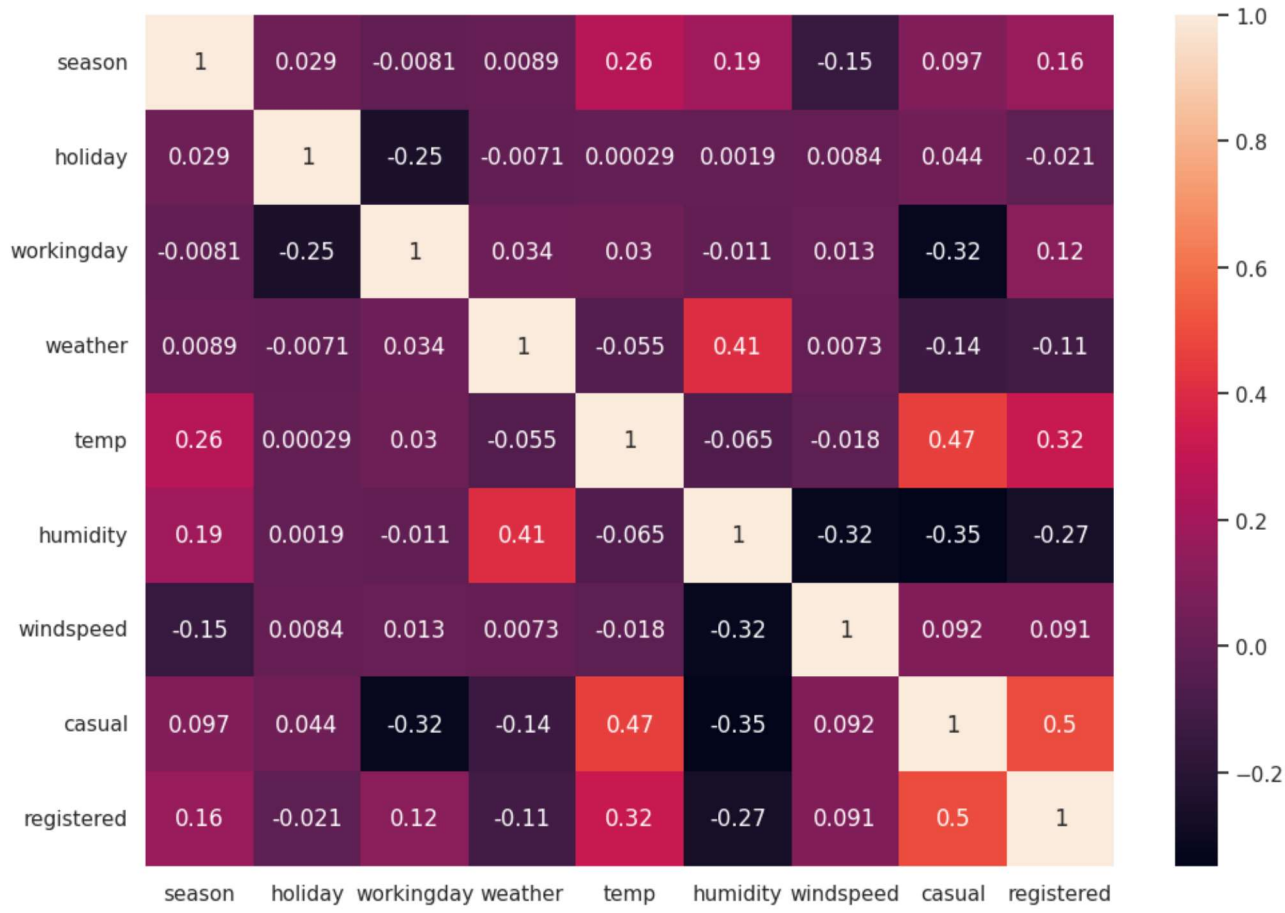
Relation between the Numerical Variables

Now, let us try to analyze the correlation between the different variables.

```
sns.set(rc={'figure.figsize':(11.7,8.27)})

features = ["season","holiday","workingday","weather","temp","humidity","windspeed","casual","registered"]
sns.heatmap(df[features].corr(), annot=True)
```

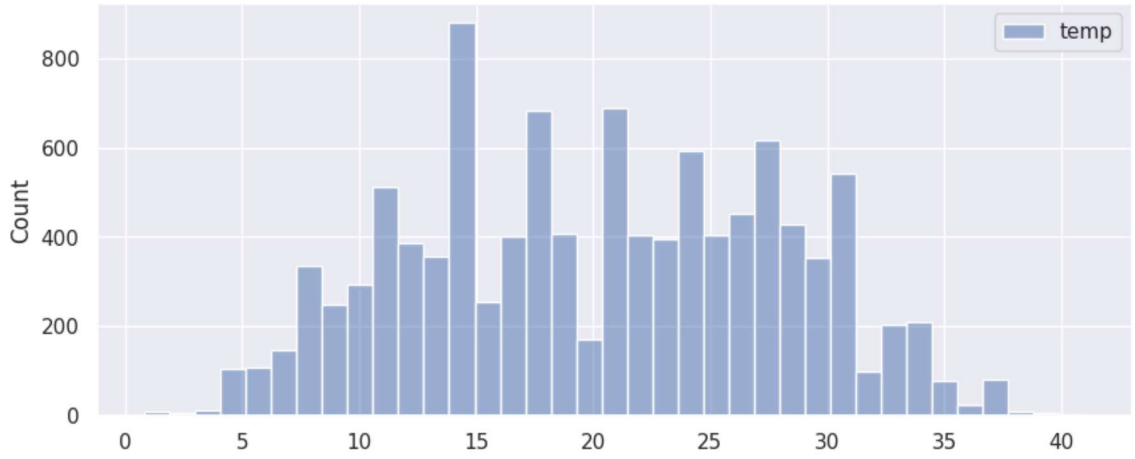
```
<Axes: >
```




```
sns.set(rc={'figure.figsize':(10,4)})
```

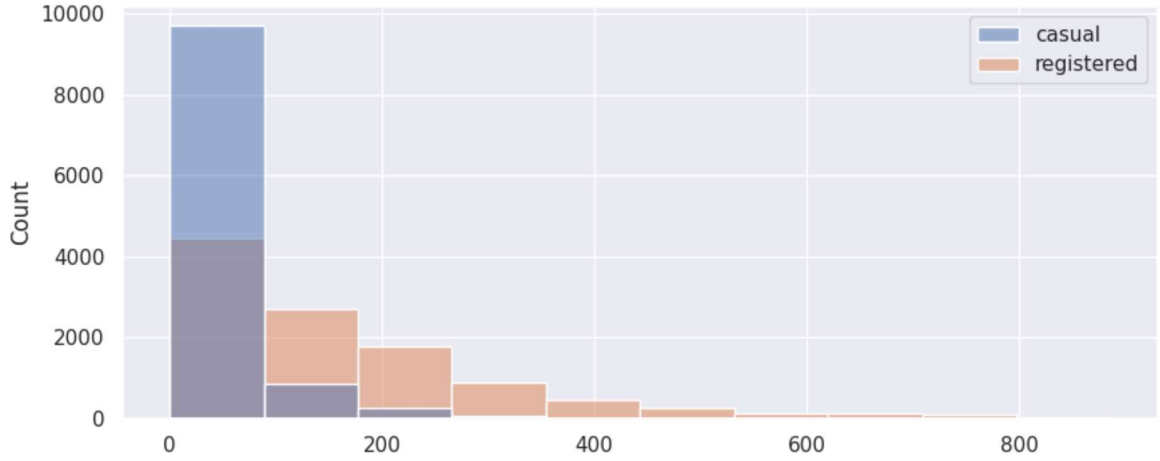
```
sns.histplot(df[["temp"]])
```

```
<Axes: ylabel='Count'>
```




```
sns.histplot(df[["casual","registered"]], bins = 10)
```

```
<Axes: ylabel='Count'>
```




```
df["Time_of_Day"]= pd.to_datetime(df['datetime']).dt.time
```

```
df.head()
```

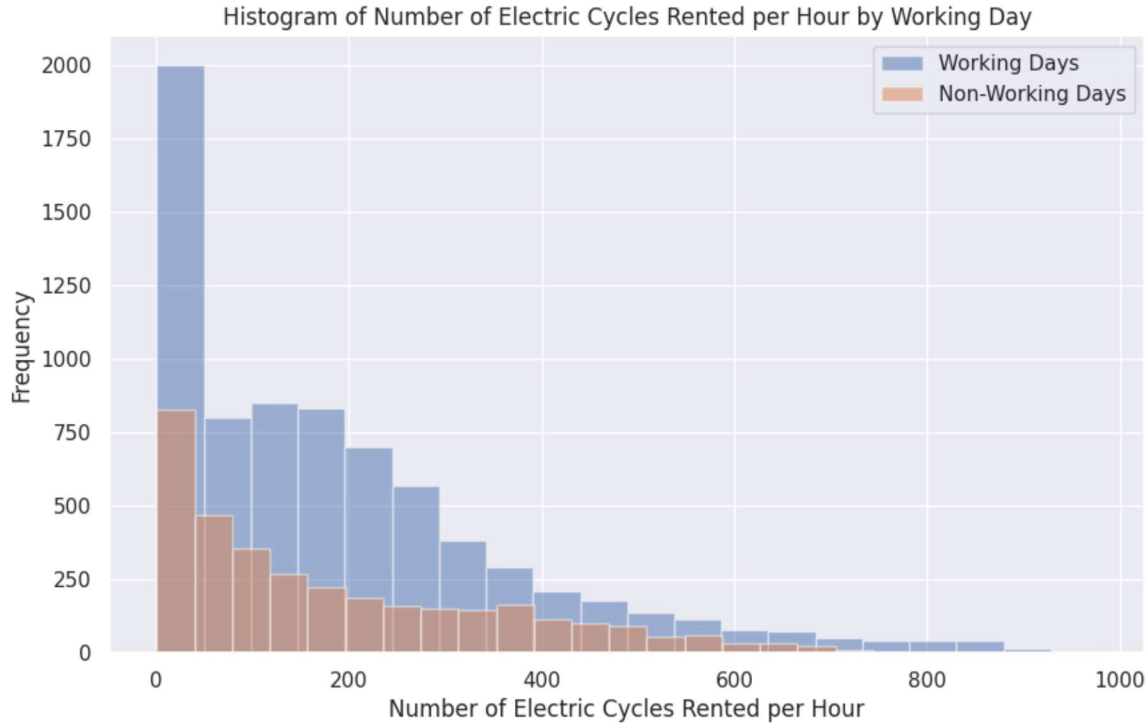
	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	count	Time_of_Day
0	2011-01-01 00:00:00	1	0	0	1	9.84	14.395	81	0.0	3	13	16	00:00:00
1	2011-01-01 01:00:00	1	0	0	1	9.02	13.635	80	0.0	8	32	40	01:00:00
2	2011-01-01 02:00:00	1	0	0	1	9.02	13.635	80	0.0	5	27	32	02:00:00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

Analysis of the distribution of the dataset.

```
# Subset data by workingday
df_work = df[df["workingday"] == 1]["count"] # working days
df_nowork = df[df["workingday"] == 0]["count"] # non-working days

# Plot histograms of cnt variable for each subset
plt.figure(figsize=(10,6))
plt.hist(df_work, bins=20, alpha=0.5, label="Working Days")
plt.hist(df_nowork, bins=20, alpha=0.5, label="Non-Working Days")
plt.xlabel("Number of Electric Cycles Rented per Hour")
```

```
plt.ylabel("Frequency")
plt.title("Histogram of Number of Electric Cycles Rented per Hour by Working Day")
plt.legend()
plt.show()
```



From the histogram, we can see that the distributions are skewed to the right, which means that they are not symmetric and have a long tail on the right. This suggests that the data may not follow a normal distribution, which is one of the assumptions of the t-test.

To further check the normality assumption, I will use Shapiro-Wilk test.

A Shapiro-Wilk test is a statistical test that tests the null hypothesis that the data is normally distributed. If the p-value of the test is less than a significance level (alpha), we reject the null hypothesis and conclude that the data is not normally distributed.

```
# Perform Shapiro-Wilk test for each subset for one random sample of size 200
print("Shapiro-Wilk test for working days:")
print(shapiro(df_work[:200])[1])

print("Shapiro-Wilk test for non-working days:")
print(shapiro(df_nowork[:200])[1])


# Perform Shapiro-Wilk test for each subset for another random sample of size 200
print("Shapiro-Wilk test for working days:")
print(shapiro(df_work[400:600])[1])

print("Shapiro-Wilk test for non-working days:")
print(shapiro(df_nowork[400:600])[1])
```

```
Shapiro-Wilk test for working days:
9.860852412391097e-11
Shapiro-Wilk test for non-working days:
2.555440747897819e-09
Shapiro-Wilk test for working days:
1.6119584556051336e-09
Shapiro-Wilk test for non-working days:
1.4146853827279315e-11
```

```
levene(df_nowork,df_work)
```

```
LeveneResult(statistic=0.004972848886504472, pvalue=0.9437823280916695)
```

```
kruskal(df_nowork,df_work)
```

```
KruskalResult(statistic=0.0016182887191034687, pvalue=0.9679113872727798)
```

```
f_oneway(df_nowork,df_work)
```

```
F_onewayResult(statistic=1.4631992635777575, pvalue=0.22644804226428558)
```

As we can see that even at 95% or 99% CI, with two random samples of size 200 the data is not normally distributed. This is statistically proved.

In a situation like this, we can perform Chi-Squared tests as chi-squared is a non-parametric test, meaning that it doesn't make any assumptions about the distributions of the data.

> p Value is greater than 0.05 in each case, which means that the means of people using the bikes on working day and non-working day are the same.

Affect of Seasons on the usage.

Lets try to figure out whether season has affect on the usage of the Yulu bikes?

```
df_season = df[["season","casual","registered"]].groupby(by = "season").mean()
df_season.reset_index(inplace=True,drop=True)
df_season["season_name"] = season_val
df_season = df_season.iloc[:,[2,0,1]]
# df_season.set_index('season_name', inplace=True)
df_season
```

	season_name	casual	registered
0	spring	15.489576	100.853686
1	summer	47.446762	167.804610
2	fall	52.220271	182.196853
3	winter	28.580834	170.407462

```
p_val = chi2_contingency(df_season[["casual","registered"]])[1]
p_val
```

```
if p_val < 0.05:
  print(f'As p value is {p_val}, we reject H0. Seasons affect the usage of Bikes')
else:
  print(f'As p value is {p_val}, we fail to reject H0. Seasons don\'t affect the usage of bikes')
```

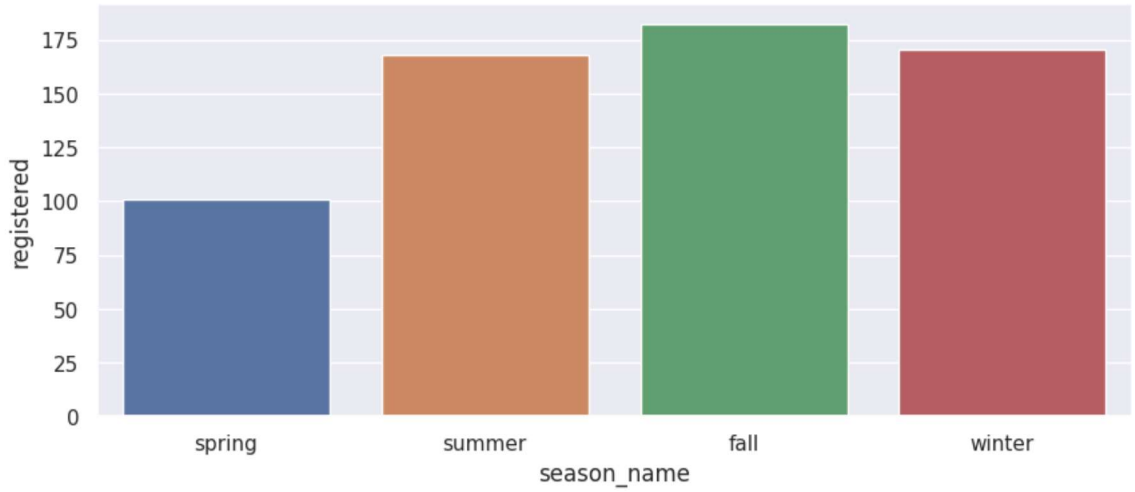
```
As p value is 0.042012107542006544, we reject H0. Seasons affect the usage of Bikes
```

```
sns.barplot(df_season, x ='season_name', y = 'casual')
```

```
<Axes: xlabel='season_name', ylabel='casual'>
```

```
sns.barplot(df_season, x ='season_name', y = 'registered')
```

```
<Axes: xlabel='season_name', ylabel='registered'>
```



Affect of Weather on the usage.

```
df_weather = df[["weather","casual","registered"]].groupby(by = "weather").mean()
df_weather.reset_index(inplace=True,drop=True)
df_weather["wthr_name"] = weather_val
df_weather.set_index('wthr_name')
df_weather = df_weather.iloc[:,[2,0,1]]
df_weather
```

	wthr_name	casual	registered
0	Clear	40.308676	164.928115
1	Mist	30.785462	148.170078
2	Light Snow/Rain	17.442375	101.403958
3	Heavy Rain	6.000000	158.000000

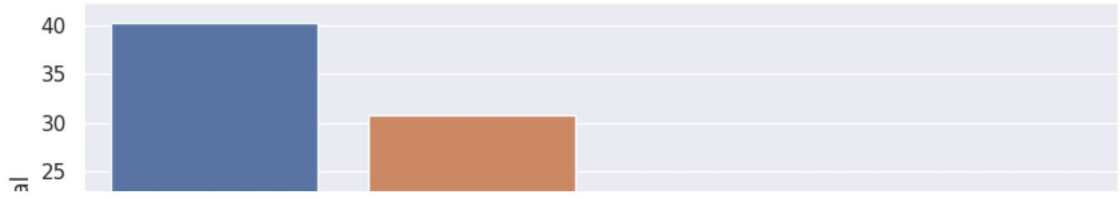
```
p_val = chi2_contingency(df_weather[["casual","registered"]])[1]
p_val

if p_val < 0.05:
  print(f'As p value is {p_val}, we reject H0. Weather affects the usage of Bikes')
else:
  print(f'As p value is {p_val}, we fail to reject H0. Weather don\'t affect the usage of bikes')
```

```
As p value is 9.017267191492623e-05, we reject H0. Weather affects the usage of Bikes
```

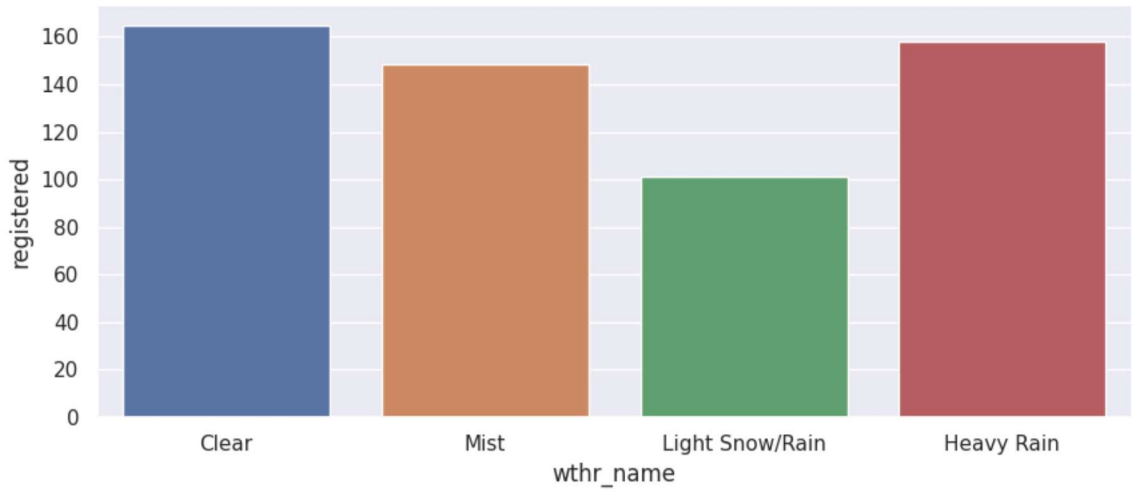
```
sns.barplot(df_weather, x ='wthr_name', y = 'casual')
```

```
<Axes: xlabel='wthr_name', ylabel='casual'>
```




```
sns.barplot(df_weather, x ='wthr_name', y = 'registered')
```

```
<Axes: xlabel='wthr_name', ylabel='registered'>
```



Double-click (or enter) to edit

Affect of Weather/ Season on No of Cycles Used

```
df["season"] = df.apply(season_name, axis =1)
```

```
df["weather"] = df.apply(wthr_name, axis =1)
```

```
df.head()
```

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

```
spring = np.array(df[df["season"]=="spring"]["count"])
summer = np.array(df[df["season"]=="summer"]["count"])
fall = np.array(df[df["season"]=="fall"]["count"])
winter = np.array(df[df["season"]=="winter"]["count"])

Clear = np.array(df[df["weather"]=="Clear"]["count"])
Mist = np.array(df[df["weather"]=="Mist"]["count"])
LSR = np.array(df[df["weather"]=="Light Snow/Rain"]["count"])
HR = np.array(df[df["weather"]=="Heavy Rain"]["count"])


_,p_val = f_oneway(spring,summer,fall,winter)
p_val
```

```
6.164843386499654e-149
```

```
_,p_val = f_oneway(Clear,Mist,LSR,HR)
p_val
```

```
5.482069475935669e-42
```

As we can see that in both the cases, the p value is much lesser than 0.05 or 0.01, which means that there is a difference in the number of cycles rented in the different seasons.

Check if Weather is dependent upon the Season.

```
ws = pd.crosstab(index=df["season"],columns=df["weather"],values=df["count"], aggfunc='mean',margins=True)
ws.fillna(0,inplace=True)
ws
```

weather	Clear	Heavy Rain	Light Snow/Rain	Mist	All
season					
fall	243.583420	0.0	156.582915	230.771523	234.417124
spring	126.781694	164.0	61.227488	106.861538	116.343261
summer	236.729595	0.0	123.906250	189.515537	215.251372
winter	209.511163	0.0	134.466667	194.784387	198.988296
All	205.236791	164.0	118.846333	178.955540	191.574132

```
chi2_contingency(ws)
```

```
Chi2ContingencyResult(statistic=636.8534636954454, pvalue=3.445768204028608e-125, dof=16, expected_freq=array([[232.55621419,
74.64792873, 135.41991182, 205.02884876,
        217.70207799],
       [154.5834818 ,  49.61955874,  90.01557558, 136.28564354,
        144.70972245],
       [205.69496969,  66.0257714 , 119.77832867, 181.34713354,
        192.5565502 ],
       [198.26368346,  63.64041212, 115.4510131 , 174.79547864,
        185.59992485],
       [230.74431358,  74.06632901, 134.36482317, 203.43142038,
        216.00590947]]))
```

As we can see that the p value is very close to 0, the Null Hypothesis doesn't stand true and the Weather and seasons are dependent upon each other.

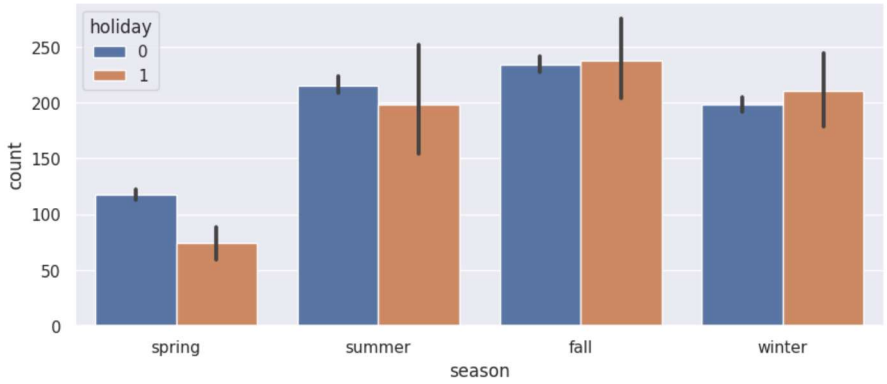
Graphical Analysis

```
df.head()
```

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

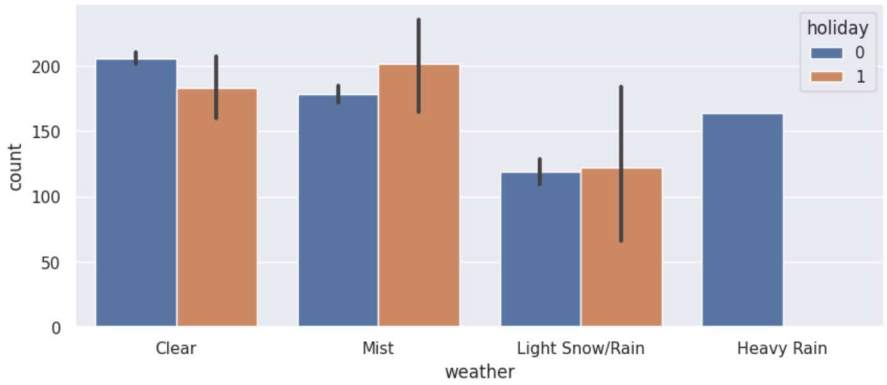
```
sns.barplot(
  df,
  x = "season",
  y="count",
  hue="holiday"
)
```

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



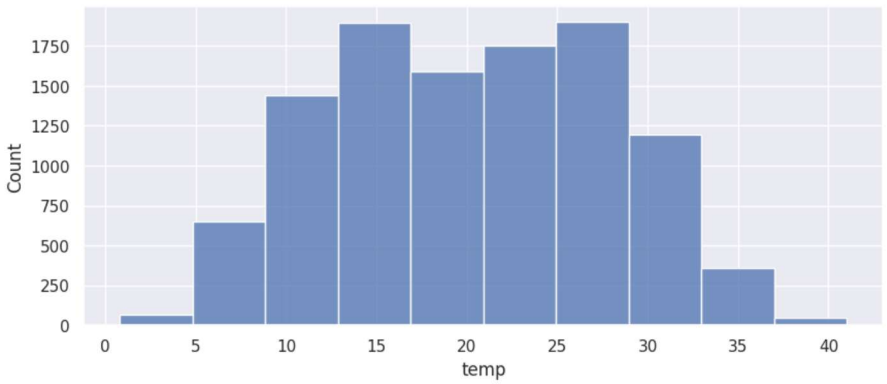

```
sns.barplot(
  df,
  x = "weather",
  y="count",
  hue="holiday"
)
```

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



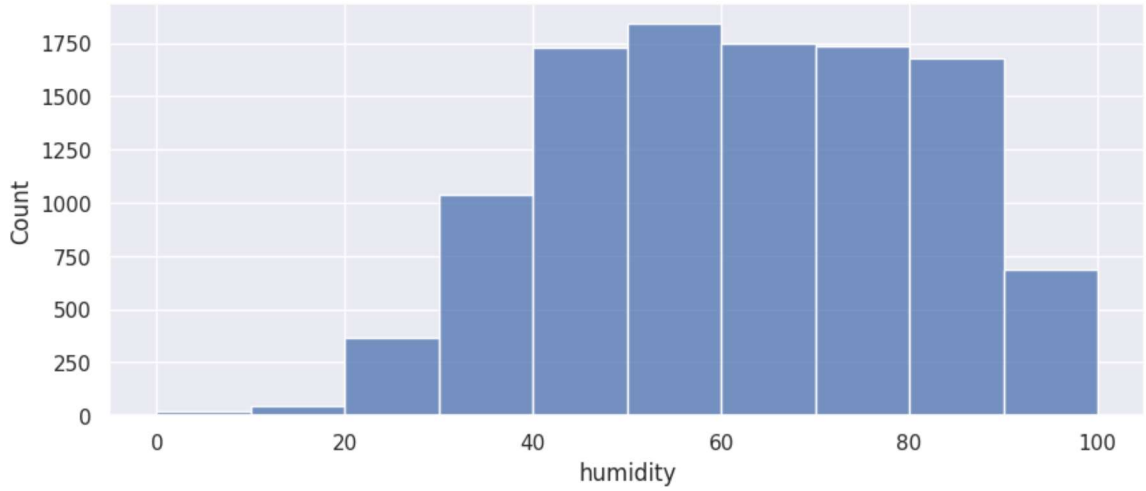

```
sns.histplot(df["temp"], bins = 10)
```

```
<Axes: xlabel='temp', ylabel='Count'>
```



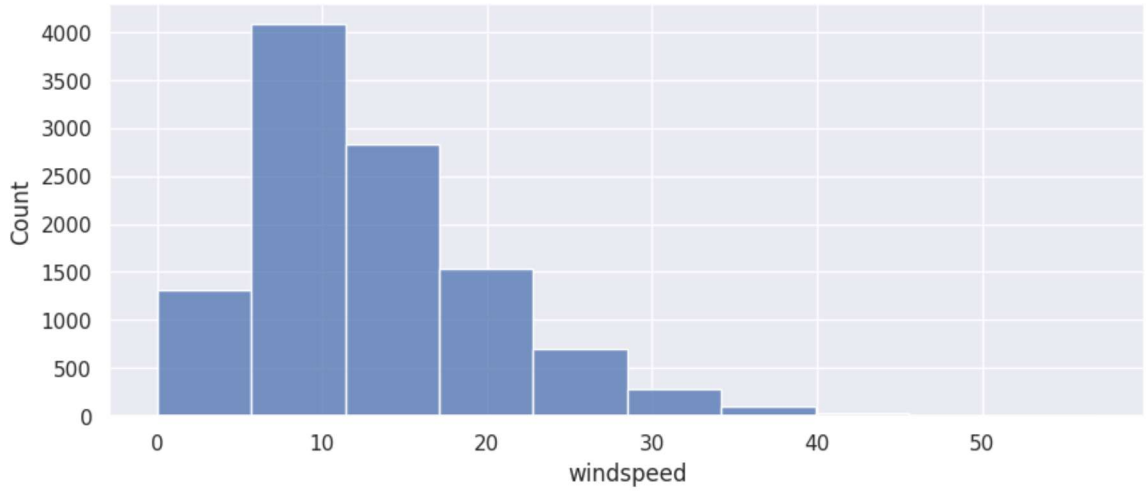

```
sns.histplot(df["humidity"], bins = 10)
```

```
<Axes: xlabel='humidity', ylabel='Count'>
```




```
sns.histplot(df["windspeed"], bins = 10)
```

```
<Axes: xlabel='windspeed', ylabel='Count'>
```



Observations and Conclusions:

```
* From the Heatmap, we can observe the following:
  - Casual users have a high correlation with windspeed, temp
  and season. This means that casual users are looking at Yul
  leisure activity and not afraid to make use of it during of

  - Registered users are highly correlated with working day a
  means that they view this as a more dependent means of comm
  activity.

  - Yulu can effectively target more robust bikes to register
  the transportation while any regular bikes can be marketted
  unregistered users.

* All the people in the sample like to use the bikes between
temperatures.

* Causal and registered users are all right skewed and not no
distributed.

* Even then, with multiple tests, we concluded that the group
the Yulu bikes for work vs for leisure are similar on an aver
```

- From the Heatmap, we can observe the following:
  - Casual users have a high correlation with windspeed, temperatures, holiday and season. This means that casual users are looking at Yulu for more of a leisure activity and not afraid to make use of it during off-season.
  - Registered users are highly correlated with working day and season, which means that they view this as a more dependent means of commute than leisure activity.
  - Yulu can effectively target more robust bikes to registered users to aid in the transportation while any regular bikes can be marketted towards unregistered users.
- All the people in the sample like to use the bikes between 12-30 degrees of temperatures.
- Causal and registered users are all right skewed and not normally distributed.

```
* Seasons seem of an affect on usage though registered and ca
them differently.

* Though there is difference in volume, both the groups show
during summer, fall and winter. So, spring can be a good time
for repair and maintenance checks.

* Weather, on the other hand, seems to have a totally differe
groups. Casual users seem to use it more during clear and mis
less in Snow / Rain scenarios.

* Registered users, on the other hand, seem to use the bikes
rain, compared to light rain or mist. This can be attributed
these users are turning towards bike for avoiding traffic. Mo
to be done to prove this hypothesis.

* Finally, we can see that weather and seasons are dependent
Further analysis can be done with individual groups to figure
seasons, how many cycles can be dispatched for effective usag

* However, with the current data, we can easily say that clea
Fall is the time when cycles are most used, closely followed
during fall.

* Clear weather on a summer seem to also be a huge draw for t

* 40-80 humidity and upto 30 windspeed also seems to affect t
posotovely.

* In conclusion, weather and temperature seems to be very pos
usage of bikes and Yulu can perform deeper analysis keeping t
to market their bikes and drive sales.
```

- Even then, with multiple tests, we concluded that the groups of people using the Yulu bikes for work vs for leisure are similar on an average.
- Seasons seem of an affect on usage though registered and casual users use them differently.
- Though there is difference in volume, both the groups show interest in bikes during summer, fall and winter. So, spring can be a good time to take the bikes for repair and maintenance checks.
- Weather, on the other hand, seems to have a totally different affect on the groups. Casual users seem to use it more during clear and misty weathers, while less in Snow / Rain scenarios.
- Registered users, on the other hand, seem to use the bikes more in heavy rain, compared to light rain or mist. This can be attributed to the fact that these users are turning towards bike for avoiding traffic. More analysis needs to be done to prove this hypothesis.
- Finally, we can see that weather and seasons are dependent on each other. Further analysis can be done with individual groups to figure out that in which seasons, how many cycles can be dispatched for effective usage.
- However, with the current data, we can easily say that clear Weather during Fall is the time when cycles are most used, closely followed by Misty weather during fall.
- Clear weather on a summer seem to also be a huge draw for the crowd.
- 40-80 humidity and upto 30 windspeed also seems to affect the usage posotovely.
- In conclusion, weather and temperature seems to be very posively affeting the usage of bikes and Yulu can perform deeper analysis keeping this factors in min to market their bikes and drive sales.