```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from scipy.stats import norm
from scipy.stats import binom, geom
import math
from scipy.stats import ttest_1samp, ttest_ind, ttest_rel
from scipy.stats import chisquare, chi, chi2, chi2_contingency
from scipy.stats import f_oneway, kruskal, shapiro
from statsmodels.graphics.gofplots import qqplot
```

```
yulu = pd.read_csv("https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/428/original/bike_sharing.csv?1642089089")
```

```
yulu.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

Exploratory data analysis steps

```
yulu.describe()
```

	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	count
count	10886.000000	10886.000000	10886.000000	10886.000000	10886.00000	10886.000000	10886.000000	10886.000000	10886.000000	10886.000000	10886.000000
mean	2.506614	0.028569	0.680875	1.418427	20.23086	23.655084	61.886460	12.799395	36.021955	155.552177	191.574132
std	1.116174	0.166599	0.466159	0.633839	7.79159	8.474601	19.245033	8.164537	49.960477	151.039033	181.144454
min	1.000000	0.000000	0.000000	1.000000	0.82000	0.760000	0.000000	0.000000	0.000000	0.000000	1.000000
25%	2.000000	0.000000	0.000000	1.000000	13.94000	16.665000	47.000000	7.001500	4.000000	36.000000	42.000000
50%	3.000000	0.000000	1.000000	1.000000	20.50000	24.240000	62.000000	12.998000	17.000000	118.000000	145.000000
75%	4.000000	0.000000	1.000000	2.000000	26.24000	31.060000	77.000000	16.997900	49.000000	222.000000	284.000000
max	4.000000	1.000000	1.000000	4.000000	41.00000	45.455000	100.000000	56.996900	367.000000	886.000000	977.000000

```
yulu.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
```

```
yulu["season"].nunique()
```

```
4
```

```
yulu["holiday"].nunique()
```

```
2
```

```
yulu["workingday"].nunique()
```

```
2
```

```
yulu["weather"].nunique()
```

```
4
```

```
yulu["temp"].nunique()
```

```
49
```

```
yulu["atemp"].nunique()
```

```
60
```

```
yulu["humidity"].nunique()
```

```
89
```

```
yulu["windspeed"].nunique()
```

```
28
```

```
yulu["casual"].nunique()
```

```
309
```

```
yulu["registered"].nunique()
```

```
731
```

```
yulu["count"].nunique()
```

```
822
```

```
yulu.groupby(["season"])["count"].agg(["min","max"])
```

season	min	max
1	1	801
2	1	873
3	1	977
4	1	948

```
yulu.groupby(["holiday"])["count"].agg(["min","max"])
```

holiday	min	max
0	1	977
1	1	712

```
yulu.groupby(["workingday"])["count"].agg(["min","max"])
```

workingday	min	max
0	1	783
1	1	977

```
yulu.groupby(["weather"])["count"].agg(["min","max"])
```

weather	min	max
1	1	977

```
yulu['datetime'] = pd.to_datetime(yulu['datetime'])
```

```
yulu.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  datetime64[ns]
 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: datetime64[ns](1), float64(3), int64(8)
memory usage: 1020.7 KB
```

```
yulu['date'] = pd.to_datetime(yulu['datetime']).dt.date
yulu['time'] = pd.to_datetime(yulu['datetime']).dt.time
yulu['year'] = pd.to_datetime(yulu['datetime']).dt.year
```

```
yulu.drop('datetime',axis=1,inplace= True)
```

```
yulu['date'].nunique()
```

```
456
```

```
yulu['time'].nunique()
```

```
24
```

```
yulu['year'].nunique()
```

```
2
```

```
yulu.groupby(['year'])['count'].agg(["min","max"])
```

	min	max
year		

```
yulu.describe(include=object)
```

	date	time
count	10886	10886
unique	456	24
top	2011-01-01	12:00:00
freq	24	456

Data Visualization

```
yulu.head()
```

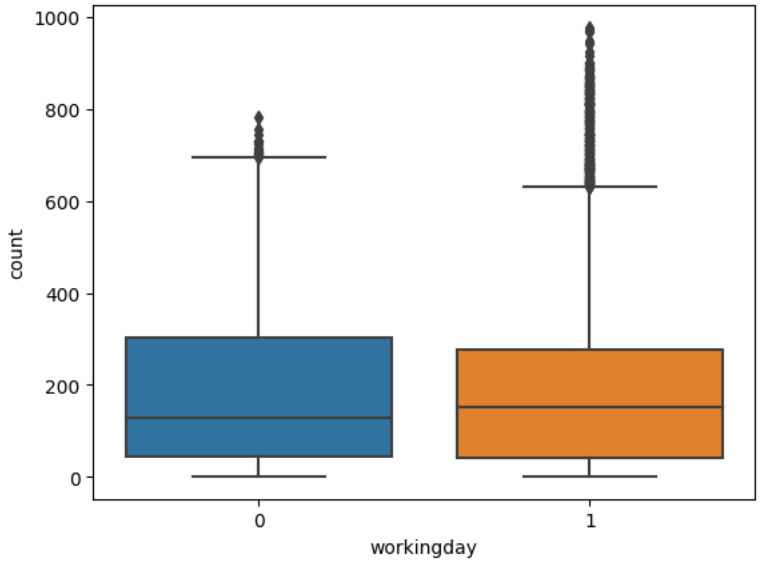
	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	count	date	time	year
0	1	0	0	1	9.84	14.395	81	0.0	3	13	16	2011-01-01	00:00:00	2011
1	1	0	0	1	9.02	13.635	80	0.0	8	32	40	2011-01-01	01:00:00	2011
2	1	0	0	1	9.02	13.635	80	0.0	5	27	32	2011-01-	02:00:00	2011

```
sns.kdeplot(data=yulu, x="count")
```

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


```
sns.boxplot(data=yulu, x='workingday', y='count')
```

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

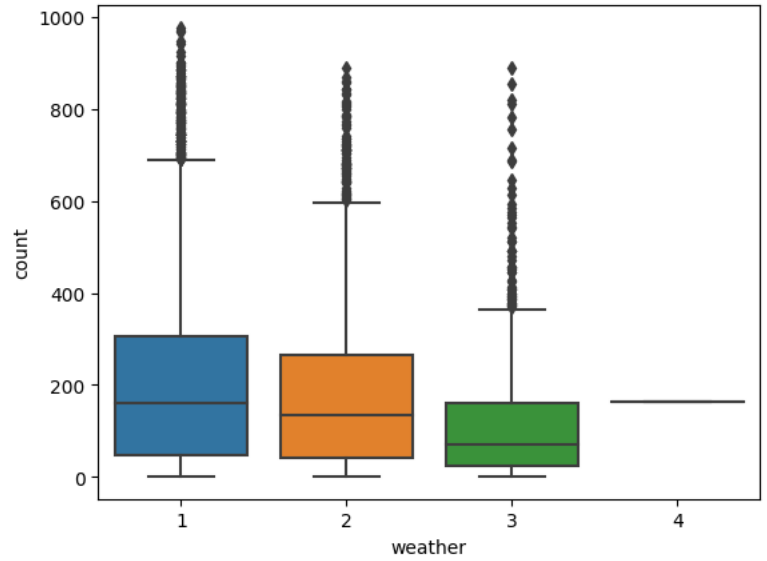



```
sns.boxplot(data=yulu, x='season', y='count')
```

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


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

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



Observations on above plot

1. The Box plot which workingday(if day is neither weekend nor holiday is 1, otherwise is 0) is relations between weather and count(no of total rental bikes including both casual and registered)* pepole use bikes on weekdays also more offten.
2. The Box plot is relations between season(1: spring, 2: summer, 3: fall, 4: winter) and count(no of total rental bikes including both casual and registered)* people use less bikes in spring where people use more bike in fall and summer and again we can see pepole using less bikes in winter.
3. The Box plot is relations between weather and count(no of total rental bikes including both casual and registered)* we can see that for Clear, Few clouds, partly cloudy & partly cloudy weather more rental bikes is used. and too low bikes is used on Heavy Rain + Ice Pallets + Thunderstorm + Mist & Snow + Fog weather.

---

*Hypothesis Testing *

2- Sample T-Test

```
workingdays = yulu.loc[(yulu["workingday"]==1)&(yulu["count"])]
not_workingdays = yulu.loc[(yulu["workingday"]==0)&(yulu["count"])]
holidays = yulu.loc[(yulu['holiday']==1)&(yulu['count'])]
not_holidays = yulu.loc[(yulu['holiday']==0)&(yulu['count'])]
```

```
workingdays.shape , not_workingdays.shape
```

```
((3676, 14), (1708, 14))
```

```
holidays.shape , not_holidays.shape
```

```
((152, 14), (5232, 14))
```

```
mean_workingdays = np.mean(workingdays["count"])
mean_not_workingdays =  np.mean(not_workingdays["count"])
print("mean_workingdays=", mean_workingdays)
print("mean_not_workingdays=", mean_not_workingdays)
```

```
mean_workingdays= 190.5963003264418
mean_not_workingdays= 188.9695550351288
```

```
sd_workingdays = np.std(workingdays["count"])
sd_not_workingdays = np.std(not_workingdays["count"])
print("sd_workingdays=", sd_workingdays)
print("sd_not_workingdays=", sd_not_workingdays)
```

```
sd_workingdays= 184.35309424998593
sd_not_workingdays= 173.95227752934838
```

```
mean_holidays = np.mean(holidays["count"])
mean_not_holodays =  np.mean(not_holidays["count"])
sd_holidays = np.std(holidays["count"])
sd_not_holidays = np.std(not_holidays["count"])

print("mean_holidays=", mean_holidays)
print("mean_not_holodays=", mean_not_holodays)
print("sd_holidays=", sd_holidays)
print("sd_not_holidays=", sd_not_holidays)
```

```
mean_holidays= 181.8815789473684
mean_not_holodays= 190.3184250764526
sd_holidays= 160.51885174622942
sd_not_holidays= 181.6779112530606
```

```
ttest_ind(workingdays["count"], not_workingdays["count"])  ## TTest to check if Working Day has an effect on the number of electric cycles rented ##
```

```
TtestResult(statistic=0.3066590378257357, pvalue=0.7591147744716455, df=5382.0)
```

```
ttest_ind(holidays["count"], not_holidays["count"]) ## Ttest to check if holi Day has an effect on the number of electric cycles rented ##
```

```
TtestResult(statistic=-0.5660430322527081, pvalue=0.5713881275169148, df=5382.0)
```

ANNOVA

```
yulu.groupby("weather")["count"].mean()
```

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

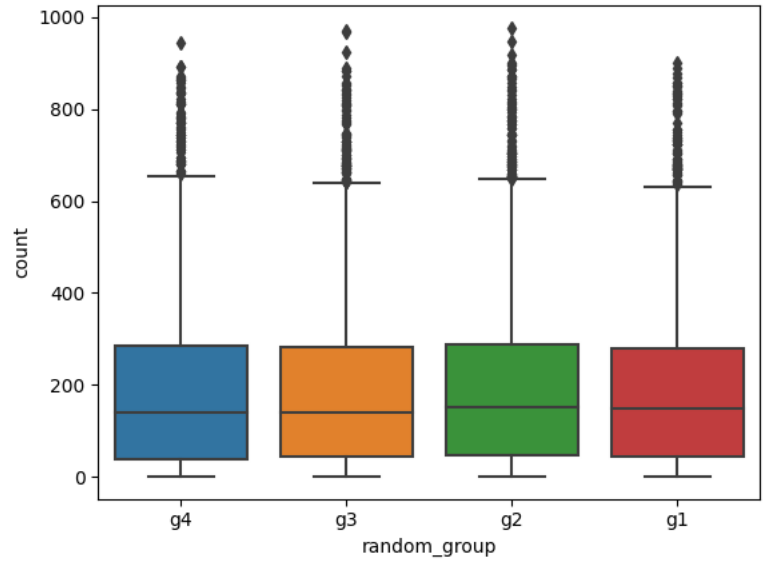
```
yulu["random_group"] = np.random.choice(["g1","g2","g3","g4"],size=len(yulu))
```

```
yulu.head()
```

	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	count	date	time	year	rando
0	1	0	0	1	9.84	14.395	81	0.0	3	13	16	2011-01-01	00:00:00	2011	
1	1	0	0	1	9.02	13.635	80	0.0	8	32	40	2011-01-01	01:00:00	2011	

```
sns.boxplot(x="random_group",y='count',data=yulu)
```

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



```
g1 = yulu[yulu["random_group"]=="g1"]["count"]
g2 = yulu[yulu["random_group"]=="g2"]["count"]
g3 = yulu[yulu["random_group"]=="g3"]["count"]
g4 = yulu[yulu["random_group"]=="g4"]["count"]
```

```
print(g1.mean())
print(g2.mean())
print(g3.mean())
print(g4.mean())
```

```
190.1445960900037
196.1858108108108
190.96884057971013
189.12431842966194
```

```
f_oneway(g1, g2, g3, g4)
```

```
F_onewayResult(statistic=0.809776511373607, pvalue=0.48822773376715767)
```

```
# H0: All groups have same mean
# Ha: One or more groups have different mean

f_stats, p_value = f_oneway(g1, g2, g3, g4)
if p_value < 0.05:
    print("Reject H0")
else:
    print("Fail to reject H0")
    print("All groups have same mean")
```

```
Fail to reject H0
All groups have same mean
```

```
yulu.head()
```

	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	count	date	time	year	random_group
0	1	0	0	1	9.84	14.395	81	0.0	3	13	16	2011-01-01	00:00:00	2011	g4
1	1	0	0	1	9.02	13.635	80	0.0	8	32	40	2011-01-01	01:00:00	2011	g3
2	1	0	0	1	9.02	13.635	80	0.0	5	27	32	2011-01-01	02:00:00	2011	g3
3	1	0	0	1	9.84	14.395	75	0.0	3	10	13	2011-01-01	03:00:00	2011	g3
4	1	0	0	1	9.84	14.395	75	0.0	0	1	1	2011-01-01	04:00:00	2011	g3

```
w1 = yulu[yulu["weather"]== 1]["count"]
w2 = yulu[yulu["weather"]== 2]["count"]
w3 = yulu[yulu["weather"]== 3]["count"]
w4 = yulu[yulu["weather"]== 4]["count"]
```

```
print(w1.mean())
print(w2.mean())
print(w3.mean())
print(w4.mean())
```

```
205.23679087875416
178.95553987297106
118.84633294528521
164.0
```

```
# H0: All groups have same mean
# Ha: One or more groups have different mean
f_oneway(w1,w2,w3,w4)
```

```
F_onewayResult(statistic=65.53024112793271, pvalue=5.482069475935669e-42)
```

```
kruskal(w1,w2,w3,w4)
```

```
KruskalResult(statistic=205.00216514479087, pvalue=3.501611300708679e-44)
```

```
kruskal(g1,g2,g3,g4)
```

```
KruskalResult(statistic=2.9299750979948245, pvalue=0.4025481236776548)
```

Chi-square test

```
yulu.head()
```

	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	count	date	time	year	rando
0	1	0	0	1	9.84	14.395	81	0.0	3	13	16	2011-01-01	00:00:00	2011	
1	1	0	0	1	9.02	13.635	80	0.0	8	32	40	2011-01-01	01:00:00	2011	

```
season_weather = pd.crosstab(index = yulu['season'],columns = yulu['weather'])
season_weather
```

weather	1	2	3	4

```
# H0: season does not impact of weather during rented the bikes
# Ha: season impact of weather during rented the bikes
chi_stat, p_value, df, exp_value = chi2_contingency(season_weather)
print(chi_stat)
print(p_value)
print(df)
print(exp_value)
if p_value < 0.05:
    print("Reject H0")
    print("Gender impacts product")
```

```
 49.158655596893624
 1.549925073686492e-07
 9
 [[1.77454639e+03 6.99258130e+02 2.11948742e+02 2.46738931e-01]
  [1.80559765e+03 7.11493845e+02 2.15657450e+02 2.51056403e-01]
  [1.80559765e+03 7.11493845e+02 2.15657450e+02 2.51056403e-01]
  [1.80625831e+03 7.11754180e+02 2.15736359e+02 2.51148264e-01]]
 Reject H0
 Gender impacts product
```