yulu-hypothesis-testing-2

May 30, 2024

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import ttest_ind,ttest_rel
import random
from scipy.stats import f_oneway
from scipy.stats import chisquare,chi2,chi2_contingency
```

[2]:
```
!wget https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/428/
  ↪original/bike_sharing.csv?1642089089
```

```
--2024-05-30 04:22:43--  https://d2beiqkhq929f0.cloudfront.net/public_assets/ass
ets/000/001/428/original/bike_sharing.csv?1642089089
Resolving d2beiqkhq929f0.cloudfront.net (d2beiqkhq929f0.cloudfront.net)…
108.157.172.173, 108.157.172.10, 108.157.172.183, …
Connecting to d2beiqkhq929f0.cloudfront.net
(d2beiqkhq929f0.cloudfront.net)|108.157.172.173|:443… connected.
HTTP request sent, awaiting response… 200 OK
Length: 648353 (633K) [text/plain]
Saving to: ‘bike_sharing.csv?1642089089’

bike_sharing.csv?16 100%[===================>] 633.16K  --.-KB/s    in 0.1s

2024-05-30 04:22:43 (6.04 MB/s) - ‘bike_sharing.csv?1642089089’ saved
[648353/648353]
```

[3]:
```
df = pd.read_csv("bike_sharing.csv?1642089089")
df.tail()
```

[3]:

	datetime	season	holiday	workingday	weather	temp	\
10881	2012-12-19 19:00:00	4	0	1	1	15.58	
10882	2012-12-19 20:00:00	4	0	1	1	14.76	
10883	2012-12-19 21:00:00	4	0	1	1	13.94	
10884	2012-12-19 22:00:00	4	0	1	1	13.94	
10885	2012-12-19 23:00:00	4	0	1	1	13.12	

```
        atemp  humidity  windspeed  casual  registered  count
10881  19.695        50    26.0027       7         329    336
10882  17.425        57    15.0013      10         231    241
10883  15.910        61    15.0013       4         164    168
10884  17.425        61     6.0032      12         117    129
10885  16.665        66     8.9981       4          84     88
```

[4]:
```
df.head()
```

[4]:
```
             datetime  season  holiday  workingday  weather  temp   atemp  \
0  2011-01-01 00:00:00       1        0           0        1  9.84  14.395
1  2011-01-01 01:00:00       1        0           0        1  9.02  13.635
2  2011-01-01 02:00:00       1        0           0        1  9.02  13.635
3  2011-01-01 03:00:00       1        0           0        1  9.84  14.395
4  2011-01-01 04:00:00       1        0           0        1  9.84  14.395

   humidity  windspeed  casual  registered  count
0        81        0.0       3          13     16
1        80        0.0       8          32     40
2        80        0.0       5          27     32
3        75        0.0       3          10     13
4        75        0.0       0           1      1
```

#EDA

[5]:
```
df.shape
```

[5]:
```
(10886, 12)
```

[6]:
```
#Checking for info like Nulls and Dtypes

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
```

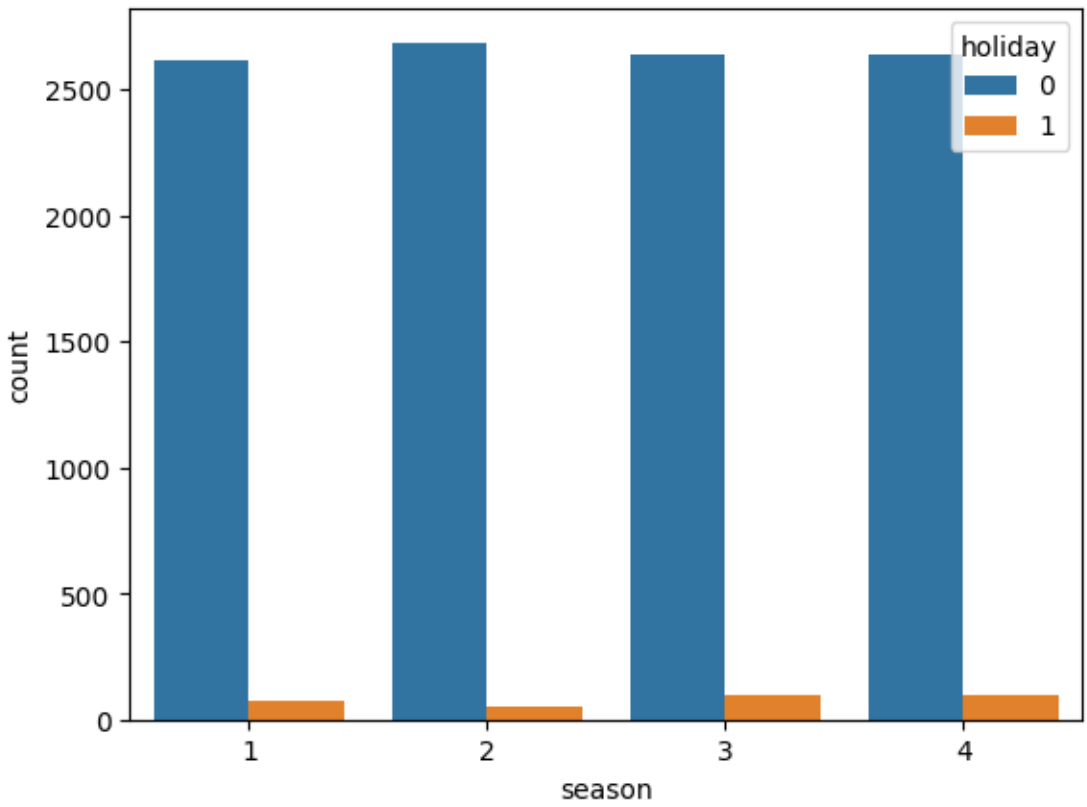
[7]:
```
df.describe()
```

[7]:

	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

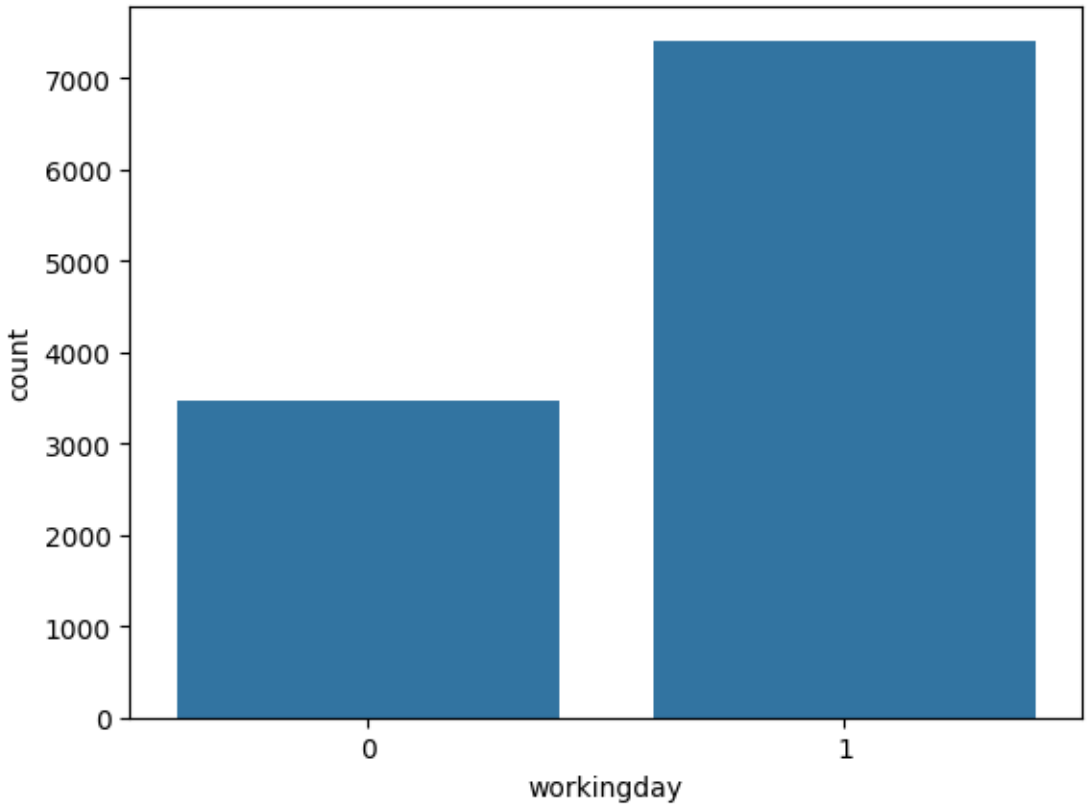
[8]:
```
# it is clear that booking are high on holidays.


sns.countplot(x=df['season'],hue='holiday',data=df)
```

[8]:
```
<Axes: xlabel='season', ylabel='count'>
```

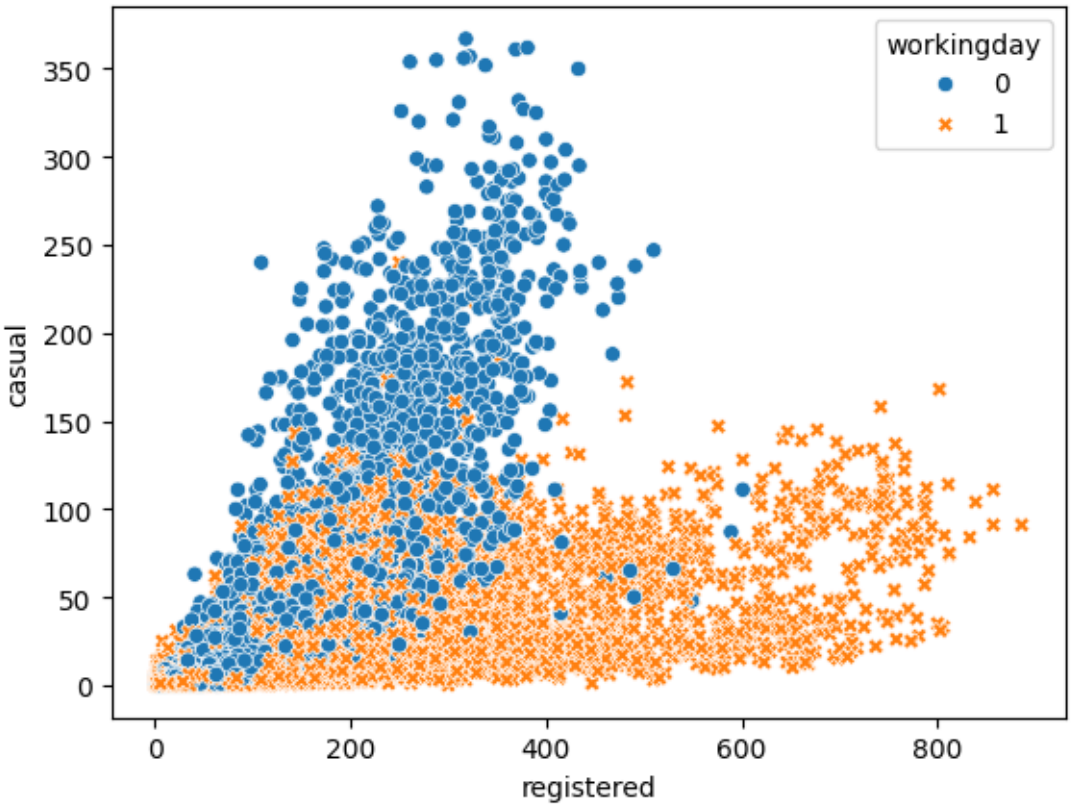


[9]:
```
sns.countplot(x=df['workingday'],data=df)
```

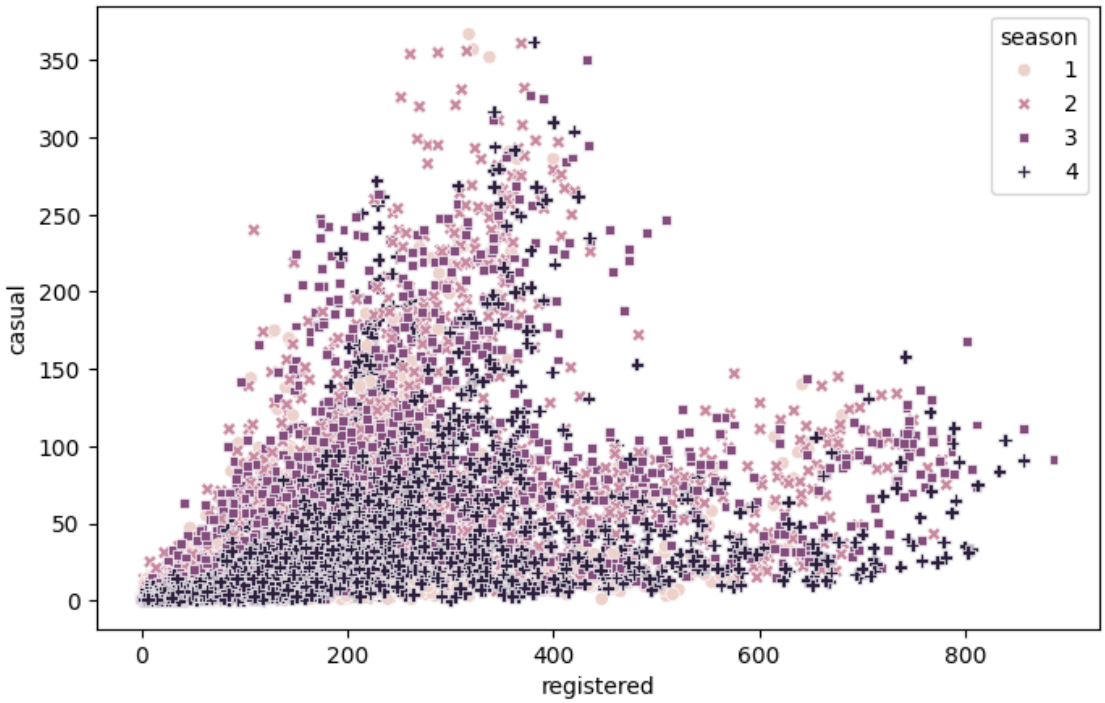
[9]:
```
<Axes: xlabel='workingday', ylabel='count'>
```



[10]:
```
colors = np.random.rand(2)
sns.
  ↪scatterplot(x=df['registered'],y=df['casual'],data=df,hue='workingday',style='workingday')
plt.show()
```



[11]:
```
#1: spring, 2: summer, 3: fall, 4: winter

colors = np.random.rand(2)
plt.figure(figsize=(8,5))
sns.
  ↪scatterplot(x=df['registered'],y=df['casual'],data=df,hue='season',style='season')
plt.show()
```



[11]:

#Hypothesis Testing

[12]:
```
df['workingday'].value_counts()
```

[12]:
```
workingday
1    7412
0    3474
Name: count, dtype: int64
```

[13]:
```
df.head()
```

[13]:

	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					
4	2011-01-01 04:00:00	1	0	0	1	9.84	14.395					

```
3        75        0.0       3          10     13
4        75        0.0       0           1      1
```

[14]:
```
# It is clear that registred users are highly using on the weekdays there is␣
 ↪huge difference

df.groupby('workingday')['registered'].sum()
```

[14]:
```
workingday
0     448835
1    1244506
Name: registered, dtype: int64
```

[15]:
```
# casula users are high on the holidays

df.groupby('workingday')['casual'].sum()
```

[15]:
```
workingday
0    206037
1    186098
Name: casual, dtype: int64
```

[16]:
```
df.groupby('workingday')['count'].sum()
```

[16]:
```
workingday
0     654872
1    1430604
Name: count, dtype: int64
```

[17]:
```
df_holidays = df[df['workingday']==0]
df_holidays.head()
```

[17]:
```
              datetime  season  holiday  workingday  weather  temp   atemp  \
0  2011-01-01 00:00:00       1        0           0        1  9.84  14.395
1  2011-01-01 01:00:00       1        0           0        1  9.02  13.635
2  2011-01-01 02:00:00       1        0           0        1  9.02  13.635
3  2011-01-01 03:00:00       1        0           0        1  9.84  14.395
4  2011-01-01 04:00:00       1        0           0        1  9.84  14.395

   humidity  windspeed  casual  registered  count
0        81        0.0       3          13     16
1        80        0.0       8          32     40
2        80        0.0       5          27     32
3        75        0.0       3          10     13
4        75        0.0       0           1      1
```

[18]:
```
df_working = df[df['workingday']==0]
df_working.head()
```

[18]:

	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

[19]:
```
from scipy.stats import ttest_ind,ttest_rel
import random
```

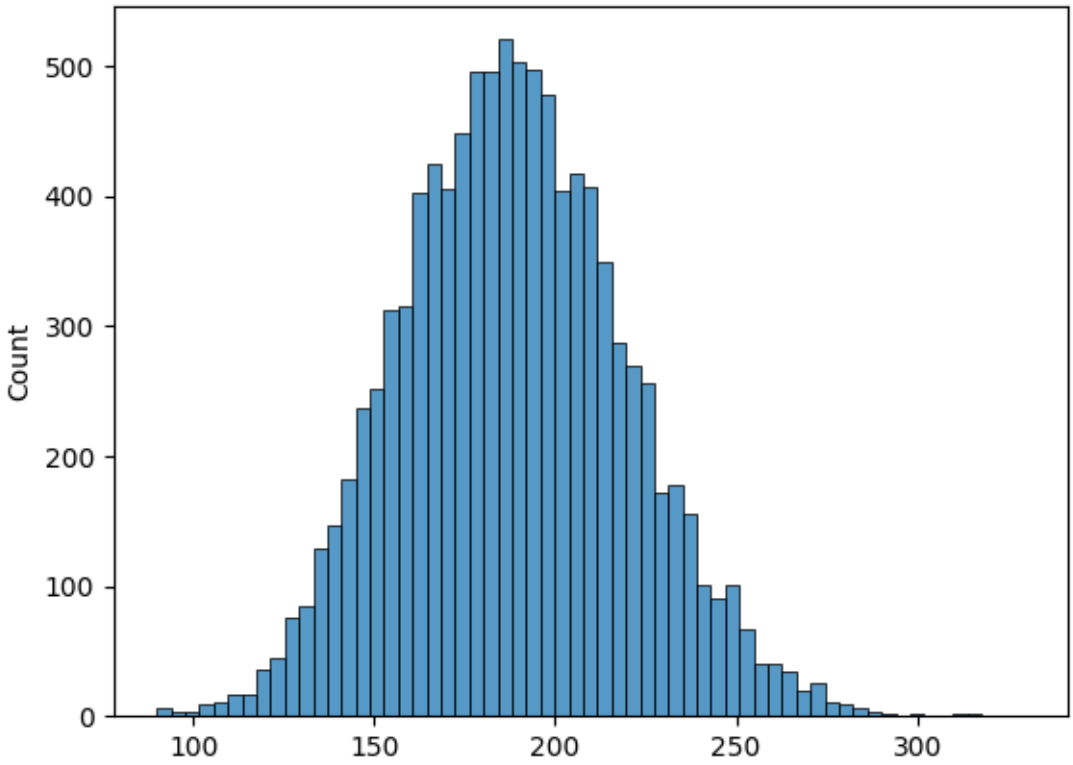
[20]:
```
df_working['count'].sample(5)
```

[20]:
```
10603      6
1174     304
5030     305
10225    504
9038     235
Name: count, dtype: int64
```

[21]:
```
# Normal distribusion
# using 30 samples and running the loop 10000 times

sample_working30 = [np.mean(df_working['count'].sample(30)) for i in␣
 ↪range(10000)]
```

[22]:
```
sns.histplot(sample_working30)
```

[22]:
```
<Axes: ylabel='Count'>
```



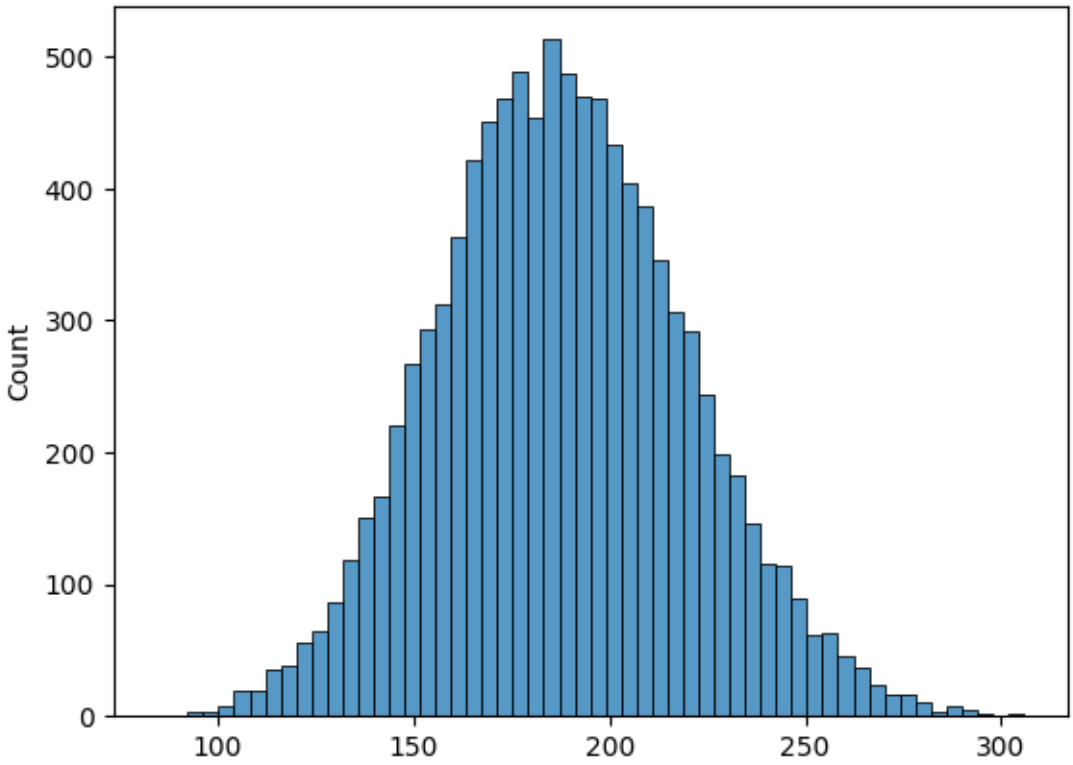
[23]:
```
df_holidays['count'].sample(5)
```

[23]:
```
3304    322
5803     28
6114     53
2750    404
1166     10
Name: count, dtype: int64
```

[65]:
```
# Normal distribusion
# using 30 samples and running the loop 10000 times

sample_holiday30 = [np.mean(df_holidays['count'].sample(30)) for i in␣
 ↪range(10000)]
```

[25]:
```
sns.histplot(sample_holiday30)
```

[25]:
```
<Axes: ylabel='Count'>
```



[26]:
```
print(df['count'].mean()),print(np.mean(sample_holiday30))
```

```
191.57413191254824
187.95442999999997
```

[26]: (None, None)

1 T-Test

[27]:
```
#H0: Workingday has no effect on the  number of electric cycles rented
#HA: Workingday has effect on the  number of electric cycles rented

#  1=2
#  1 2
```

[28]:
```
t_stat,pval = ttest_ind(sample_working30,sample_holiday30)
t_stat,pval
```

[28]: (1.0724813399883246, 0.2835168325773322)

[29]:
```
alpha = 0.05

if pval<alpha:
  print('Reject H0')
else:
  print('Fail to Reject H0')
```

```
Fail to Reject H0
```

#ANOVA

[30]:
```
from scipy.stats import f_oneway
```

[31]:
```
df.head()
```

[31]:

	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

[32]:
```
df['season'].value_counts()
```

[32]:
```
season
4    2734
2    2733
3    2733
1    2686
Name: count, dtype: int64
```

[33]:
```
#season: season (1: spring, 2: summer, 3: fall, 4: winter)

spring = df[df['season'] == 1]
```

[34]:
```
summer = df[df['season'] == 2]
```

[35]:
```
fall = df[df['season'] == 3]
```

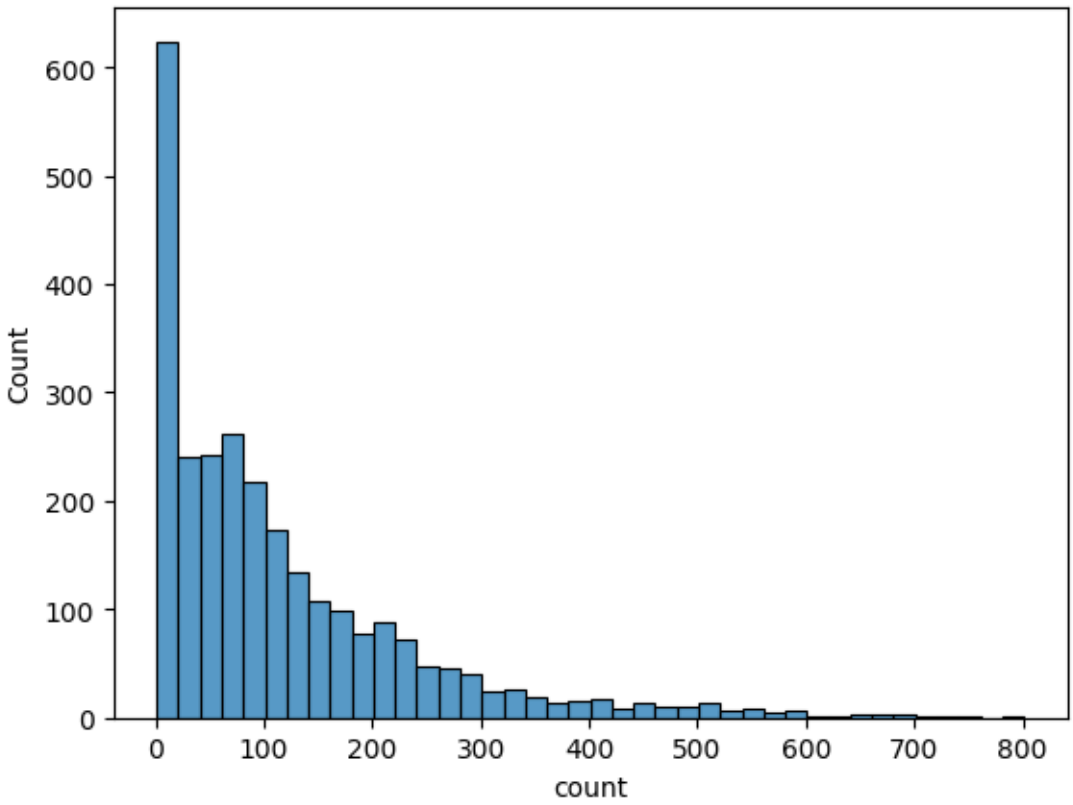
[36]:
```
winter = df[df['season'] == 4]
```

[37]:
```
spring['count'].sample(5)
```

[37]:
```
6598     16
6730    648
5891    283
6664     91
1055     99
Name: count, dtype: int64
```

[38]:
```
sns.histplot(data=spring,x='count')
```

[38]:
```
<Axes: xlabel='count', ylabel='Count'>
```

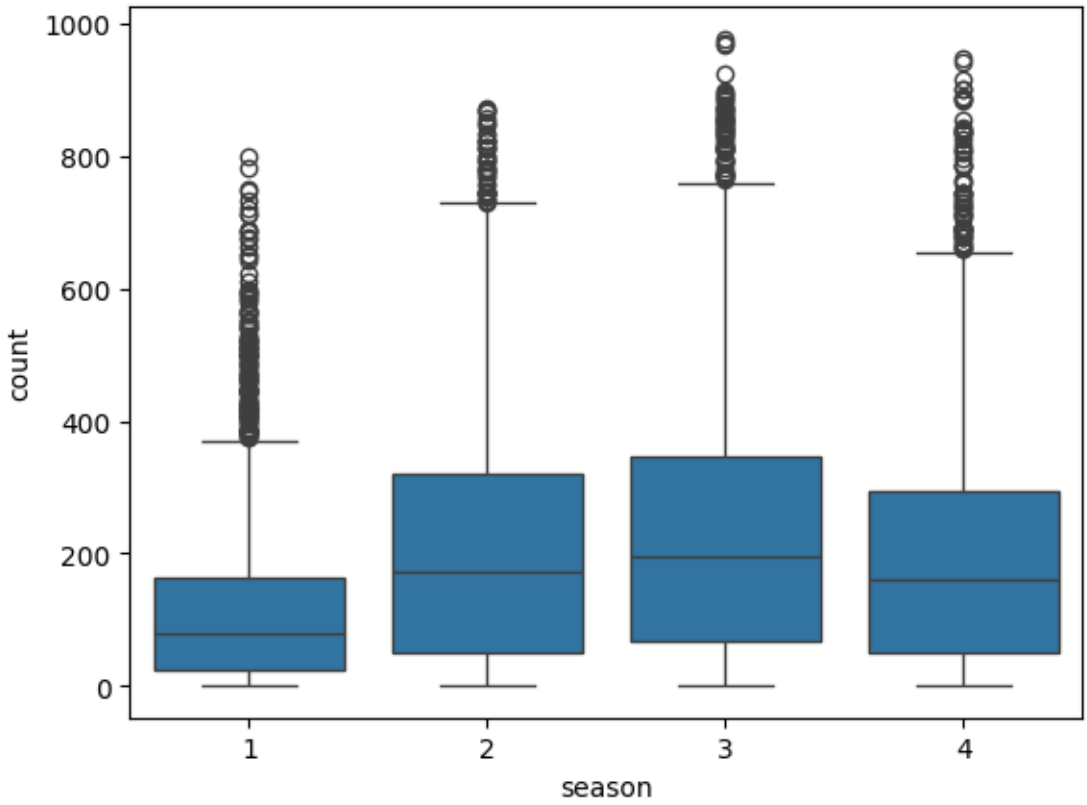


[39]:
```
np.mean(spring['count'].sample(10))
```

[39]:
```
93.9
```

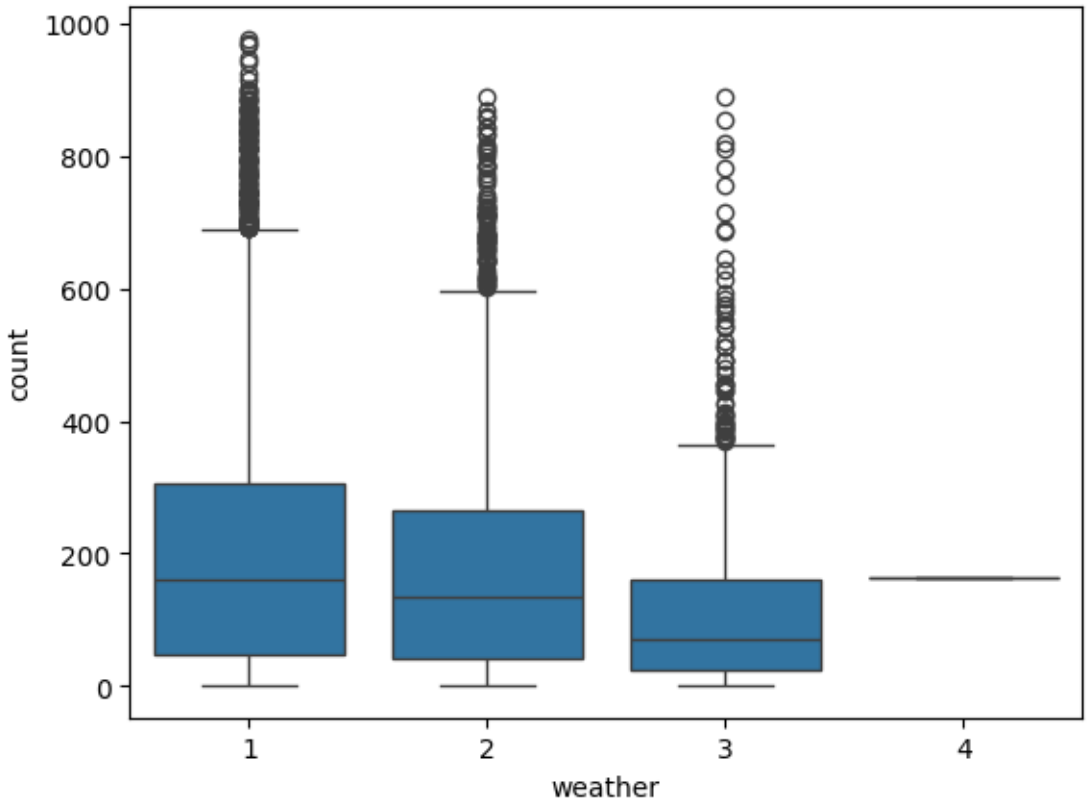
[40]:
```
sns.boxplot(x='season',y='count',data=df)
```

[40]:
```
<Axes: xlabel='season', ylabel='count'>
```



[41]:
```
sns.boxplot(x='weather',y='count',data=df)
```

[41]:
```
<Axes: xlabel='weather', ylabel='count'>
```



#Weather

[42]:
```
#weather:
 #      1: Clear, Few clouds, partly cloudy, partly cloudy
 #      2: Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist
 #      3: Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain␣
 ↪+ Scattered clouds
 #      4: Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog
```

[43]:
```
weather1 = df[df['weather']==1]['count']
weather2 = df[df['weather']==2]['count']
weather3 = df[df['weather']==3]['count']
weather4 = df[df['weather']==4]['count']
```

[44]:
```
weather1.mean()
```

[44]: 205.23679087875416

[45]:
```
weather2.mean()
```

[45]: 178.95553987297106

[46]:
```
weather3.mean()
```

[46]:
```
118.84633294528521
```

[47]:
```
weather4.mean()
```

[47]:
```
164.0
```

[48]:
```
#H0 : No. Of cycles rented is no differance in differentweather
#Ha : No. Of cycles rented is different in different weathers


# 1=2
# 1 2
```

[49]:
```
f_stats, p_value = f_oneway(weather1,weather2,weather3,weather4)
```

[50]:
```
if p_value < 0.05:
    print("Reject H0")
    print("Atleast one group have different mean")
else:
    print("Fail to reject H0")
    print("All groups have same mean")
```

```
Reject H0
Atleast one group have different mean
```

#**(2) Season**

[51]:
```
#H0 : No. Of cycles rented is no differance in different Season
#Ha : No. Of cycles rented is different in different Season


# 1=2
# 1 2
```

[52]:
```
season1 = df[df['season']==1]['count']
season2 = df[df['season']==2]['count']
season3 = df[df['season']==3]['count']
season4 = df[df['season']==4]['count']
```

[53]:
```
f_stats, p_value = f_oneway(season1,season2,season3,season4)
f_stats, p_value
```

[53]:
```
(236.94671081032106, 6.164843386499654e-149)
```

[54]:
```
if p_value < 0.05:
    print("Reject H0")
```

```
    print("Atleast one group have different mean")
else:
    print("Fail to reject H0")
    print("All groups have same mean")
```

```
Reject H0
Atleast one group have different mean
```

#**Chi-Square**

What should be the null and alternate hypothesis?

H0 : weather is not dependent on the season

H1 : weather is dependent on the season

[60]:
```
from scipy.stats import chisquare,chi2,chi2_contingency

df.head()
```

[60]:

	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

[58]:
```
weather_season = pd.crosstab(index=df['weather'],columns=df['season'])
weather_season
```

[58]:

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

[61]:
```
chi_stat, p_value, df, exp_freq = chi2_contingency(weather_season)


print('chi_stat :',chi_stat)
print('p_value :',p_value)
```

```
print('df :',df)
print('exp_freq :',exp_freq)
```

```
chi_stat : 49.15865559689363
p_value : 1.5499250736864862e-07
df : 9
exp_freq : [[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]]
```

[63]:
```
alpha = 0.05

if p_value < alpha:
    print("Reject H0")
    print("weather is dependent on the season")
else:
    print("Fail to reject H0")
    print("weather is not dependent on the season")
```

```
Reject H0
weather is dependent on the season
```