```
!gdown https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/428/original/bike_sharing.csv?16420890
```

```
Downloading...
From: https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/428/original/bike_sharing.csv?16420890
To: /content/bike_sharing.csv?16420890
100% 648k/648k [00:00<00:00, 23.6MB/s]
```

```
import numpy as np, pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import spearmanr
from scipy.stats import norm
from statsmodels.stats.weightstats import ztest
from scipy.stats import ttest_1samp, ttest_ind
from scipy.stats import f_oneway
from scipy.stats import chi2_contingency
```

```
df=pd.read_csv('bike_sharing.csv?16420890')
df
```

	datetime	season	holiday	workingday	weather	temp	atemp	humidity	win
0	2011-01-01 00:00:00	1	0	0	1	9.84	14.395	81	
1	2011-01-01 01:00:00	1	0	0	1	9.02	13.635	80	
2	2011-01-01 02:00:00	1	0	0	1	9.02	13.635	80	
3	2011-01-01 03:00:00	1	0	0	1	9.84	14.395	75	
4	2011-01-01 04:00:00	1	0	0	1	9.84	14.395	75	
...	...	...	...	...	...	...	...	...	...

```
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.nunique()
```

```
datetime      10886
season            4
holiday           2
workingday        2
weather           4
temp             49
atemp            60
humidity         89
windspeed        28
```

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

```
df.isnull().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
```

```
#object => typically categorical/IDs
#Numeric => Int64, Float64

df['datetime']=pd.to_datetime(df['datetime'])
df['season']=df['season'].astype('object')
df['holiday']=df['holiday'].astype('object')
df['workingday']=df['workingday'].astype('object')
df['weather']=df['weather'].astype('object')
```

```
df.dtypes
```

```
datetime      datetime64[ns]
season                object
holiday               object
workingday            object
weather               object
temp                 float64
atemp                float64
humidity               int64
windspeed            float64
casual                 int64
registered             int64
count                  int64
dtype: object
```

```
df.describe()
# only numeric features
```

	temp	atemp	humidity	windspeed	casual	regist
count	10886.00000	10886.000000	10886.000000	10886.000000	10886.000000	10886.00
mean	20.23086	23.655084	61.886460	12.799395	36.021955	155.55
std	7.79159	8.474601	19.245033	8.164537	49.960477	151.03
min	0.82000	0.760000	0.000000	0.000000	0.000000	0.00
25%	13.94000	16.665000	47.000000	7.001500	4.000000	36.00
50%	20.50000	24.240000	62.000000	12.998000	17.000000	118.00
75%	26.24000	31.060000	77.000000	16.997900	49.000000	222.00
max	41.00000	45.455000	100.000000	56.996900	367.000000	886.00

```
df.describe(include = ['object'])
# string/ obj features
```

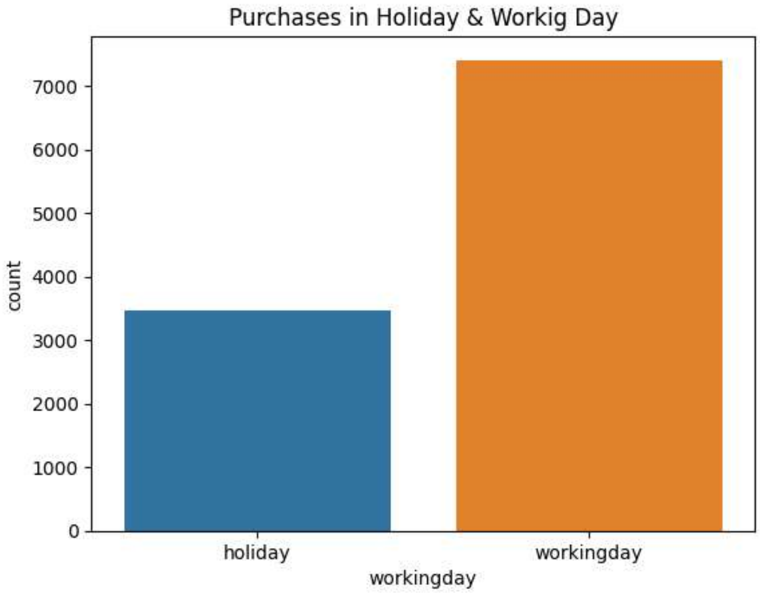
```
season  holiday  workingday  weather
```

```
df.head()
```

	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspee
0	2011-01-01 00:00:00	1	0	0	1	9.84	14.395	81	0.
1	2011-01-01 01:00:00	1	0	0	1	9.02	13.635	80	0.
	2011-01-								

Univariate Analysis

```
sns.countplot(data=df,x='workingday')
plt.xticks([0, 1], ['holiday', 'workingday'])
plt.title('Purchases in Holiday & Workig Day')
plt.show()
```

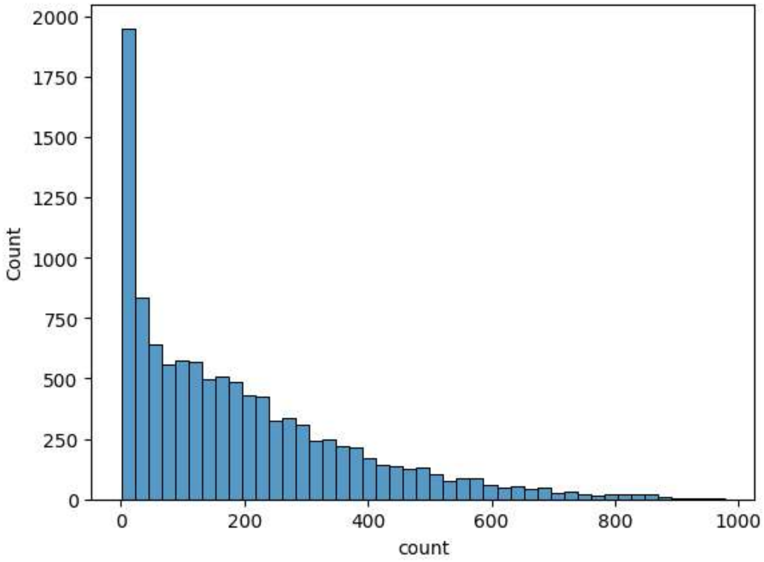


ChatGPT

```
sns.countplot(data=df,x='weather')
plt.title('No of purchases in diff Weather')
plt.show()
```

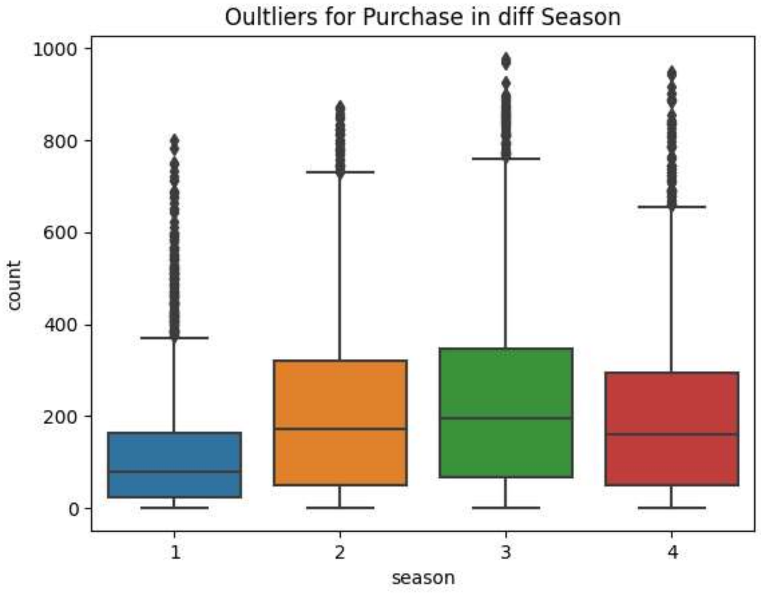
```
### Distribution of "count" variable is right skewed distribution:

sns.histplot(df['count'])
plt.show()
```

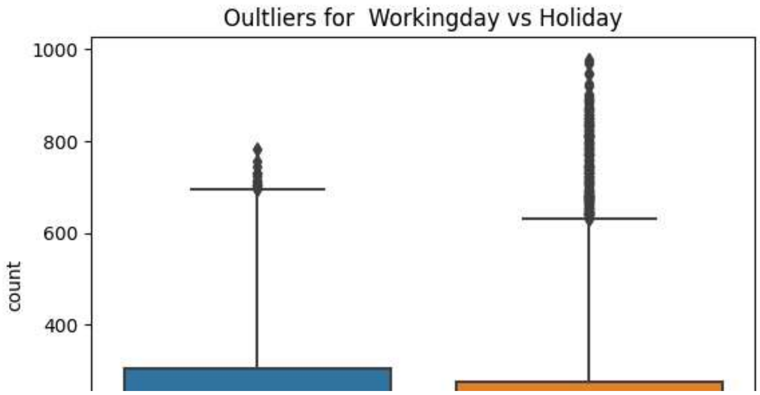


Bivariate Analysis

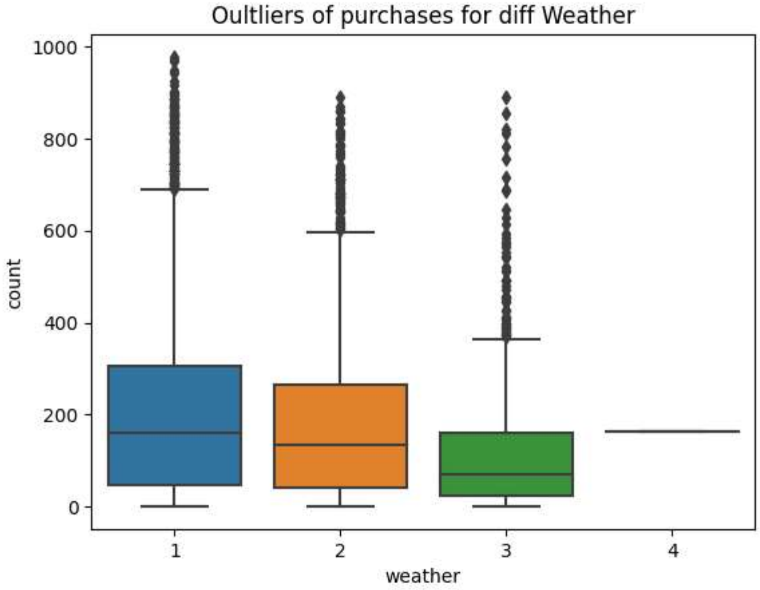
```
sns.boxplot(data=df,x='season',y='count')
plt.title('Oultliers for Purchase in diff Season')
plt.show()
```



```
sns.boxplot(data=df,x='workingday',y='count')
plt.xticks([0, 1], ['holiday', 'work_day'])
plt.title('Oultliers for  Workingday vs Holiday')
plt.show()
```



```
sns.boxplot(data=df,x='weather',y='count')
plt.title('Oultliers of purchases for diff Weather')
plt.show()
```



```
df.head()
```

	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspee
0	2011-01-01 00:00:00	1	0	0	1	9.84	14.395	81	0.
1	2011-01-01 01:00:00	1	0	0	1	9.02	13.635	80	0.
	2011-01-								

2- Sample T-Test

```
# H0 : no relation b/w workingday and cycles rented
# HA : relation b/w workingday and cycles rented

d1= df[df['workingday']==1]['count']
d2= df[df['workingday']==0]['count']
```

```
test_stat, p_value = ttest_ind(d1, d2,equal_var=True)
p_value
```

```
    0.22644804226361348
```

```
alpha = 0.05
print(p_value)
```

```
if p_value < 0.05:
    print("Reject H0")
else:
    print("failed to reject H0")
```

```
0.22644804226361348
failed to reject H0
```

ANNOVA

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

```
# H0 : mean of cycles rented is similar in diff weather
# Ha : mean of cycles rented varries in diff weather

f, p_value = f_oneway(weather_1, weather_2, weather_3,weather_4)
p_value
```

```
5.482069475935669e-42
```

ChatGPT

```
alpha = 0.05
print(p_value)

if p_value < 0.05:
    print("Reject H0")
else:
    print("failed to reject H0")
```

```
5.482069475935669e-42
Reject H0
```

Chi-square test

```
# H0 : Weather is not dependent on the season
# HA : Weather is  dependent on the season

weather_season = pd.crosstab(index = df['weather'],
                            columns=df['season'])
weather_season
```

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_stat, p_value, dof, exp_freq = chi2_contingency(weather_season)
print(p_value)
print(exp_freq)
```

```
1.5499250736864862e-07
[[1.77454639e+03 1.80559765e+03 1.80559765e+03 1.80625831e+03]
 [6.99258130e+02 7.11493845e+02 7.11493845e+02 7.11754180e+02]
 [2.11948742e+02 2.15657450e+02 2.15657450e+02 2.15736359e+02]
 [2.46738931e-01 2.51056403e-01 2.51056403e-01 2.51148264e-01]]
```

```
alpha = 0.05
print(p_value)

if p_value < 0.05:
    print("Reject H0")
else:
    print("failed to reject H0")
```

```
1.5499250736864862e-07
Reject H0
```

INSIGHTS :

1. The count column (i.e) no of purchases has a right skewed distribution
2. The most no of purchases are done during the working days and it has the high no of outliers
3. during the '4' weather(Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog) no purchases have been made
4. The median of "count" of season 2 & 3 is almost the same
5. In 2 sample t-test : the p-value is 22% and the confidence intervel is 5% so we faield to reject the null hypothesis
6. In the Annova test : since the p-value is very close to 0 we reject reject the null hypothesis so atleast one population mean differs from the rest of the group
7. In the chi-square test: the p-value is very low ,we reject reject the null hypothesis. And the weather is dependent on the season

RECOMENDATIONS :

1. We could implement an option for working people to lessen the price to get more no of working customers
2. We should also increase the number of bikes to be rented during the rush hours
3. we could give offers to tourist and first time users in order to get new customers
4. we could also make the purchase price vary for diff seasons so it could attract prople to use when the price is less

✓ 0s completed at 10:45 PM ● ×