```
import pandas as pd
import numpy as np
from scipy.stats import norm

import matplotlib.pyplot as plt
import seaborn as sns


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

	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	0.0000	3	13	16
1	2011-01-01 01:00:00	1	0	0	1	9.02	13.635	80	0.0000	8	32	40
2	2011-01-01 02:00:00	1	0	0	1	9.02	13.635	80	0.0000	5	27	32
3	2011-01-01 03:00:00	1	0	0	1	9.84	14.395	75	0.0000	3	10	13
4	2011-01-01 04:00:00	1	0	0	1	9.84	14.395	75	0.0000	0	1	1
...	...	...	...	...	...	...	...	...	...	...	...	...
10881	2012-12-19 19:00:00	4	0	1	1	15.58	19.695	50	26.0027	7	329	336
10882	2012-12-19 20:00:00	4	0	1	1	14.76	17.425	57	15.0013	10	231	241
10883	2012-12-19 21:00:00	4	0	1	1	13.94	15.910	61	15.0013	4	164	168
10884	2012-12-19 22:00:00	4	0	1	1	13.94	17.425	61	6.0032	12	117	129
10885	2012-12-19 23:00:00	4	0	1	1	13.12	16.665	66	8.9981	4	84	88

10886 rows × 12 columns

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
df.head(10)
```

	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	0.0000	3	13	16
1	2011-01-01 01:00:00	1	0	0	1	9.02	13.635	80	0.0000	8	32	40
2	2011-01-01 02:00:00	1	0	0	1	9.02	13.635	80	0.0000	5	27	32
3	2011-01-01 03:00:00	1	0	0	1	9.84	14.395	75	0.0000	3	10	13
4	2011-01-01 04:00:00	1	0	0	1	9.84	14.395	75	0.0000	0	1	1
5	2011-01-01 05:00:00	1	0	0	2	9.84	12.880	75	6.0032	0	1	1
6	2011-01-01 06:00:00	1	0	0	1	9.02	13.635	80	0.0000	2	0	2
7	2011-01-01 07:00:00	1	0	0	1	8.20	12.880	86	0.0000	1	2	3
8	2011-01-01 08:00:00	1	0	0	1	9.84	14.395	75	0.0000	1	7	8
9	2011-01-01 09:00:00	1	0	0	1	13.12	17.425	76	0.0000	8	6	14

Next steps: Generate code with df | View recommended plots | New interactive sheet

Examine dataset structure, characteristics, and statistical summary.

```
df.dtypes
```

	0
datetime	object
season	int64
holiday	int64
workingday	int64
weather	int64
temp	float64
atemp	float64
humidity	int64
windspeed	float64
casual	int64
registered	int64
count	int64

datetime is in object datatype, we need to convert it to datetime format

```
df.describe(include='object')
```

	datetime
count	10886
unique	10886
top	2012-12-19 23:00:00
freq	1

```
df['datetime']=pd.to_datetime(df['datetime'])
df.dtypes
```

	0
datetime	datetime64[ns]
season	int64
holiday	int64
workingday	int64
weather	int64
temp	float64
atemp	float64
humidity	int64
windspeed	float64
casual	int64
registered	int64
count	int64

```
df['datetime'].dt.year.max(),df['datetime'].dt.year.min(),df['datetime'].dt.month.max(),df['datetime'].dt.month.min()
```

```
(2012, 2011, 12, 1)
```

Dataset which we have is from Jan 2011 to Dec 2012 i.e., 2Years

Identify missing values and imputation if required?

```
df.isna().sum()
```

	0
datetime	0
season	0
holiday	0
workingday	0
weather	0
temp	0
atemp	0
humidity	0
windspeed	0
casual	0
registered	0
count	0

Checking Duplicate Rows

```
df[df.duplicated()].count()
```

	0
datetime	0
season	0
holiday	0
workingday	0
weather	0
temp	0
atemp	0
humidity	0
windspeed	0
casual	0
registered	0
count	0

Insights:

There are no null values

There are No duplicate values bold text

Double-click (or enter) to edit

```
for i in df.columns:
  print(i,':',df[i].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
```

```
df_copy=df.copy()
df_copy
```

	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	0.0000	3	13	16
1	2011-01-01 01:00:00	1	0	0	1	9.02	13.635	80	0.0000	8	32	40
2	2011-01-01 02:00:00	1	0	0	1	9.02	13.635	80	0.0000	5	27	32
3	2011-01-01 03:00:00	1	0	0	1	9.84	14.395	75	0.0000	3	10	13
4	2011-01-01 04:00:00	1	0	0	1	9.84	14.395	75	0.0000	0	1	1
...	...	...	...	...	...	...	...	...	...	...	...	...
10881	2012-12-19 19:00:00	4	0	1	1	15.58	19.695	50	26.0027	7	329	336
10882	2012-12-19 20:00:00	4	0	1	1	14.76	17.425	57	15.0013	10	231	241
10883	2012-12-19 21:00:00	4	0	1	1	13.94	15.910	61	15.0013	4	164	168
10884	2012-12-19 22:00:00	4	0	1	1	13.94	17.425	61	6.0032	12	117	129
10885	2012-12-19 23:00:00	4	0	1	1	13.12	16.665	66	8.9981	4	84	88

10886 rows × 12 columns

Next steps: Generate code with df_copy | View recommended plots | New interactive sheet

"Convert all categorical attributes in the dataset to the 'category' data type?"

```
mapping_dict={'season':{1:'spring',2:'summer',3:'fall',4:'winter'},'holiday':{1:'yes',0:'no'},'workingday':{1:'yes',0:'no'},'weather':{1:'clear',2:'cl
df_copy.replace(mapping_dict,inplace=True)
```

```
df_copy
```

	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	count
0	2011-01-01 00:00:00	spring	no	no	clear	9.84	14.395	81	0.0000	3	13	16
1	2011-01-01 01:00:00	spring	no	no	clear	9.02	13.635	80	0.0000	8	32	40
2	2011-01-01 02:00:00	spring	no	no	clear	9.02	13.635	80	0.0000	5	27	32
3	2011-01-01 03:00:00	spring	no	no	clear	9.84	14.395	75	0.0000	3	10	13
4	2011-01-01 04:00:00	spring	no	no	clear	9.84	14.395	75	0.0000	0	1	1
...	...	...	...	...	...	...	...	...	...	...	...	...
10881	2012-12-19 19:00:00	winter	no	yes	clear	15.58	19.695	50	26.0027	7	329	336
10882	2012-12-19 20:00:00	winter	no	yes	clear	14.76	17.425	57	15.0013	10	231	241
10883	2012-12-19 21:00:00	winter	no	yes	clear	13.94	15.910	61	15.0013	4	164	168
10884	2012-12-19 22:00:00	winter	no	yes	clear	13.94	17.425	61	6.0032	12	117	129
10885	2012-12-19 23:00:00	winter	no	yes	clear	13.12	16.665	66	8.9981	4	84	88

10886 rows × 12 columns

Next steps: Generate code with df_copy | View recommended plots | New interactive sheet

```
df.rename(columns={'count':'Totalcount'},inplace=True)
```

```
for i in df_copy.select_dtypes(include=['object','category']).columns:
  print(i,':',df_copy[i].value_counts())
```

```
season : season
winter    2734
summer    2733
fall      2733
spring    2686
Name: count, dtype: int64
holiday : holiday
no     10575
yes      311
Name: count, dtype: int64
workingday : workingday
yes    7412
no     3474
Name: count, dtype: int64
weather : weather
clear         7192
cloudy        2834
light rain     859
heavy rain       1
Name: count, dtype: int64
```

Statistical Summary

```
df_copy.describe()
```

	datetime	temp	atemp	humidity	windspeed	casual	registered	count
count	10886	10886.00000	10886.000000	10886.000000	10886.000000	10886.000000	10886.000000	10886.000000
mean	2011-12-27 05:56:22.399411968	20.23086	23.655084	61.886460	12.799395	36.021955	155.552177	191.574132
min	2011-01-01 00:00:00	0.82000	0.760000	0.000000	0.000000	0.000000	0.000000	1.000000
25%	2011-07-02 07:15:00	13.94000	16.665000	47.000000	7.001500	4.000000	36.000000	42.000000
50%	2012-01-01 20:30:00	20.50000	24.240000	62.000000	12.998000	17.000000	118.000000	145.000000
75%	2012-07-01 12:45:00	26.24000	31.060000	77.000000	16.997900	49.000000	222.000000	284.000000
max	2012-12-19 23:00:00	41.00000	45.455000	100.000000	56.996900	367.000000	886.000000	977.000000
std	NaN	7.79159	8.474601	19.245033	8.164537	49.960477	151.039033	181.144454

Insights

Average temp is 20.23 degree.

Average number of bike rented/session is 191.

Maximum number of bike rented /session is 977.

```
df_copy.describe(include='object')
```

	season	holiday	workingday	weather
count	10886	10886	10886	10886
unique	4	2	2	4
top	winter	no	yes	clear
freq	2734	10575	7412	7192

Weather is most of the times clear i.e.,7192

Data Info

```
df_copy.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  object
 2   holiday     10886 non-null  object
 3   workingday  10886 non-null  object
 4   weather     10886 non-null  object
 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(4), object(4)
memory usage: 1020.7+ KB
```

```
df_copy.dtypes
```

	0
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

```
df_copy.shape
```

```
(10886, 12)
```

Check for Outliers and Treatment if required

```
df_num=df_copy.select_dtypes(include=['float64','int64'])
df_num.head(5)
```

	temp	atemp	humidity	windspeed	casual	registered	count
0	9.84	14.395	81	0.0	3	13	16
1	9.02	13.635	80	0.0	8	32	40
2	9.02	13.635	80	0.0	5	27	32
3	9.84	14.395	75	0.0	3	10	13
4	9.84	14.395	75	0.0	0	1	1

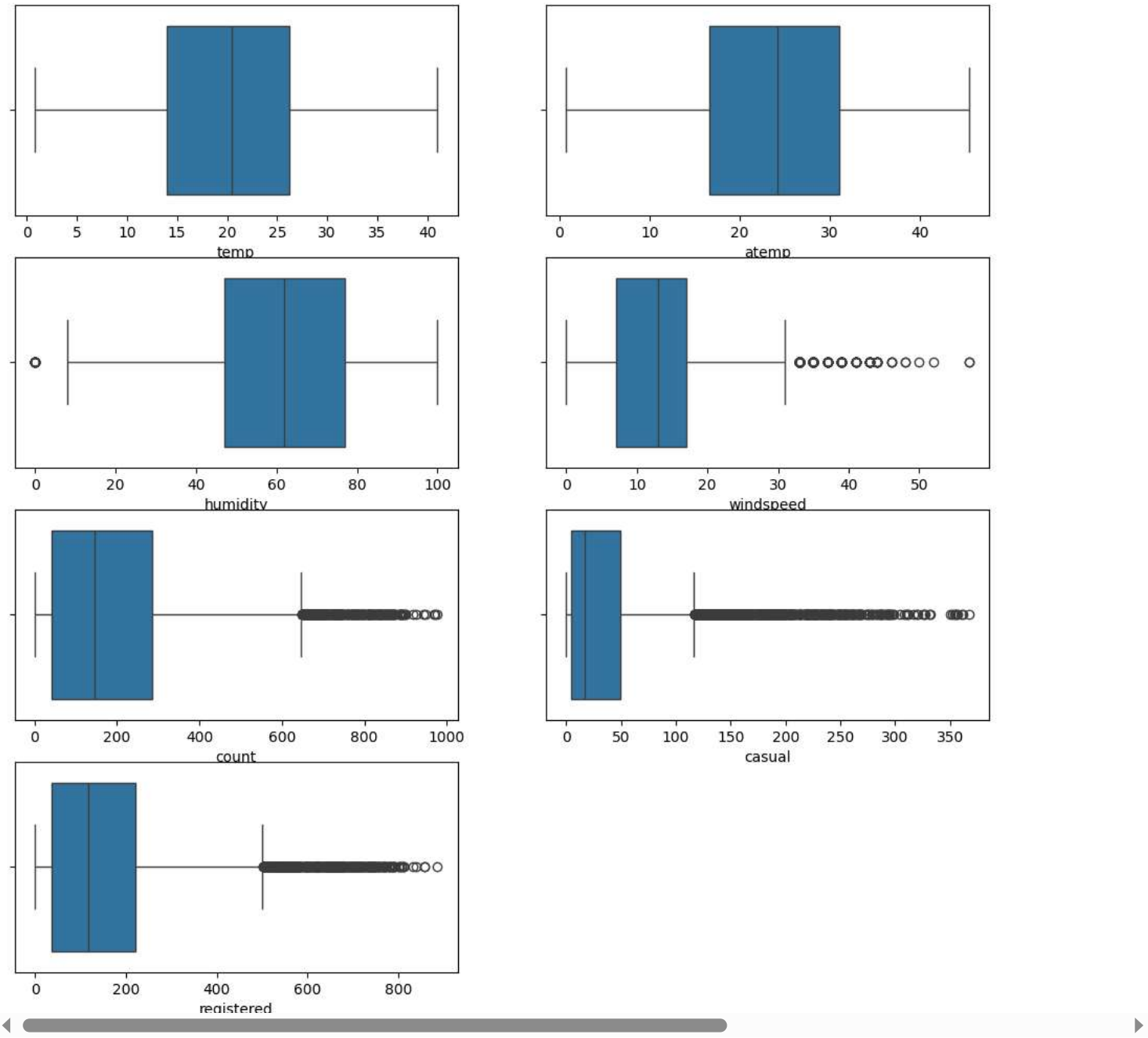
Next steps: Generate code with df_num | View recommended plots | New interactive sheet

```
df_num.describe()
```

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

```
plt.figure(figsize=(12,12))

plt.subplot(4,2,1)
sns.boxplot(data=df_num,x='temp')

plt.subplot(4,2,2)
sns.boxplot(data=df_num,x='atemp')

plt.subplot(4,2,3)
sns.boxplot(data=df_num,x='humidity')

plt.subplot(4,2,4)
sns.boxplot(data=df_num,x='windspeed')

plt.subplot(4,2,5)
sns.boxplot(data=df_num,x='count')

plt.subplot(4,2,6)
sns.boxplot(data=df_num,x='casual')
```

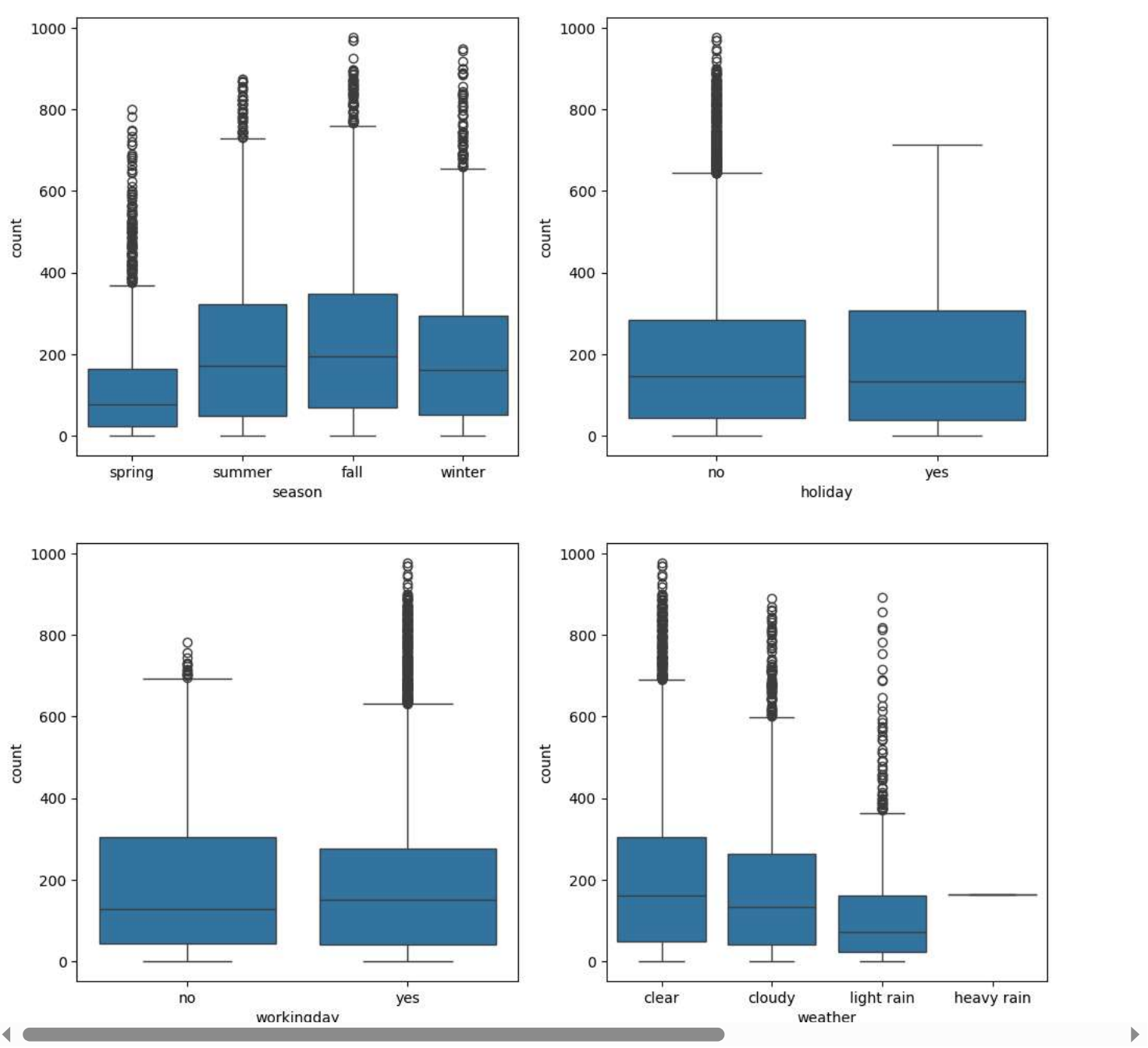
```
plt.subplot(4,2,7)
sns.boxplot(data=df_num,x='registered')

plt.show()
```



Insights:

Most of the outliers are in casual, registered and count columns

```
plt.figure(figsize=(12,12))

plt.subplot(2,2,1)
sns.boxplot(x=df_copy['season'],y=df_copy['count'])

plt.subplot(2,2,2)
sns.boxplot(x=df_copy['holiday'],y=df_copy['count'])

plt.subplot(2,2,3)
sns.boxplot(x=df_copy['workingday'],y=df_copy['count'])

plt.subplot(2,2,4)
sns.boxplot(x=df_copy['weather'],y=df_copy['count'])

plt.show()
```



We can see outliers are present for every feature column and we have to figure out whether to keep these outliers or not or to treat them.

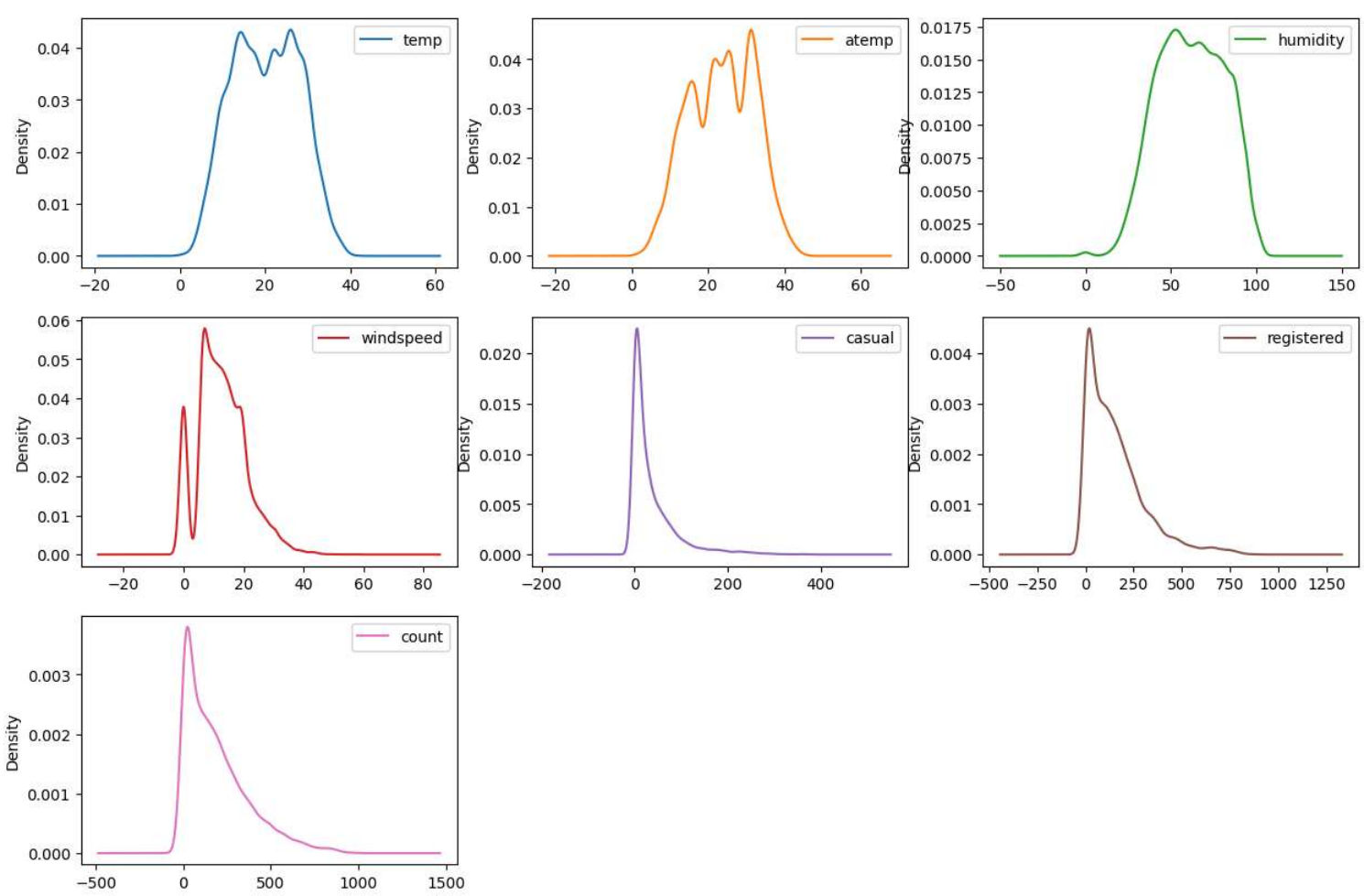
- Outliers are given as number of bike rides per session/day and column Total_count can be treated as target column which will help in finding the revenue for the company.
- These total counts could be higher because of some occasions, festivals. We cant remove or clip these outliers as these outliers are important and will increase revenue of the company and these outliers are not an valid error they are valid occurences. Hypothesis testing is not done on dataframe where outliers are not present because of this reason only. However removing outliers and clipping clipping is done after making a different copy of DF.

This case study primary goal is to help the management to improve the revenue and align with companies vision.

Treatment of outliers

Density Curve

```
plt.rcParams['figure.figsize']=[15,10]
df_num.plot(kind= 'density',subplots= True,layout=(3,3),sharex=False)
plt.show()
```



Insights

- Columns such as casual,registered,totak_count are right skewed having outliers at the right tail end.
- Whereas columns of temp,atemp,humidity are uniformly distributed

```
Q1=df_num.quantile(.25)
Q3=df_num.quantile(.75)
IQR=Q3-Q1
IQR
```

	0
temp	12.3000
atemp	14.3950
humidity	30.0000
windspeed	9.9964
casual	45.0000
registered	186.0000
count	242.0000

```
print('Q1:' ,Q1)
print('Q3:',Q3)
```

```
Q1: temp          13.9400
atemp         16.6650
humidity      47.0000
windspeed      7.0015
casual         4.0000
registered    36.0000
count         42.0000
Name: 0.25, dtype: float64
Q3: temp           26.2400
atemp          31.0600
humidity       77.0000
windspeed      16.9979
casual         49.0000
registered    222.0000
```

```
count         284.0000
Name: 0.75, dtype: float64
```

```
((df_num<(Q1-1.5*IQR))|(df_num>(Q3+1.5*IQR))).any(axis=1).sum()
```

```
np.int64(1368)
```

1368 rows has outliers

```
df[~((df_num<(Q1-1.5*IQR))|(df_num>(Q3+1.5*IQR))).any(axis=1)]
```

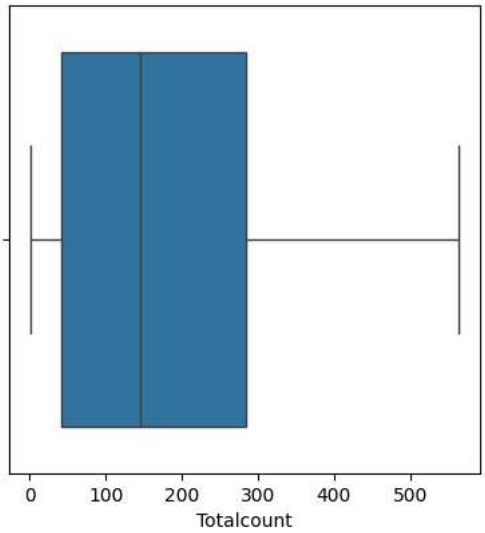
	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	Totalcount
0	2011-01-01 00:00:00	1	0	0	1	9.84	14.395	81	0.0000	3	13	16
1	2011-01-01 01:00:00	1	0	0	1	9.02	13.635	80	0.0000	8	32	40
2	2011-01-01 02:00:00	1	0	0	1	9.02	13.635	80	0.0000	5	27	32
3	2011-01-01 03:00:00	1	0	0	1	9.84	14.395	75	0.0000	3	10	13
4	2011-01-01 04:00:00	1	0	0	1	9.84	14.395	75	0.0000	0	1	1
...	...	...	...	...	...	...	...	...	...	...	...	...
10881	2012-12-19 19:00:00	4	0	1	1	15.58	19.695	50	26.0027	7	329	336
10882	2012-12-19 20:00:00	4	0	1	1	14.76	17.425	57	15.0013	10	231	241
10883	2012-12-19 21:00:00	4	0	1	1	13.94	15.910	61	15.0013	4	164	168
10884	2012-12-19 22:00:00	4	0	1	1	13.94	17.425	61	6.0032	12	117	129
10885	2012-12-19 23:00:00	4	0	1	1	13.12	16.665	66	8.9981	4	84	88

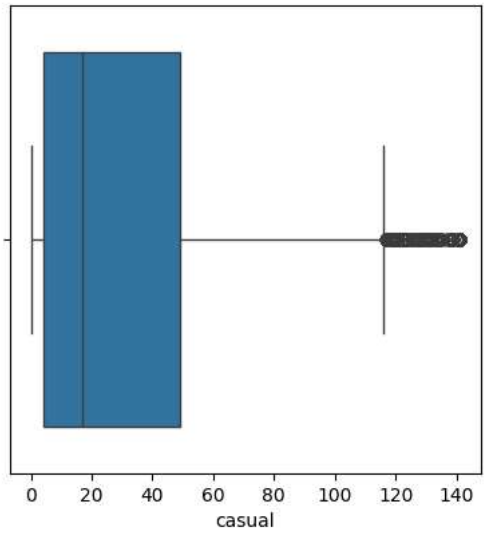
9518 rows × 12 columns

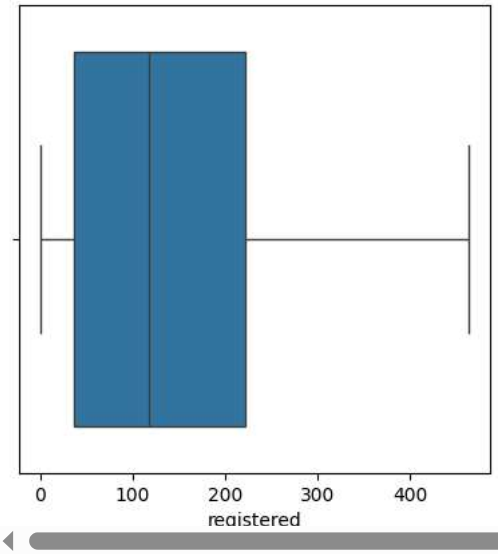
9518 Rows doesn't have Outliers

Columns such as casual,registered and total count are postive skewed i.e outliers are at the right tailend more we should clip them to 95th percentile

```
df_clip=df.copy()
percentile=df_clip['Totalcount'].quantile([0,.95]).values
df_clip['Totalcount']=np.clip(df_clip['Totalcount'],percentile[0],percentile[1])


percentile2=df_clip['casual'].quantile([0,.95]).values
df_clip['casual']=np.clip(df_clip['casual'],percentile2[0],percentile2[1])


percentile3=df_clip['registered'].quantile([0,.95]).values
df_clip['registered']=np.clip(df_clip['registered'],percentile3[0],percentile3[1])


plt.figure(figsize=(10,10))
plt.subplot(2,2,1)
sns.boxplot(x=df_clip['Totalcount'])

plt.subplot(2,2,2)
sns.boxplot(x=df_clip['casual'])

plt.subplot(2,2,3)
sns.boxplot(x=df_clip['registered'])

plt.show()
```







Outliers are Clipped to 95 Percentile

Skewness Values

```
for i in df_num.columns:
  print(i,':',df_num[i].skew())
```

```
temp : 0.003690844422472008
atemp : -0.10255951346908665
humidity : -0.08633518364548581
windspeed : 0.5887665265853944
casual : 2.4957483979812567
registered : 1.5248045868182296
count : 1.2420662117180776
```

Kurtosis Values

```
for i in df_num.columns:
  print(i,':',df_num[i].kurt())
```

```
temp : -0.9145302637630794
atemp : -0.8500756471754651
humidity : -0.7598175375208864
windspeed : 0.6301328693364932
casual : 7.551629305632764
registered : 2.6260809999210672
count : 1.3000929518398334
```

Insights:

- No of outliers in Casual column are high
- No of outliers in count and registered are present
- Casual column is more peaked i.e less data is spread out.

Univarite Analysis

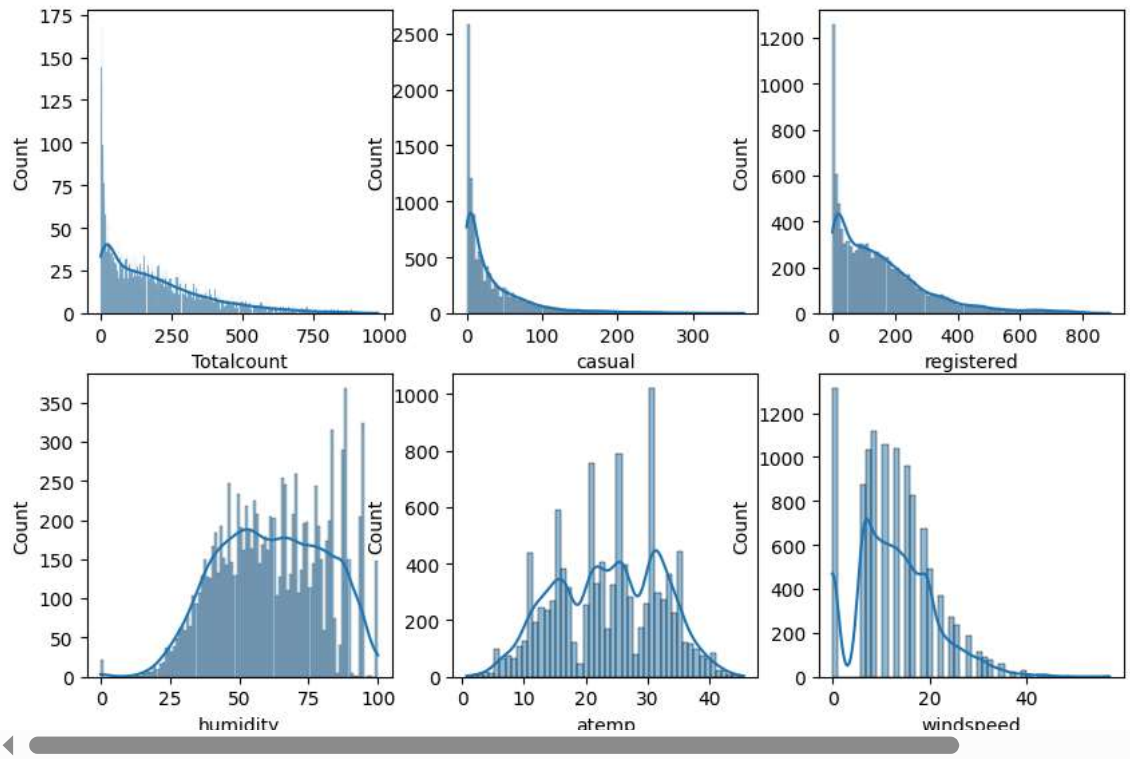
```
df.head(5)
```

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

Next steps: Generate code with df | View recommended plots | New interactive sheet

Histogram

```
plt.figure(figsize=(10,10))

plt.subplot(3,3,1)
sns.histplot(df['Totalcount'],kde=True,bins = 1000)

plt.subplot(3,3,2)
sns.histplot(df['casual'],kde=True,bins = 100)

plt.subplot(3,3,3)
sns.histplot(df['registered'],kde=True,bins = 100)

plt.subplot(3,3,4)
sns.histplot(df['humidity'],kde=True,bins = 100)

plt.subplot(3,3,5)
sns.histplot(df['atemp'],kde=True,bins = 50)

plt.subplot(3,3,6)
sns.histplot(df['windspeed'],kde=True,bins = 50)

plt.show()
```



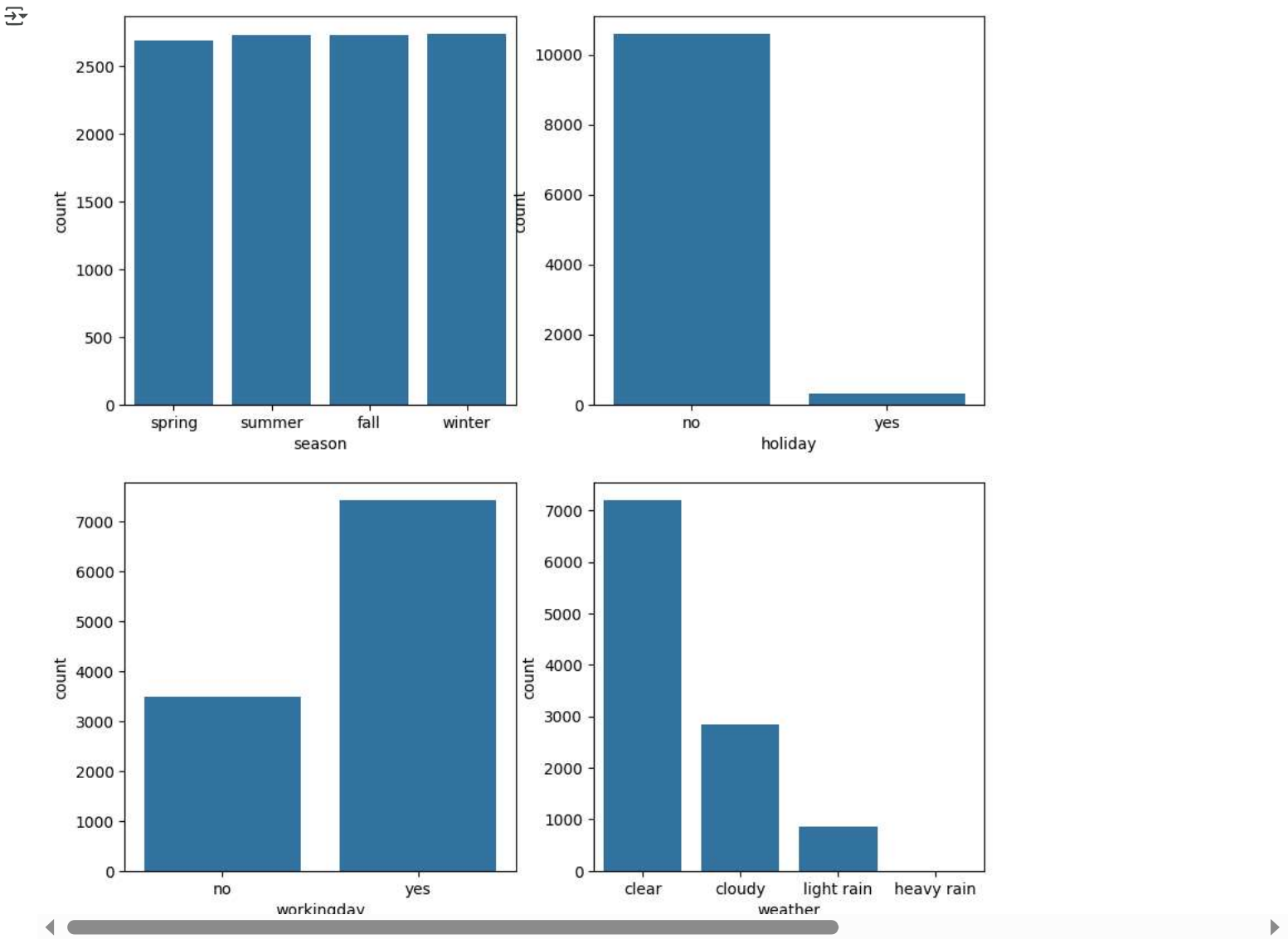
Insights:

- Total count, casual and registered are right skewed having high no of outliers
- Humidity and atemp are most like uniformly distributed but humidity has little left skewed

Countplot

```
plt.figure(figsize=(10,10))

plt.subplot(2,2,1)
sns.countplot(x=df_copy['season'],)
```

```
plt.subplot(2,2,2)
sns.countplot(x=df_copy['holiday'])

plt.subplot(2,2,3)
sns.countplot(x=df_copy['workingday'])

plt.subplot(2,2,4)
sns.countplot(x=df_copy['weather'])

plt.show()
```



Insights:

Weather is clear on most of the days.

Bikes are used more on working days and on non holidays because people use it to commute to office.

On Heavy rain people don't prefer to use it

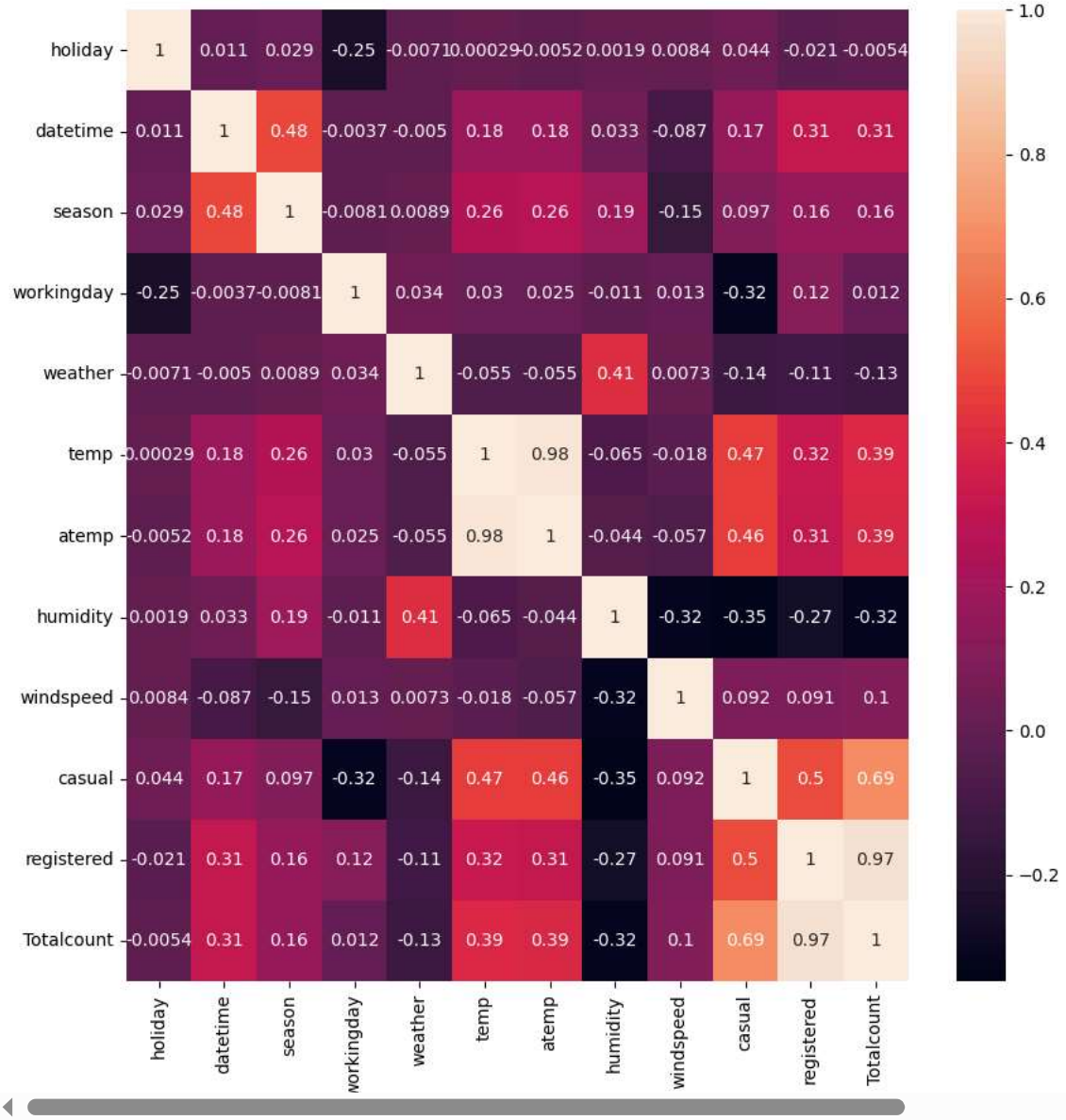
Relation between variables (Corelation)

```
df[['holiday','datetime','season','workingday','weather','temp','atemp','humidity','windspeed','casual','registered','Totalcount']].corr()
```

	holiday	datetime	season	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	Totalcount
holiday	1.000000	0.010988	0.029368	-0.250491	-0.007074	0.000295	-0.005215	0.001929	0.008409	0.043799	-0.020956	-0.005393
datetime	0.010988	1.000000	0.480021	-0.003658	-0.005048	0.180986	0.181823	0.032856	-0.086888	0.172728	0.314879	0.310187
season	0.029368	0.480021	1.000000	-0.008126	0.008879	0.258689	0.264744	0.190610	-0.147121	0.096758	0.164011	0.163439
workingday	-0.250491	-0.003658	-0.008126	1.000000	0.033772	0.029966	0.024660	-0.010880	0.013373	-0.319111	0.119460	0.011594
weather	-0.007074	-0.005048	0.008879	0.033772	1.000000	-0.055035	-0.055376	0.406244	0.007261	-0.135918	-0.109340	-0.128655
temp	0.000295	0.180986	0.258689	0.029966	-0.055035	1.000000	0.984948	-0.064949	-0.017852	0.467097	0.318571	0.394454
atemp	-0.005215	0.181823	0.264744	0.024660	-0.055376	0.984948	1.000000	-0.043536	-0.057473	0.462067	0.314635	0.389784
humidity	0.001929	0.032856	0.190610	-0.010880	0.406244	-0.064949	-0.043536	1.000000	-0.318607	-0.348187	-0.265458	-0.317371
windspeed	0.008409	-0.086888	-0.147121	0.013373	0.007261	-0.017852	-0.057473	-0.318607	1.000000	0.092276	0.091052	0.101369
casual	0.043799	0.172728	0.096758	-0.319111	-0.135918	0.467097	0.462067	-0.348187	0.092276	1.000000	0.497250	0.690414
registered	-0.020956	0.314879	0.164011	0.119460	-0.109340	0.318571	0.314635	-0.265458	0.091052	0.497250	1.000000	0.970948
Totalcount	-0.005393	0.310187	0.163439	0.011594	-0.128655	0.394454	0.389784	-0.317371	0.101369	0.690414	0.970948	1.000000

```
plt.figure(figsize=(10,10))
sns.heatmap(df[['holiday','datetime','season','workingday','weather','temp','atemp','humidity','windspeed','casual','registered','Totalcount']].corr(),
```

```
<Axes: >
```



Insights:

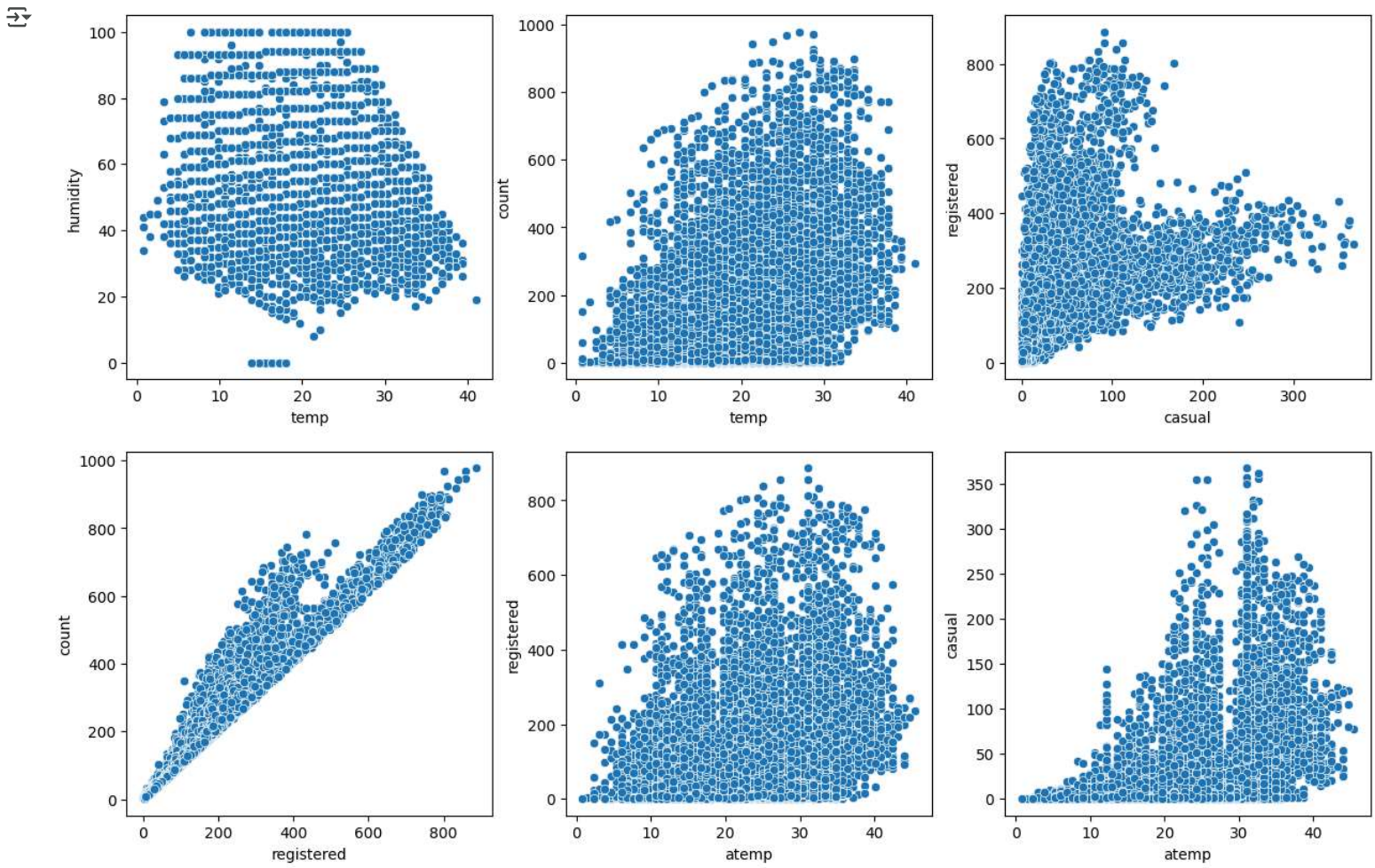
Casual,registered and count column are highly positively corelated.

atemp,total_count are slight postively corelated

Scatter Plot

```
plt.figure(figsize=(15,15))

plt.subplot(3,3,1)
sns.scatterplot(x=df_copy['temp'],y=df_copy['humidity'])
```

```
plt.subplot(3,3,2)
sns.scatterplot(x=df_copy['temp'],y=df_copy['count'])

plt.subplot(3,3,3)
sns.scatterplot(x=df_copy['casual'],y=df_copy['registered'])

plt.subplot(3,3,4)
sns.scatterplot(x=df_copy['registered'],y=df_copy['count'])

plt.subplot(3,3,5)
sns.scatterplot(x=df_copy['atemp'],y=df['registered'])

plt.subplot(3,3,6)
sns.scatterplot(x=df_copy['atemp'],y=df['casual'])

plt.show()
```



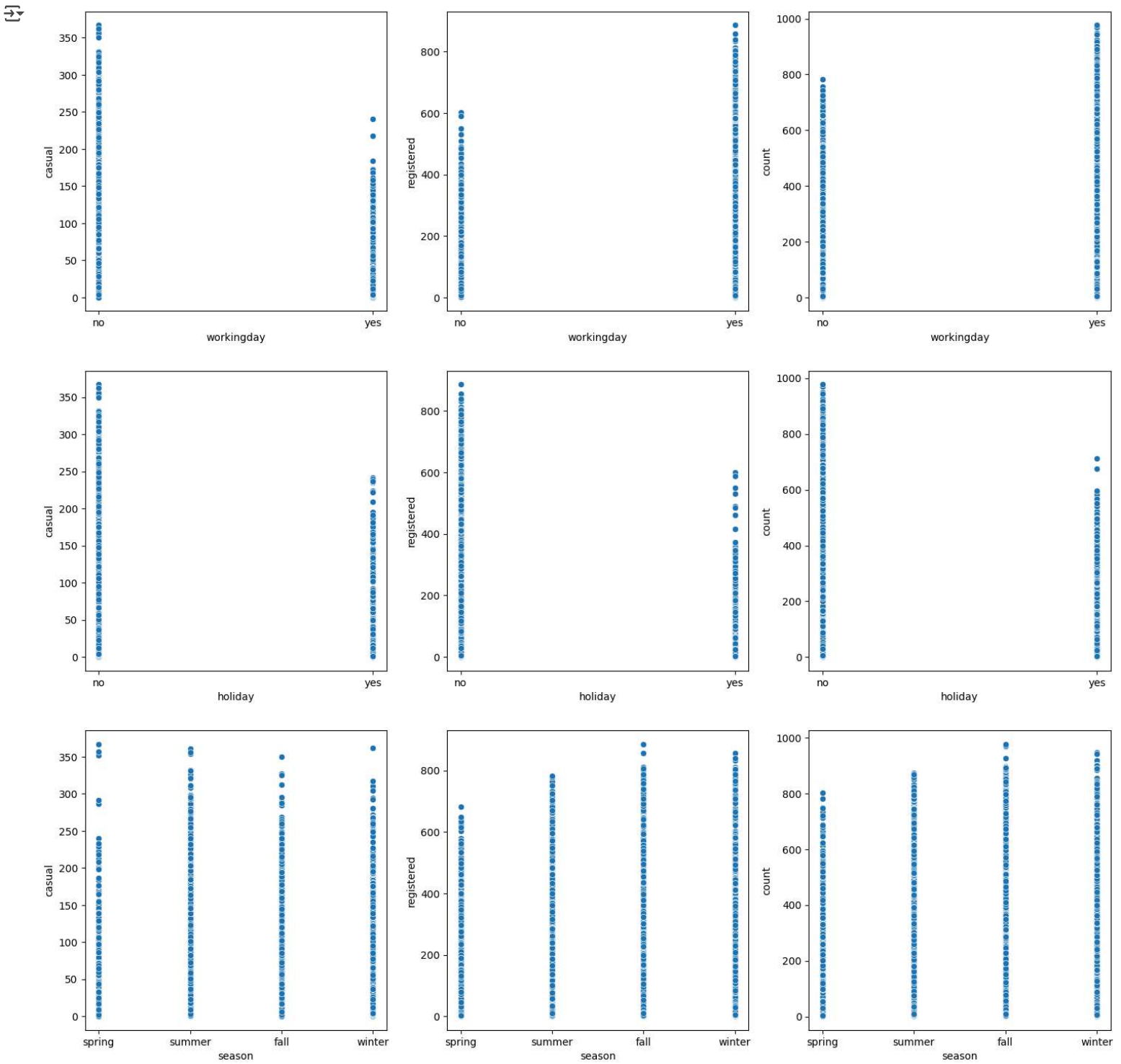
Insights

Count, registered columns are slighlt postiviely corelated with atemp.

Casual column is postively corelated with atem that means average casual customers of bike rides increases when atemp rises

```
plt.figure(figsize=(18,18))

plt.subplot(3,3,1)
sns.scatterplot(x=df_copy['workingday'],y=df_copy['casual'])

plt.subplot(3,3,2)
sns.scatterplot(x=df_copy['workingday'],y=df_copy['registered'])

plt.subplot(3,3,3)
sns.scatterplot(x=df_copy['workingday'],y=df_copy['count'])

plt.subplot(3,3,4)
sns.scatterplot(x=df_copy['holiday'],y=df_copy['casual'])

plt.subplot(3,3,5)
sns.scatterplot(x=df_copy['holiday'],y=df_copy['registered'])

plt.subplot(3,3,6)
sns.scatterplot(x=df_copy['holiday'],y=df_copy['count'])
```

```
plt.subplot(3,3,7)
sns.scatterplot(x=df_copy['season'],y=df_copy['casual'])

plt.subplot(3,3,8)
sns.scatterplot(x=df_copy['season'],y=df_copy['registered'])

plt.subplot(3,3,9)
sns.scatterplot(x=df_copy['season'],y=df_copy['count'])

plt.show()
```

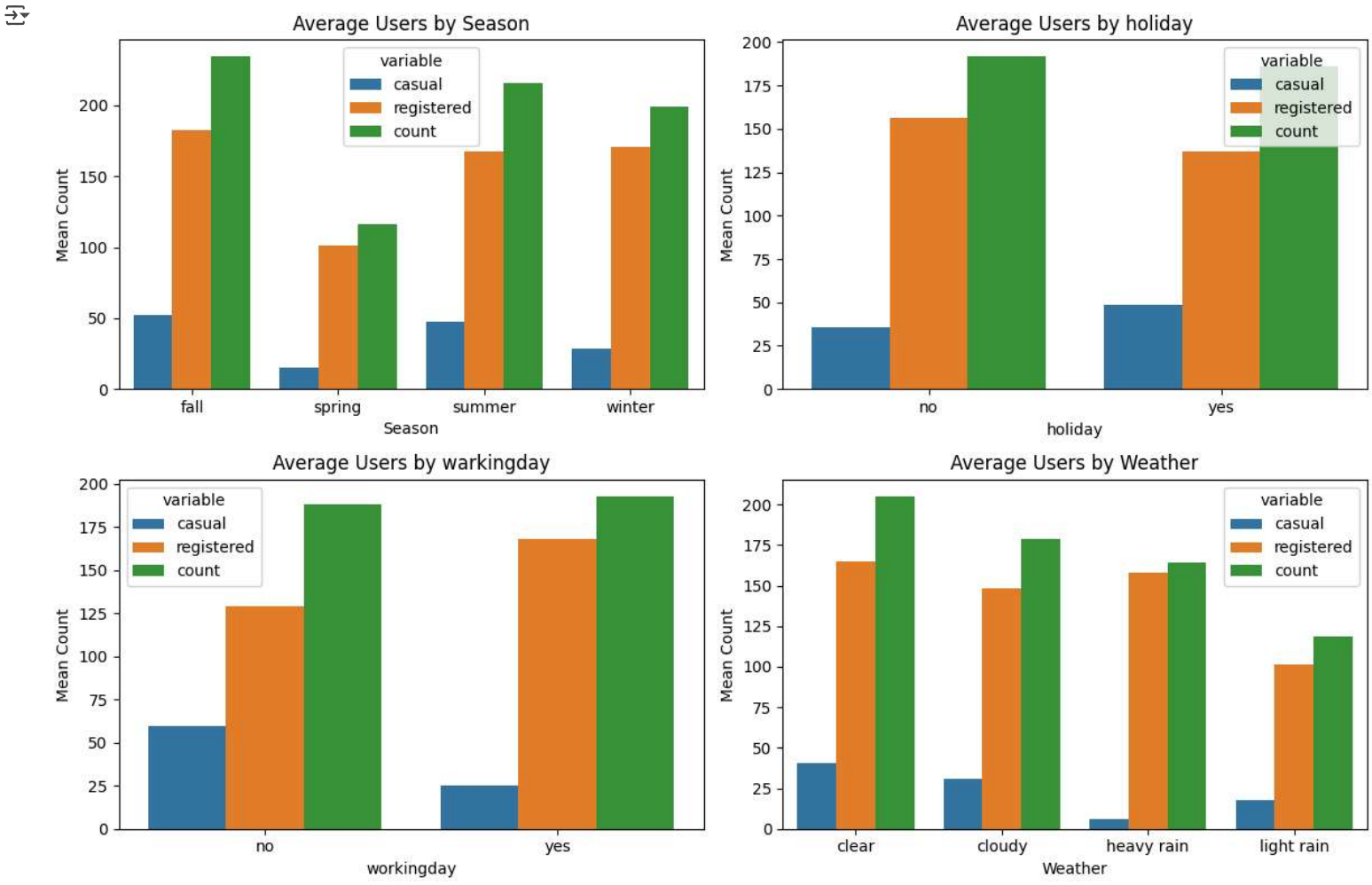


Insights:

- Casual customers prefer renting the bikes on weekend
- Registered customers prefer renting the bikes on working day
- Total number of bike renting is on Non holiday
- Casual, registered customers are distributed equally in seasons

Bivarite Analysis

```
fig,axes=plt.subplots(2,2,figsize=(12,8))
df_grouped_season=df_copy.groupby(['season'])[['casual','registered','count']].mean().reset_index()
df_melt_season=pd.melt(df_grouped_season,id_vars=['season'],value_vars=['casual','registered','count'])
sns.barplot(x='season',y='value',hue='variable',data=df_melt_season,ax=axes[0,0])
axes[0,0].set_title('Average Users by Season')
axes[0,0].set_xlabel('Season')
axes[0,0].set_ylabel('Mean Count')


df_grouped_holiday=df_copy.groupby(['holiday'])[['casual','registered','count']].mean().reset_index()
df_melt_holiday=pd.melt(df_grouped_holiday,id_vars=['holiday'],value_vars=['casual','registered','count'])
sns.barplot(x='holiday',y='value',hue='variable',data=df_melt_holiday,ax=axes[0,1])
axes[0,1].set_title('Average Users by holiday')
axes[0,1].set_xlabel('holiday')
axes[0,1].set_ylabel('Mean Count')

df_grouped_workingday=df_copy.groupby(['workingday'])[['casual','registered','count']].mean().reset_index()
df_melt_workingday=pd.melt(df_grouped_workingday,id_vars=['workingday'],value_vars=['casual','registered','count'])
sns.barplot(x='workingday',y='value',hue='variable',data=df_melt_workingday,ax=axes[1,0])
axes[1,0].set_title('Average Users by warkingday')
axes[1,0].set_xlabel('workingday')
axes[1,0].set_ylabel('Mean Count')

df_groupby_weather=df_copy.groupby(['weather'])[['casual','registered','count']].mean().reset_index()
df_melt_weather=pd.melt(df_groupby_weather,id_vars=['weather'],value_vars=['casual','registered','count'])
sns.barplot(x='weather',y='value',hue='variable',data=df_melt_weather,ax=axes[1,1])
axes[1,1].set_title('Average Users by Weather')
axes[1,1].set_xlabel('Weather')
axes[1,1].set_ylabel('Mean Count')

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



Insights

Average number of bike rents are more in fall season compare to other season for every user_type.

Average number of bike rents are more in working day for registered users where as it is lesser for casual users.

Average number of bike rents are more when sky is clear for every user type.

```
fig,axes=plt.subplots(3,1,figsize=(12,8))

df_datetime_year=df_copy.groupby([df_copy['datetime'].dt.year])[['casual','registered','count']].mean().reset_index()
df_melt_year=pd.melt(df_datetime_year,id_vars=['datetime'],value_vars=['casual','registered','count'])
sns.barplot(x='datetime',y='value',hue='variable',data=df_melt_year,ax=axes[0])
axes[0].set_title('Average Users by Year')
axes[0].set_xlabel('Year')
axes[0].set_ylabel('Mean Count')

df_datetime_month=df_copy.groupby([df_copy['datetime'].dt.month])[['casual','registered','count']].mean().reset_index()
df_melt_month=pd.melt(df_datetime_month,id_vars=['datetime'],value_vars=['casual','registered','count'])
sns.barplot(x='datetime',y='value',hue='variable',data=df_melt_month,ax=axes[1])
axes[1].set_title('Average Users by Month')
axes[1].set_xlabel('Month')
axes[1].set_ylabel('Mean Count')

df_datetime_weekday=df_copy.groupby([df_copy['datetime'].dt.weekday])[['casual','registered','count']].mean().reset_index()
df_melt_weekday=pd.melt(df_datetime_weekday,id_vars=['datetime'],value_vars=['casual','registered','count'])
sns.barplot(x='datetime',y='value',hue='variable',data=df_melt_weekday,ax=axes[2])
axes[2].set_title('Average Users by Weekday')
axes[2].set_xlabel('Weekday')
axes[2].set_ylabel('Mean Count')

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

Insights

Average number of bike rents have increased in 2012 every user_type.

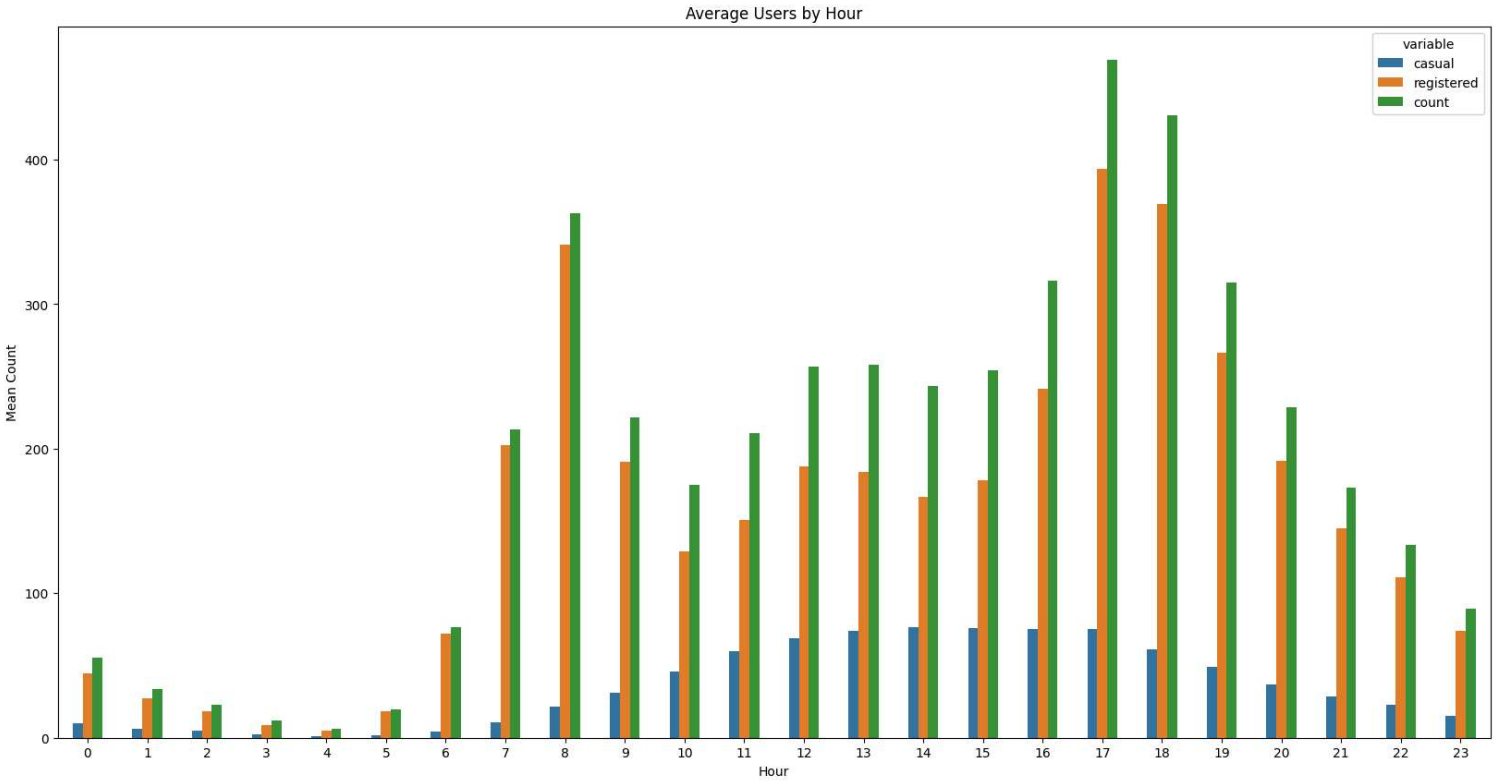
Average number of bike rents are more in the month of may-oct compare other i.e customers prefer riding this in summer, fall season

Casual customers are more in weekend but where registered customers are less.

Average total number of bikes rented on weekend i.e 5th and 6th day are more higher.

```
plt.figure(figsize=(20,10))
df_datetime_hour=df_copy.groupby([df_copy['datetime'].dt.hour])[['casual','registered','count']].mean().reset_index()
df_meltted_hour=pd.melt(df_datetime_hour,id_vars=['datetime'],value_vars=['casual','registered','count'])
sns.barplot(x='datetime',y='value',hue='variable',data=df_meltted_hour,width=0.5)
plt.xlabel('Hour')
plt.ylabel('Mean Count')
plt.title('Average Users by Hour')
```

```
Text(0.5, 1.0, 'Average Users by Hour')
```



Insights

Average number of bike rents are more around 5pm,6pm during evening.

Average number of bike rents are more around 8am in the morning.

During midnight negligible number of bike rented.

Casual customers are using more bikes int he evening only

```
pd.DataFrame(df_copy.groupby(['workingday'])['count'].describe())
```

workingday	count	mean	std	min	25%	50%	75%	max
no	3474.0	188.506621	173.724015	1.0	44.0	128.0	304.0	783.0
yes	7412.0	193.011873	184.513659	1.0	41.0	151.0	277.0	977.0

Insights:

Statistical mean of bike rented are almost same for working day and non working day

```
pd.DataFrame(df_copy.groupby(['season'])['count'].describe())
```

season	count	mean	std	min	25%	50%	75%	max
fall	2733.0	234.417124	197.151001	1.0	68.0	195.0	347.0	977.0
spring	2686.0	116.343261	125.273974	1.0	24.0	78.0	164.0	801.0
summer	2733.0	215.251372	192.007843	1.0	49.0	172.0	321.0	873.0
winter	2734.0	198.988296	177.622409	1.0	51.0	161.0	294.0	948.0

Insights

Statistical mean of bike rented are more in fall season compare to other but we should do hypothesis testing on this to have a strong evidence.

Demand of bike is lowest in spring time.

```
pd.DataFrame(df_copy.groupby(['weather'])['count'].describe())
```

weather	count	mean	std	min	25%	50%	75%	max
clear	7192.0	205.236791	187.959566	1.0	48.0	161.0	305.0	977.0
cloudy	2834.0	178.955540	168.366413	1.0	41.0	134.0	264.0	890.0
heavy rain	1.0	164.000000	NaN	164.0	164.0	164.0	164.0	164.0
light rain	859.0	118.846333	138.581297	1.0	23.0	71.0	161.0	891.0

Insights:

Avg number of bike rented are more when weather is clear compare to other but we should do hypothesis testing on this to have a strong evidence.

```
pd.DataFrame(df_copy.groupby(['holiday'])['count'].describe())
```

holiday	count	mean	std	min	25%	50%	75%	max
no	10575.0	191.741655	181.513131	1.0	43.0	145.0	283.0	977.0
yes	311.0	185.877814	168.300531	1.0	38.5	133.0	308.0	712.0

Insights

Avg number of bike rented are almost same on holiday and non holiday.

```
pd.DataFrame(df_copy.groupby(df_copy['datetime'].dt.year)['count'].describe())
```

datetime	count	mean	std	min	25%	50%	75%	max
2011	5422.0	144.223349	133.312123	1.0	32.0	111.0	210.0	638.0
2012	5464.0	238.560944	208.114003	1.0	59.0	199.0	354.0	977.0

```
pd.DataFrame(df_copy.groupby(df_copy['datetime'].dt.year)['casual'].describe())
```

datetime	count	mean	std	min	25%	50%	75%	max
2011	5422.0	28.73792	39.554419	0.0	3.0	13.0	38.0	272.0
2012	5464.0	43.25000	57.584101	0.0	5.0	20.0	61.0	367.0

```
pd.DataFrame(df_copy.groupby(df_copy['datetime'].dt.year)['registered'].describe())
```

datetime	count	mean	std	min	25%	50%	75%	max
2011	5422.0	115.485430	108.847868	0.0	27.0	91.0	168.0	567.0
2012	5464.0	195.310944	174.709050	1.0	51.0	161.0	281.0	886.0

Insights:

Will conduct Hypothesis testing to have a strong evidence

Hypothesis Testing

```
from scipy import stats
from scipy.stats import ttest_ind
from scipy.stats import f_oneway
from scipy.stats import shapiro
from scipy.stats import levene
from scipy.stats import chi2_contingency
```

Q1-Is there a statistically significant difference in the average number of bikes rented on non-working days compared to working days?

H0 is Null Hypothesis

H1 is Alternate Hypothesis

H0: The demand of bikes in Non working day is less or same when compared to Working day

H1: The demand of bikes in Non working day is more when compared to working days

Let μ1 and μ2 be the average no of bikes rented on Working day and Non working day.

H0:μ1>=μ2 H1:μ1<μ2

This is an example of two sample Ttest as population mean and deviation is unknown.

```
df_copy.groupby(['workingday'])['count'].describe()
```

workingday	count	mean	std	min	25%	50%	75%	max
no	3474.0	188.506621	173.724015	1.0	44.0	128.0	304.0	783.0
yes	7412.0	193.011873	184.513659	1.0	41.0	151.0	277.0	977.0

```
def result(p_value,alpha):
  if p_value>alpha:
    print('Accept Null Hypothesis')
  else:
    print('Reject Null Hypothesis')


alpha=0.05
```

Sample size of both the variable are different. so for accuracy in test we have to take same sample size.

```
workingday=df_copy[df_copy['workingday']=='yes']['count'].sample(3000)
non_workingday=df_copy[df_copy['workingday']=='no']['count'].sample(3000)
```

To check whether for both the group variance is different or not , we will do Levene test

H0: All the count variances are equal

H1: At least one variance is different from the rest

```
stat, p_value = levene(workingday, non_workingday)
print(result(p_value,.05),'The p-value is : ', p_value,)
```

```
Accept Null Hypothesis
None The p-value is :  0.3135230715243583
```

```
ttest,p_value=stats.ttest_ind(workingday,non_workingday,alternative='less')
print('The p-value is : ', p_value,)
result(p_value,alpha)
```

```
The p-value is :  0.7267530145798322
Accept Null Hypothesis
```

Observation:

Since the p-value is greater than the 5% significance level, we fail to reject the null hypothesis.

Hence, we have enough statistical evidence to say that the demand of bikes on Non working day is < or = to the demand of bikes on Working day

There is no significant difference in average no. of rented bikes for working day and non working day.

Q2-Is there a statistically significant difference in the average number of bike rides on holidays compared to regular (non-holiday) days?

H0: The demand of bikes on holidays is less or equal when compared to regular days.

H1: The demand of bikes on holidays is more when compared to regular days.

Let μ1 and μ2 be the average no. of bikes rented on regular days and holidays respectively.

H0:μ1>=μ2

H1:μ1<μ2

```
df_copy.groupby(['holiday'])['count'].describe()
```

holiday	count	mean	std	min	25%	50%	75%	max
no	10575.0	191.741655	181.513131	1.0	43.0	145.0	283.0	977.0
yes	311.0	185.877814	168.300531	1.0	38.5	133.0	308.0	712.0

```
holiday=df_copy[df_copy['holiday']=='yes']['count'].sample(250)
regular=df_copy[df_copy['holiday']=='no']['count'].sample(250)
```

To check whether for both the group variance is different or not , we will do Levene test

H0: All the count variances are equal

H1: At least one variance is different from the rest

```
stat,p_value=levene(holiday,regular)
print('The p-value is : ', p_value)
result(p_value,.05)
```

```
The p-value is :  0.7828790331340838
Accept Null Hypothesis
```

```
ttest,p_value=ttest_ind(holiday,regular,alternative='less')
print('The p-value is : ', p_value)
result(p_value,alpha)
```

```
The p-value is :  0.5136959324182044
Accept Null Hypothesis
```

Observation:

Since the p-value is greater than the 5% significance level, we fail to reject the null hypothesis.

Hence, we don't have enough statistical evidence to say that the demand of bikes on Holidays is greater than regular days

Q3-| Does weather condition have a significant impact on the demand for bike rentals?

H0: The demand of bikes are not impacted by weather conditions.

H1: The demand of bikes are imoacted by weather conditions

```
df_copy.groupby(['weather'])['count'].describe()
```

weather	count	mean	std	min	25%	50%	75%	max
clear	7192.0	205.236791	187.959566	1.0	48.0	161.0	305.0	977.0
cloudy	2834.0	178.955540	168.366413	1.0	41.0	134.0	264.0	890.0
heavy rain	1.0	164.000000	NaN	164.0	164.0	164.0	164.0	164.0
light rain	859.0	118.846333	138.581297	1.0	23.0	71.0	161.0	891.0

Sample size of the variable are different, so for accuracy in test we have to take same sample size.

```
Clear=df_copy[df_copy['weather']=='clear']['count'].sample(750)
Cloudy=df_copy[df_copy['weather']=='cloudy']['count'].sample(750)
light_rain=df_copy[df_copy['weather']=='light rain']['count'].sample(750)
```

This is a problem, concerning three independent population means. **One-way ANOVA** could be the appropriate test here provided normality and equality of variance assumptions are verified.

The ANOVA test has important assumptions that must be satisfied in order for the associated p-value to be valid.

- The samples are independent.
- Each sample is from a normally distributed population.
- The population variance of the groups are all equal.

Now, we will be using the following statistical tests to check the normality and euality of variance of the data set -

For testing of normality, Shapiro-Wilk's test is applied to the response variable.

For equality of variance, Levene test is applied to the response variable.

Shapiro-Wilk's test

H0: count follows normal distribution

Ha: count does not follows normal distribution

```
test_stat,p_value=shapiro(df_copy['count'].sample(4999))
print('The p-value is : ', p_value)
result(p_value,.05)
```

```
The p-value is :  1.6932739235502307e-52
Reject Null Hypothesis
```

Null Hypothesis is rejected, Data is not normal

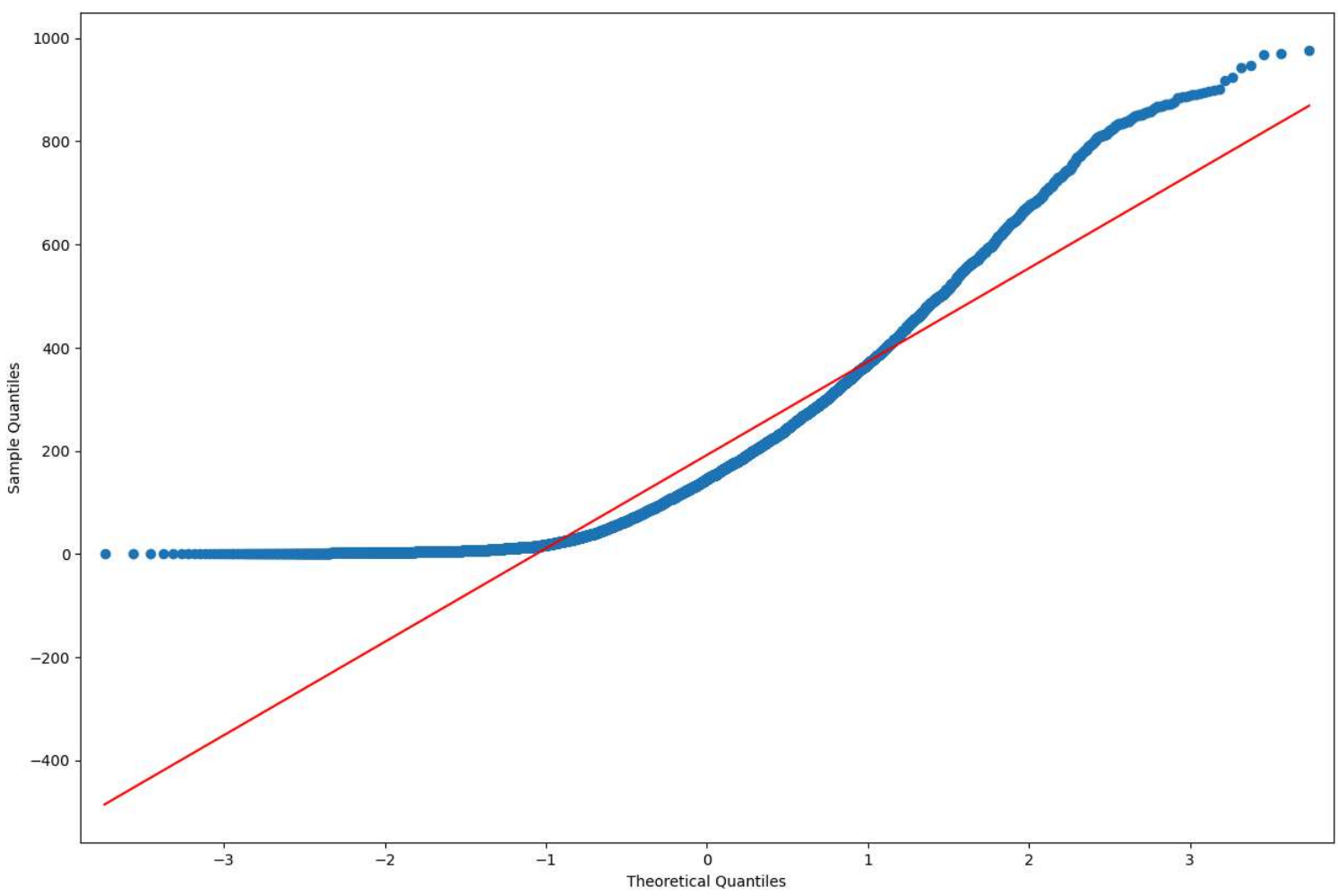
```
df_copy['count'].skew()
```

```
np.float64(1.2420662117180776)
```

```
df_copy['count'].kurt()
```

```
np.float64(1.3000929518398334)
```

```
from statsmodels.graphics.gofplots import qqplot
qqplot(df_copy['count'],line='s')
plt.show()
```



Insights:

qqplots suggests the data is not normal, it is positively skewed

Levene test

To check whether for both the group variance is different or not , we will do Levene test

H0: Variances are equal among groups(Clear,cloudy,Rain)

H1: Variances are not equal among groups

```
stat,p_value=levene(Clear,Cloudy,light_rain)
print('The p-value is : ', p_value)
result(p_value,.05)
```

```
The p-value is :  3.401985010940654e-16
Reject Null Hypothesis
```

Assumptions of Anova fails for the weather category

Kruskal Wallis Test

```
from scipy.stats import kruskal

stats,p_value=kruskal(Clear,Cloudy,light_rain)
print('Test_statistics:', stats)
print('The p-value is : ', p_value)
result(p_value,alpha)
```

```
Test_statistics: 103.0414164513174
The p-value is :  4.2154186619692606e-23
Reject Null Hypothesis
```

Anova Test

```
f_stats,p_value=f_oneway(Clear,Cloudy,light_rain)
print('Test_statistics:', f_stats)
print('The p-value is : ', p_value)
result(p_value,alpha)
```

```
Test_statistics: 50.758449954366846
The p-value is :  2.7500555948912555e-22
Reject Null Hypothesis
```

Since the p_value is lesser than the alpha, we can reject the null hypothesis.

That means, The demand of bikes will be impacted by weather conditions

Q3a -| Does the 'clear' weather condition result in higher bike rental demand compared to 'cloudy' weather conditions?

H0: Average no of bike rented is same on both clear and cloudy weather conditions

H1: Average no of bike rented is more in clear weather condition when compared to cloudy weather condition

```
test_stat,p_value=ttest_ind(Clear,Cloudy,alternative='greater')
print('The p-value is : ', p_value)
result(p_value,alpha)
```

```
The p-value is :  0.0004279160378077581
Reject Null Hypothesis
```

There is significant evidence to say that Average no of bike rented is more in clear condition when compared to cloudy weather condition

Q3b-| Is the demand for bikes significantly lower on 'rainy' days compared to 'cloudy' days?

H0: Average no of bikes is same on both rainy days and cloudy days

H1: Average no of bikes is lower on rainy days compared to cloudy days

```
test_stat,p_value=ttest_ind(light_rain,Cloudy,alternative='less')
print('The p-value is : ', p_value)
result(p_value,alpha)
```

```
The p-value is :  3.2182726215560017e-12
Reject Null Hypothesis
```

There is Significant evidence to say that Average no of bikes is lower on rainy days compared to cloudy days

Q4-| Is the demand for bicycle rentals the same across different seasons?

```
df_copy.groupby(['season'])['count'].describe()
```

season	count	mean	std	min	25%	50%	75%	max
fall	2733.0	234.417124	197.151001	1.0	68.0	195.0	347.0	977.0
spring	2686.0	116.343261	125.273974	1.0	24.0	78.0	164.0	801.0
summer	2733.0	215.251372	192.007843	1.0	49.0	172.0	321.0	873.0
winter	2734.0	198.988296	177.622409	1.0	51.0	161.0	294.0	948.0

H0: The demand of bikes is not impacted by SEASON i.e average no. of bike rented is not impacted by season

H1: The demand of bikes is impacted by SEASON i.e average no. of bike rented is impacted by season

Sample size of both the variable are different, so for accuracy in test we have to take same sample size.

```
Fall=df_copy[df_copy['season']=='fall']['count'].sample(1500)
Spring=df_copy[df_copy['season']=='spring']['count'].sample(1500)
Summer=df_copy[df_copy['season']=='summer']['count'].sample(1500)
Winters=df_copy[df_copy['season']=='winter']['count'].sample(1500)
```

There are four independent population means. One-way ANOVA could be the appropriate test here provided normality and equality of variance assumptions are verified.

The ANOVA test has important assumptions that must be satisfied in order for the associated p-value to be valid.

- The samples are independent.
- Each sample is from a normally distributed population.
- The population variance of the groups are all equal.

Now, we will be using the following statistical tests to check the normality and equality of variance of the data set -

For testing of normality, Shapiro-Wilk's test is applied to the response variable.

For equality of variance, Levene test is applied to the response variable.

Shapiro-wilks Test

H0: Total_count follows normal distribution

Ha: Total_count does not follows normal distribution

```
stats,p_value=shapiro(df_copy['count'].sample(1500))
print('The p-value is : ', p_value)
result(p_value,.05)
```

```
The p-value is :  7.281066507018742e-33
Reject Null Hypothesis
```

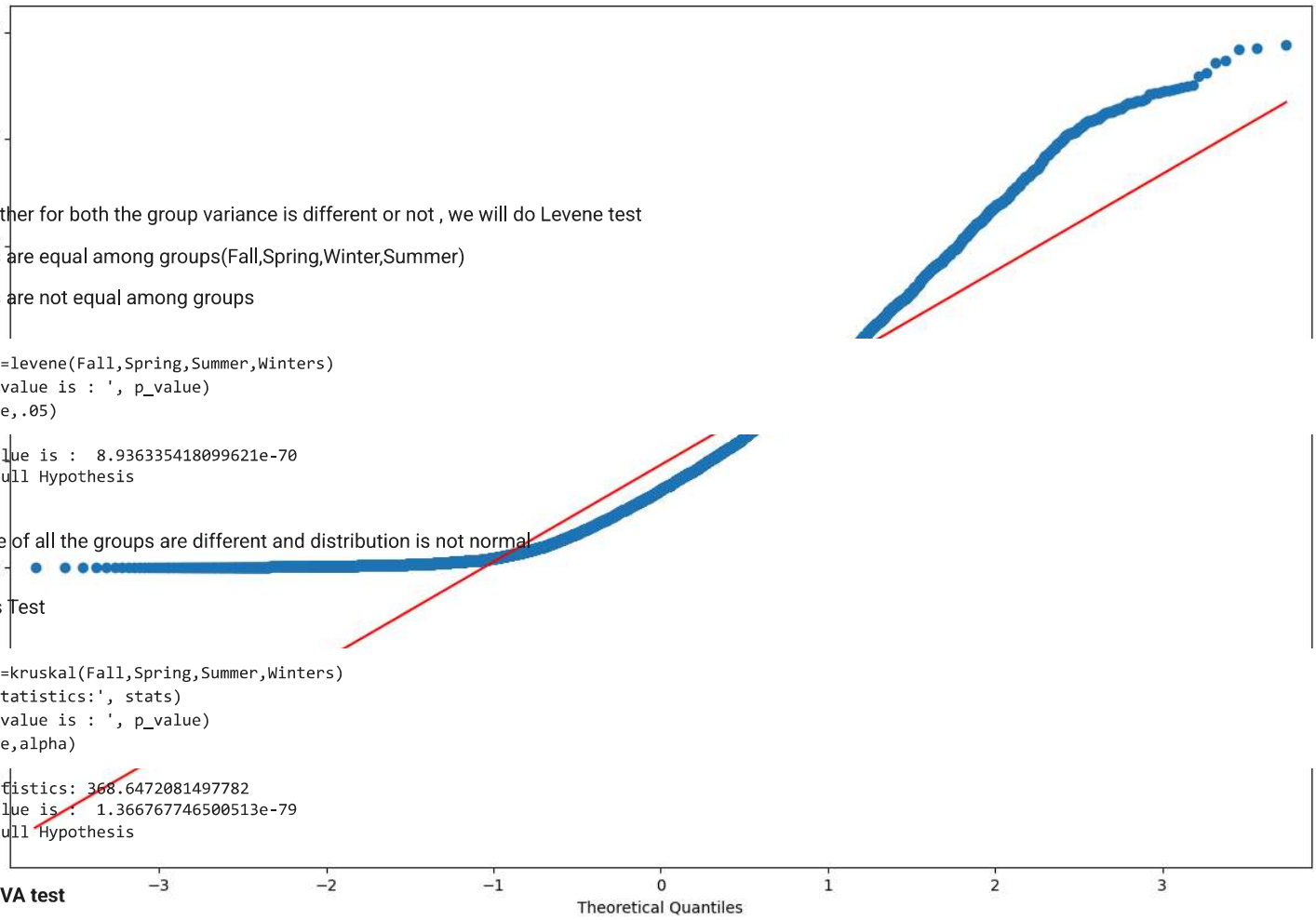
```
df_copy['count'].skew()
```

```
np.float64(1.2420662117180776)
```

```
df_copy['count'].kurt()
```

```
np.float64(1.3000929518398334)
```

```
from statsmodels.graphics.gofplots import qqplot
qqplot(df_copy['count'],line='s')
plt.show()
```



Levene test

To check whether for both the group variance is different or not , we will do Levene test

H0: Variances are equal among groups(Fall,Spring,Winter,Summer)

H1: Variances are not equal among groups

```
stats,p_value=levene(Fall,Spring,Summer,Winters)
print('The p-value is : ', p_value)
result(p_value,.05)
```

```
The p-value is :  8.936335418099621e-70
Reject Null Hypothesis
```

Since variance of all the groups are different and distribution is not normal

Kruskal Wallis Test

```
stats,p_value=kruskal(Fall,Spring,Summer,Winters)
print('Test_statistics:', stats)
print('The p-value is : ', p_value)
result(p_value,alpha)
```

```
Test_statistics: 368.6472081497782
The p-value is :  1.366767746500513e-79
Reject Null Hypothesis
```

One way ANOVA test

```
f_stats,p_value=f_oneway(Fall,Spring,Summer,Winters)
print('Test_statistics:', f_stats)
print('The p-value is : ', p_value)
result(p_value,alpha)
```

```
Test_statistics: 129.44203437682455
The p-value is :  3.0079647579001096e-81
Reject Null Hypothesis
```

Since, the p-value is less than 5% of significance level for both Kruskal and Anova Test.

Hence, we can say that The demand of bikes are impacted by different seasons

Q-4a-| Is the demand for bicycle rentals higher in the fall season compared to the winter?

H0: The demand of bikes is same in both Fall and Winter Seasons

H1: The demand of bikes is more in Fall season when compared to winter season

```
stats,p_value=ttest_ind(Fall,Winters,alternative='greater')
print('The p-value is : ', p_value)
result(p_value,alpha)
```

```
The p-value is :  2.477667356504498e-09
Reject Null Hypothesis
```

Since the Null Hypothesis is rejected.

We have Significant evidence to say that the demand of bikes is more in Fall season when compared to Winter season.

Q4b-| Is the demand for bicycle rentals higher in the fall season compared to the summer season?

H0: The demand of bikes is same in both Fall and Summer Season

H1: The demand of bikes is more in fall season when compared to summer season

```
[96] stats,p_value=ttest_ind(Fall,Summer,alternative='greater')
     print('The p-value is : ', p_value)
     result(p_value,alpha)
```

```
The p-value is :  0.039852381998129495
Reject Null Hypothesis
```

There is a significant evidence to say that The demand of bikes are more in Fall season when compared to Summer season

Q5-| Do weather conditions vary significantly across different seasons?

H0: Weather conditions are independent of season

H1: Weather conditions depend on season

```
[97] df_weath=df_copy[df_copy['weather']!='heavy rain']
```

```
[98] contigency_table=pd.crosstab(df_weath['season'],df_weath['weather'])
     contigency_table
```

weather	clear	cloudy	light rain
season			
fall	1930	604	199
spring	1759	715	211
summer	1801	708	224
winter	1702	807	225

Next steps: Generate code with contigency_table | View recommended plots | New interactive sheet

```
chi2,p_value,dof,expected=chi2_contingency(contigency_table)
print('Chi-square statistics: {} \n P_value: {} \n Degree of Freedom: {} \n Expected Frequencies: {}'.format(chi2,p_value,dof,np.round(expected,2)))
result(p_value,alpha)
```

```
Chi-square statistics: 46.10145731073249
 P_value: 2.8260014509929343e-08
 Degree of Freedom: 6
 Expected Frequencies: [[1805.76  711.56  215.68]
 [1774.05  699.06  211.89]
 [1805.76  711.56  215.68]
 [1806.42  711.82  215.76]]
Reject Null Hypothesis
```

There is significant eveidence to say weather conditions depend on season

Insights and Recommendations

EDA based insights:

1. Total number of registered customers who rents bike are more than casual.
2. Dataset is given from January 2011 to December 2012.
3. Total no. bike rented on Fall season are significantly more compared to other season.
4. Casual customers renting a bike are more on weekend.
5. Avg number of bikes rented by registered users are more on working day.
6. Distribution is right skewed distribution i.e. outliers are in the right tail end; this means there are occurrence of event where number of bikes rented have gone up.
7. Neither missing values, nor duplicate rows were found.
8. Temp and atemp columns were found to be positively corelated.
9. Casual and registered customers are positively corelated with atem column that means bike rides by these customers increases when atmospheric temp rises.
10. Total count,casual,registered column distribution are heavily right skewed.
11. Average number of bikes rented on holiday and non holiday are not having any different.

Insights from hypothesis testing:

1. The number of bikes rented on weekdays is comparatively higher than on weekends.
2. The number of bikes rented on regular (non-holiday) is almost same compared to holidays.
3. The demand of bikes on rent are impacted by weather condition.
4. The number of bikes rented more when weather was clear.
5. The number of bikes rented more during Fall season compared to winter and summer.
6. Weather conditions are significantly different during different seasons.

Recommendations:

1. The demand of bike rentals are higher during weekdays we can say that bikes are used by working professionals i.e. gig workers, office goers, delivery agents. so, company should target in terms of customer segment to improve their revenue.
2. Company should focus on ad campaigns to show how easier a transportation becomes after using bikes.
3. Bike renting gone up when the weather is clear that means company should increase the number of bikes at the station to cop up the demand.
4. Company should focus on repair and maintainence of the bikes at station during spring because demand is very low at that time.
5. Company should focus on making a station near public places like mall, food plaza places, iconic places where people spend more time on weekend so that more casual customers increase, and also ample amount of bike should be available in those spots in weekend.
6. Company should increase bike stocks in the weekend as there is strong demand.