In [1]:

```
import numpy as np
import pandas
import seaborn
import matplotlib.pyplot as plt
```

In [2]:

```
df = pandas.read_csv('datasets/bike.csv')
df.head()
```

Out[2]:

	instant	dteday	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	
0	1	2011-01-01	1	0	1	0	6	0	2	0.344167	0.
1	2	2011-01-02	1	0	1	0	0	0	2	0.363478	0.
2	3	2011-01-03	1	0	1	0	1	1	1	0.196364	0.
3	4	2011-01-04	1	0	1	0	2	1	1	0.200000	0.
4	5	2011-01-05	1	0	1	0	3	1	1	0.226957	0.

In [3]:

```
continuous_features = ['temp', 'atemp', 'hum', 'windspeed','casual', 'registered',
'cnt']
discrete_features = ['season', 'yr', 'mnth', 'holiday', 'weekday','workingday', 'wea
thersit']
```

Calculate the following statistical metrics for each one of the features (except dteday): min, max, mean, standard deviation, minN, maxN, Q1, Q2 (median), Q3, IQR

In [4]:

```
def calculate_statistical_metrics(names,df):
    nfeatures = len(names)
    m = 10  # min, max, mean, std, Q1, Q2 (median), Q3, IRQ, minN, maxN

    values = np.zeros((m,nfeatures))
    columns = df.columns
    index = ['min', 'max', 'mean', 'std', 'Q1', 'Q2 (median)', 'Q3', 'IRQ', 'minN',
'maxN']

    for i,column in enumerate(names):
        values[0,i] = np.min( df[column].values  )
        values[1,i] = np.max( df[column].values  )
        values[2,i] = np.mean( df[column].values  )
        values[3,i] = np.std( df[column].values  )
        values[4,i] = np.percentile( df[column].values, 25 )
        values[5,i] = np.percentile( df[column].values, 50 )
        values[6,i] = np.percentile( df[column].values, 75 )
        values[7,i] = values[6,i]-values[4,i]
        values[8,i] = values[4,i] - 1.5*values[7,i]
        values[9,i] = values[6,i] + 1.5*values[7,i]

    return pandas.DataFrame(values,columns=names,index=index)
```

In [5]:

```
dfcontinuos = calculate_statistical_metrics(continuous_features,df)
dfcontinuos
```

Out[5]:

	temp	atemp	hum	windspeed	casual	registered	cr
min	0.059130	0.079070	0.000000	0.022392	2.000000	20.000000	22.00000
max	0.861667	0.840896	0.972500	0.507463	3410.000000	6946.000000	8714.00000
mean	0.495385	0.474354	0.627894	0.190486	848.176471	3656.172367	4504.34883
std	0.182926	0.162850	0.142332	0.077445	686.152682	1559.188805	1935.88595
Q1	0.337083	0.337842	0.520000	0.134950	315.500000	2497.000000	3152.00000
Q2 (median)	0.498333	0.486733	0.626667	0.180975	713.000000	3662.000000	4548.00000
Q3	0.655417	0.608602	0.730209	0.233214	1096.000000	4776.500000	5956.00000
IRQ	0.318333	0.270760	0.210209	0.098265	780.500000	2279.500000	2804.00000
minN	-0.140416	-0.068297	0.204687	-0.012447	-855.250000	-922.250000	-1054.00000
maxN	1.132916	1.014741	1.045521	0.380611	2266.750000	8195.750000	10162.00000

In [6]:

```
dfdiscrete = calculate_statistical_metrics(discrete_features,df)
dfdiscrete
```

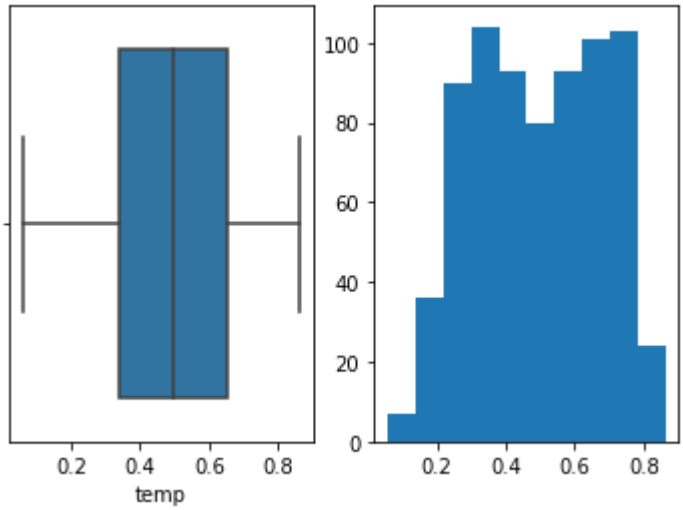
Out[6]:

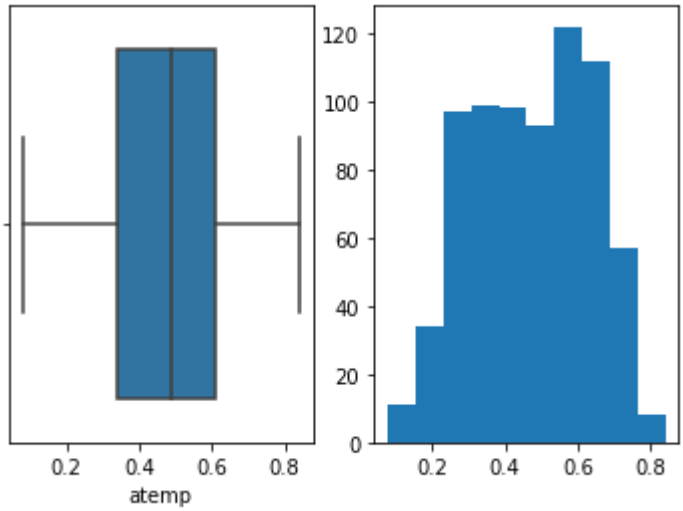
	season	yr	mnth	holiday	weekday	workingday	weathersit
min	1.000000	0.000000	1.000000	0.000000	0.000000	0.000000	1.000000
max	4.000000	1.000000	12.000000	1.000000	6.000000	1.000000	3.000000
mean	2.496580	0.500684	6.519836	0.028728	2.997264	0.683995	1.395349
std	1.110047	0.500000	3.449551	0.167040	2.003415	0.464915	0.544522
Q1	2.000000	0.000000	4.000000	0.000000	1.000000	0.000000	1.000000
Q2 (median)	3.000000	1.000000	7.000000	0.000000	3.000000	1.000000	1.000000
Q3	3.000000	1.000000	10.000000	0.000000	5.000000	1.000000	2.000000
IRQ	1.000000	1.000000	6.000000	0.000000	4.000000	1.000000	1.000000
minN	0.500000	-1.500000	-5.000000	0.000000	-5.000000	-1.500000	-0.500000
maxN	4.500000	2.500000	19.000000	0.000000	11.000000	2.500000	3.500000

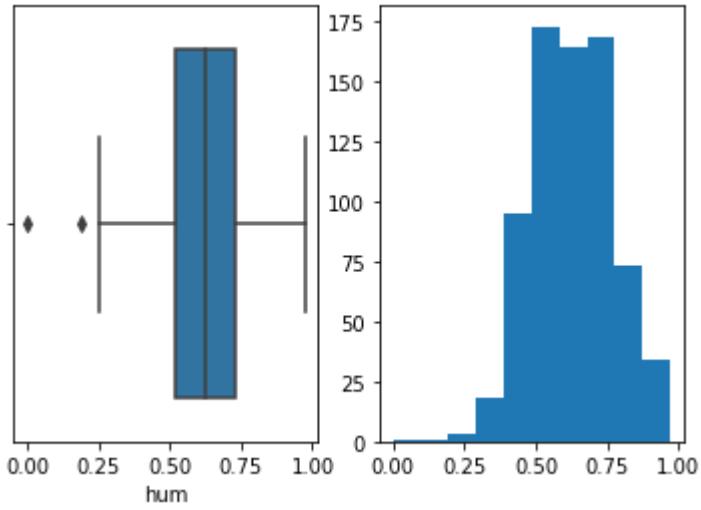
Graph for each feature (except dteday): histogram and boxplot

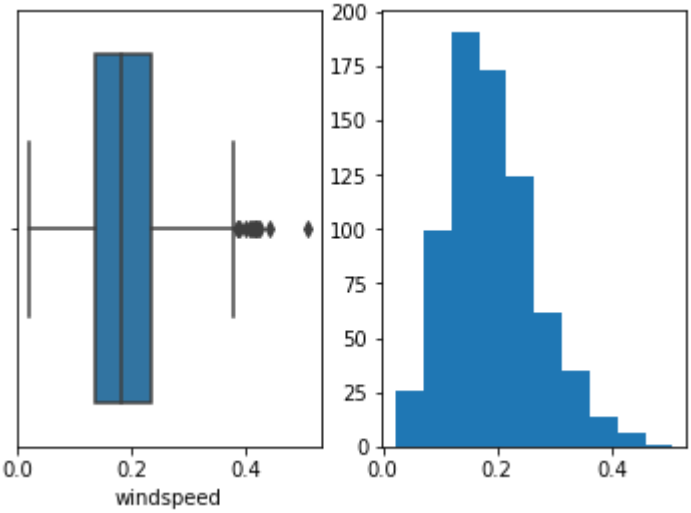
In [7]:

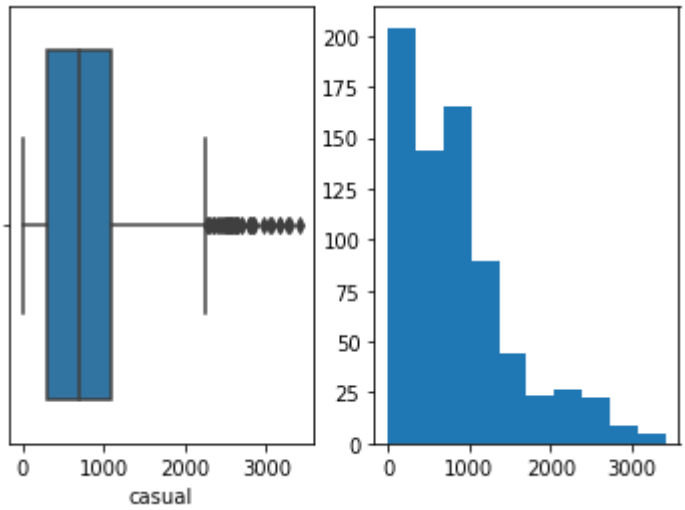
```
for feature in continuous_features:
    plt.figure()
    plt.subplot(1,2,1)
    seaborn.boxplot(data=df,x=feature)
    plt.subplot(1,2,2)
    plt.hist(df[feature].values)
    plt.show()
```

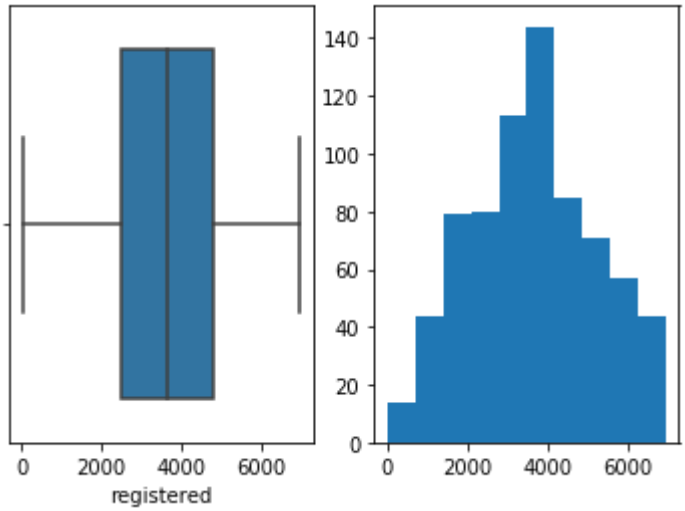

0.2
0.4
0.6
0.8
temp
0
20
40
60
80
100
0.2
0.4
0.6
0.8

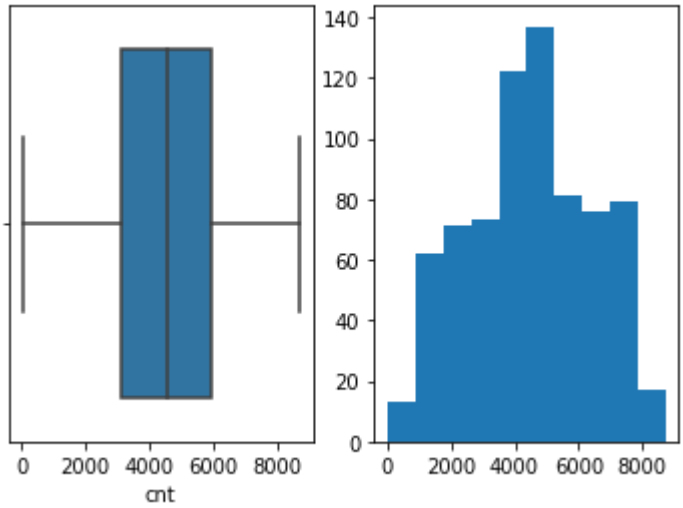

0.2
0.4
0.6
0.8
atemp
0
20
40
60
80
100
120
0.2
0.4
0.6
0.8


0.00
0.25
0.50
0.75
1.00
hum
0
25
50
75
100
125
150
175
0.00
0.25
0.50
0.75
1.00


windspeed

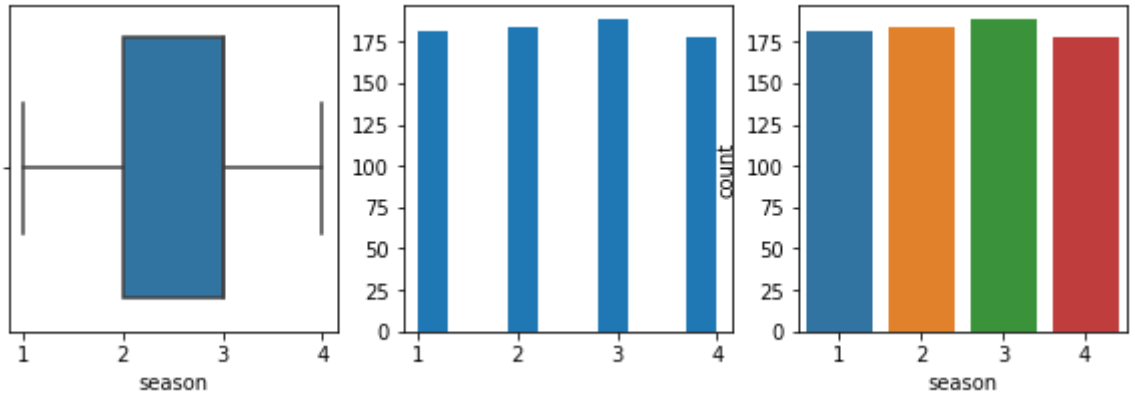

casual

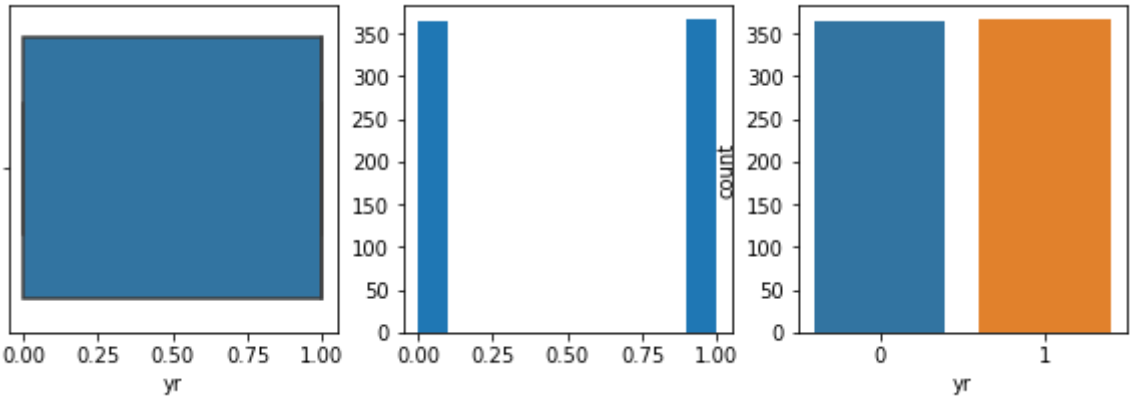

registered

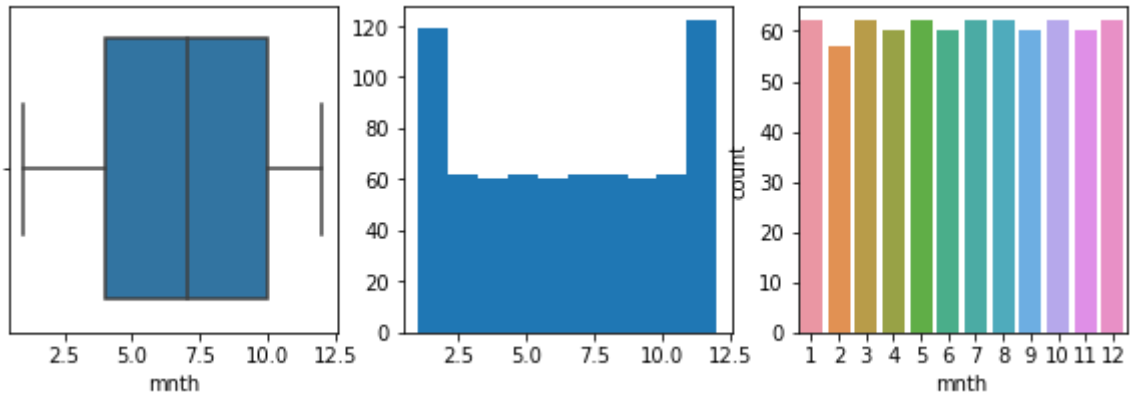

0
2000
4000
6000
8000
cnt
140
120
100
80
60
40
20
0
0
2000
4000
6000
8000

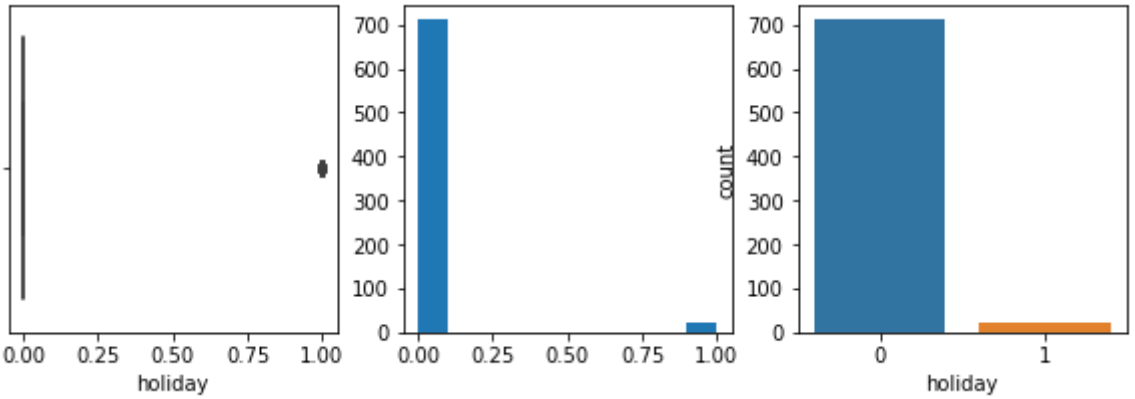
In [8]:

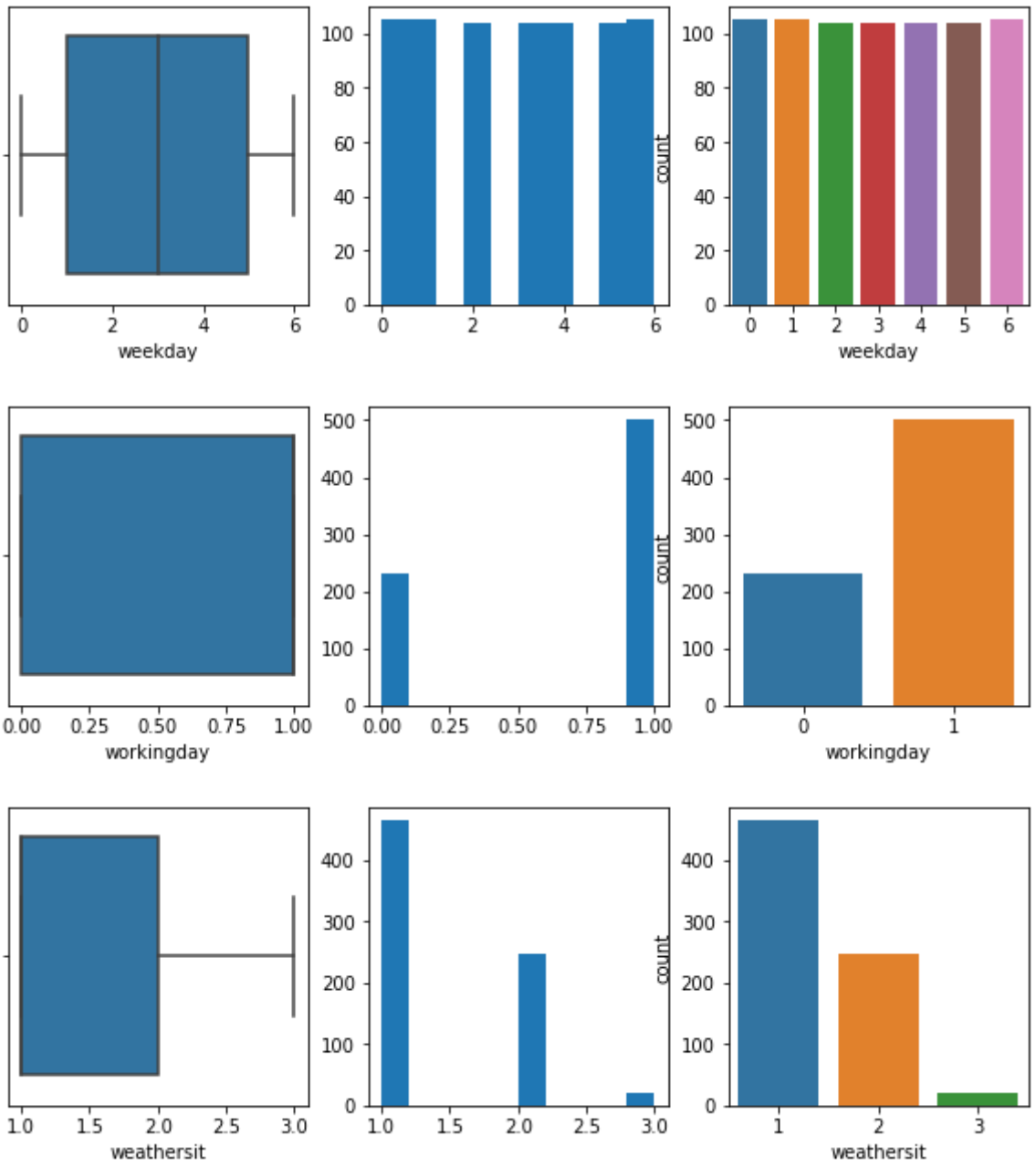
```
for feature in discrete_features:
    plt.figure(figsize=(10,3))
    plt.subplot(1,3,1)
    seaborn.boxplot(data=df,x=feature)
    plt.subplot(1,3,2)
    plt.hist(df[feature].values)
    plt.subplot(1,3,3)
    seaborn.countplot(data=df,x=feature)
    plt.show()
```


season
count
season


yr
count
yr


mnth
count
mnth

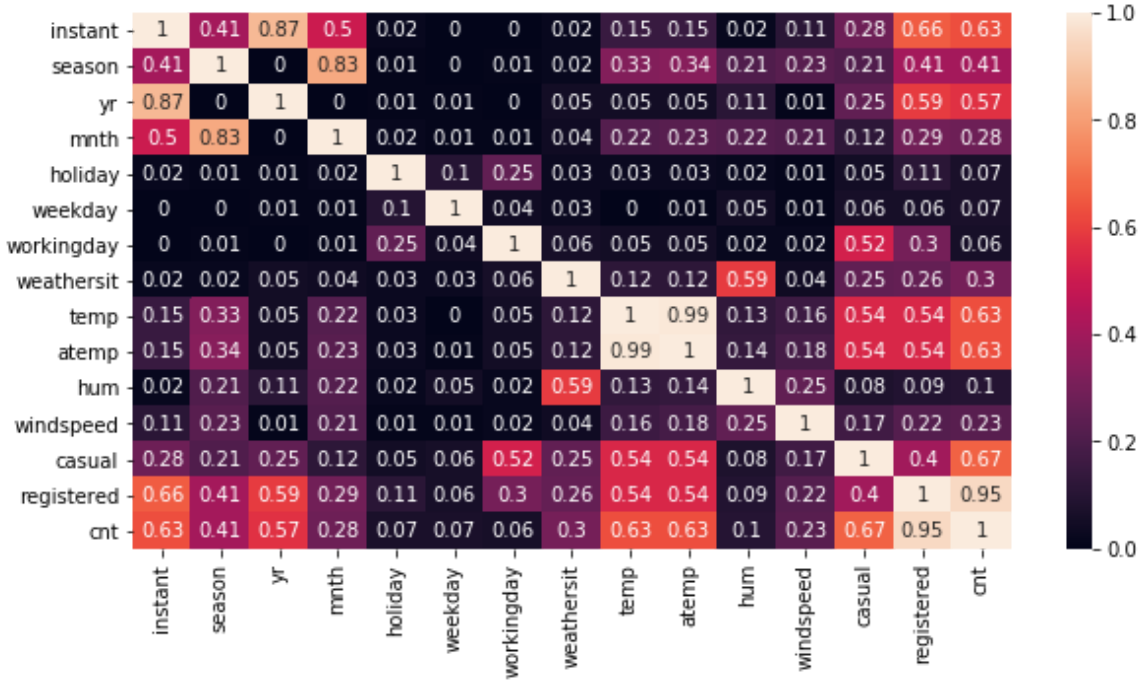

holiday
count
holiday



Graph the correlation matrix (showing the absolute values)

In [9]:

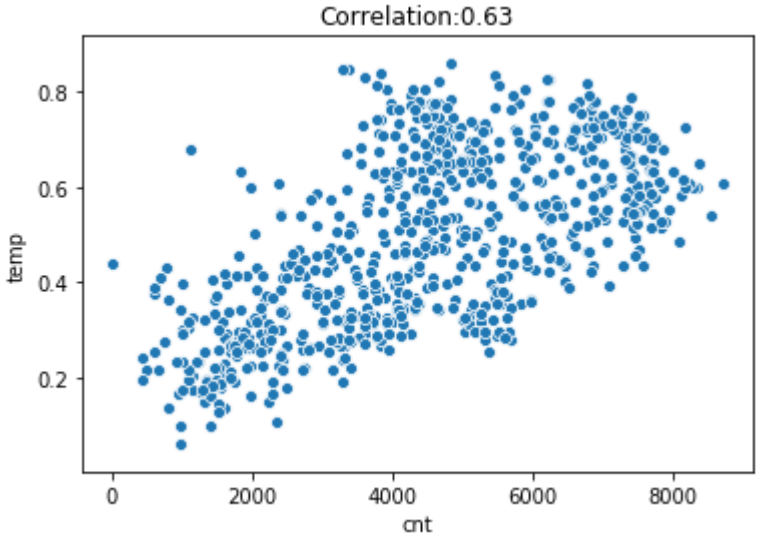
```
dfc = df.corr()
dfcabs = pandas.DataFrame(np.round(np.abs(dfc.values),2),index=dfc.index, columns=df
c.columns)
plt.figure(figsize=(10,5))
seaborn.heatmap(dfcabs,annot=True)
plt.show()
```

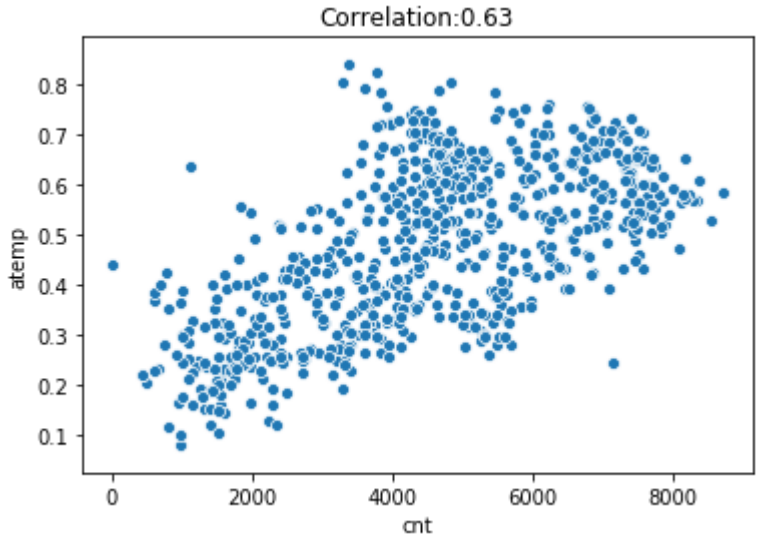


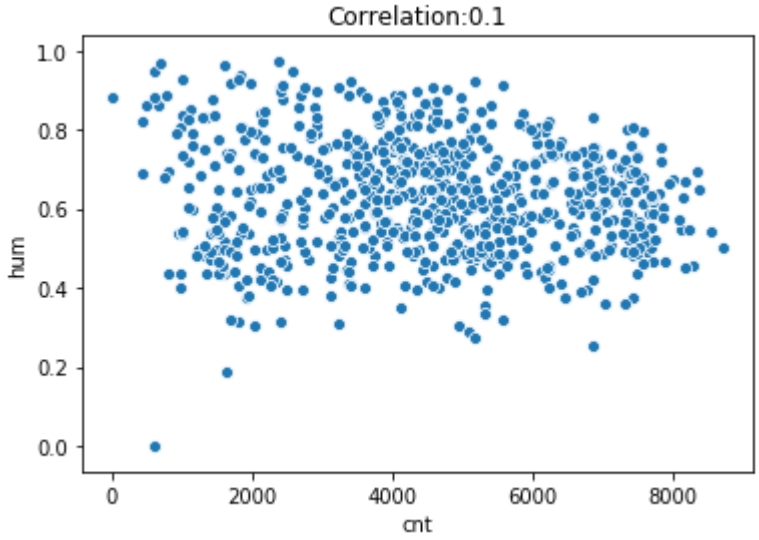
Graph the scatter plot using the feature cnt and the one with the highest correlation with that feature (cnt). Do the same with the feature with the lowest correlation.

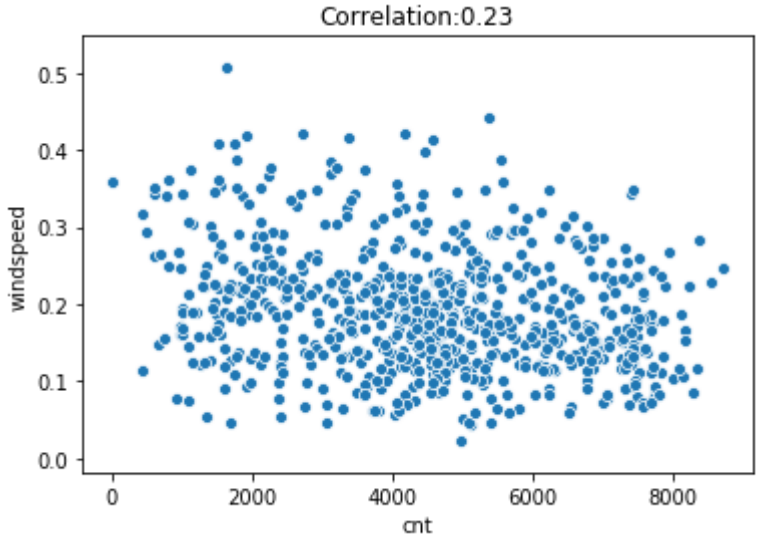
In [10]:

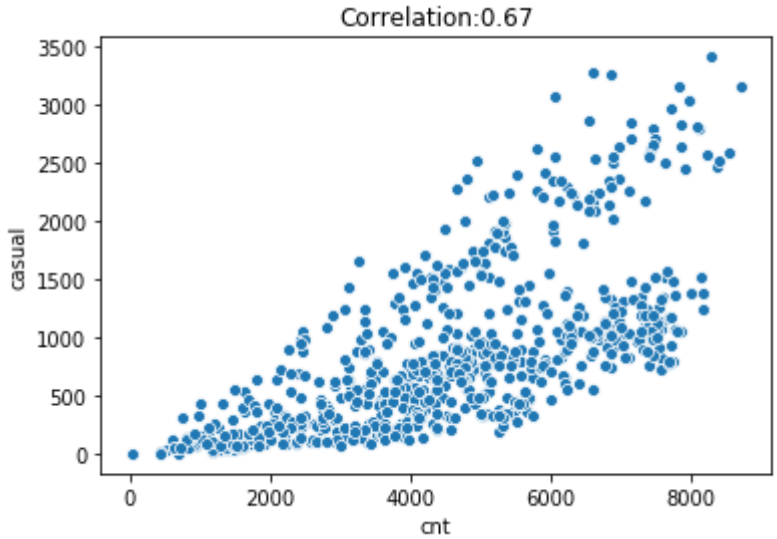
```
for feature in continuous_features:
    plt.figure()
    plt.title( 'Correlation:'+ str(dfcabs.loc['cnt'][feature]) )
    seaborn.scatterplot(data=df, x='cnt', y=feature)
    plt.show()
```

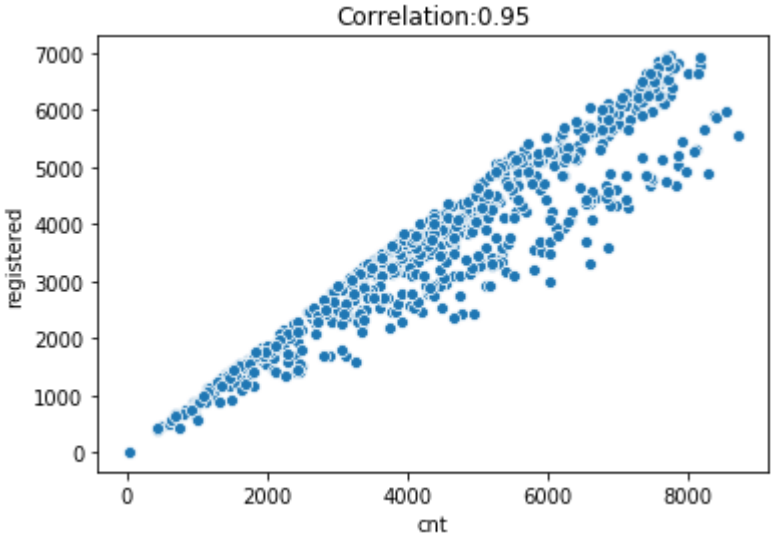

Correlation:0.63
temp
cnt
0.2
0.4
0.6
0.8
0
2000
4000
6000
8000

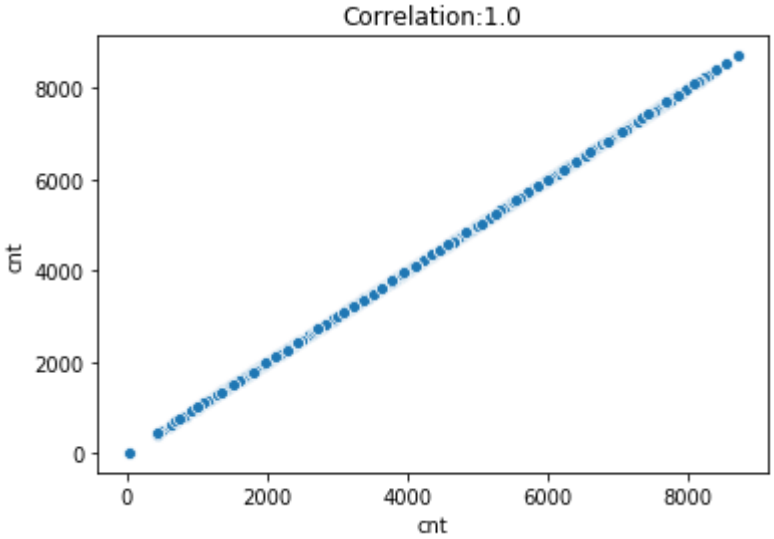

Correlation:0.63
atemp
cnt
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0
2000
4000
6000
8000


Correlation:0.1
hum
cnt
0.0
0.2
0.4
0.6
0.8
1.0
0
2000
4000
6000
8000


Correlation:0.23
windspeed
cnt
0.0
0.1
0.2
0.3
0.4
0.5
0
2000
4000
6000
8000

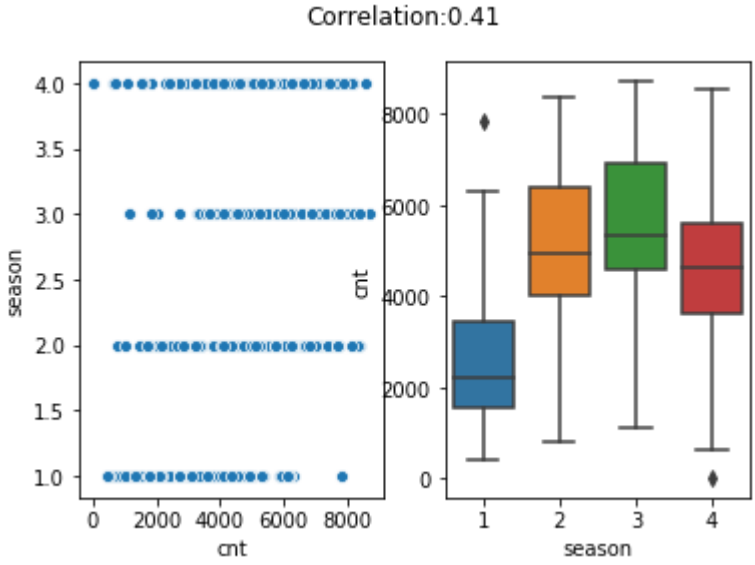

Correlation:0.67
casual
cnt
0
500
1000
1500
2000
2500
3000
3500
0
2000
4000
6000
8000

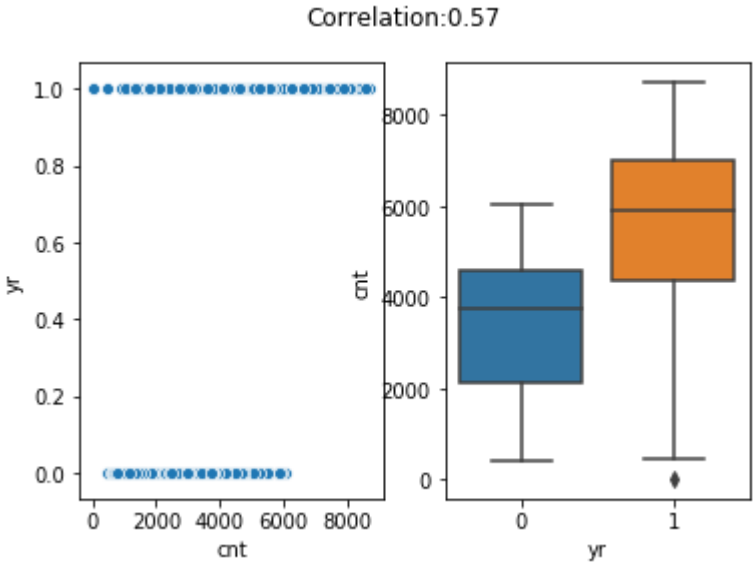

Correlation:0.95
registered
cnt
0
1000
2000
3000
4000
5000
6000
7000
0
2000
4000
6000
8000

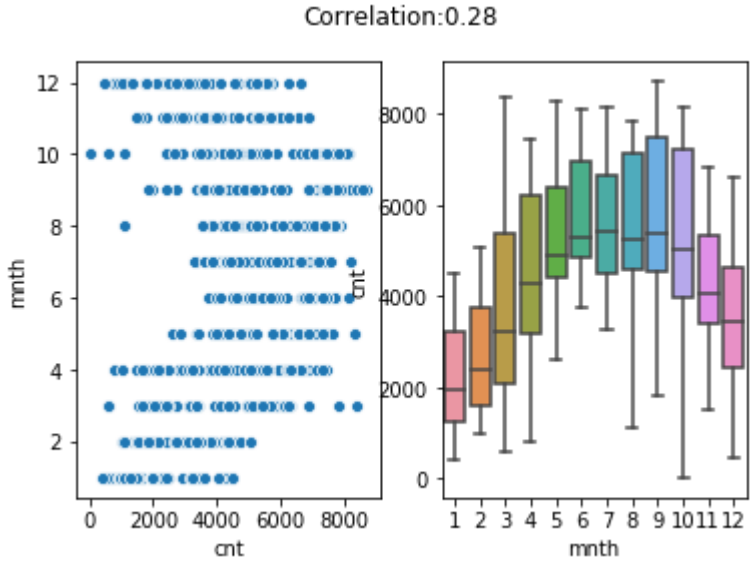

Correlation:1.0
cnt
cnt
0
2000
4000
6000
8000
0
2000
4000
6000
8000

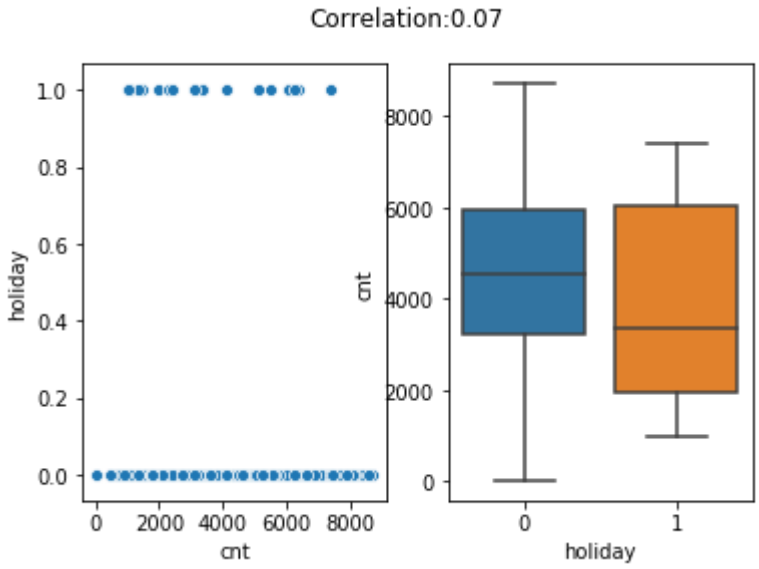
In [11]:

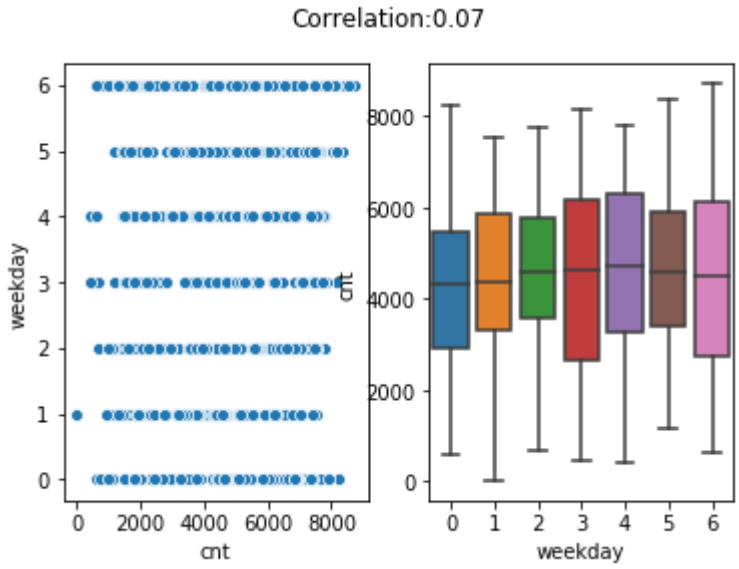
```
for feature in discrete_features:
    plt.figure()
    plt.suptitle( 'Correlation:'+ str(dfcabs.loc['cnt'][feature]) )
    plt.subplot(1,2,1)
    seaborn.scatterplot(data=df, x='cnt', y=feature)
    plt.subplot(1,2,2)
    seaborn.boxplot(data=df, y='cnt',x=feature )
    plt.show()
```

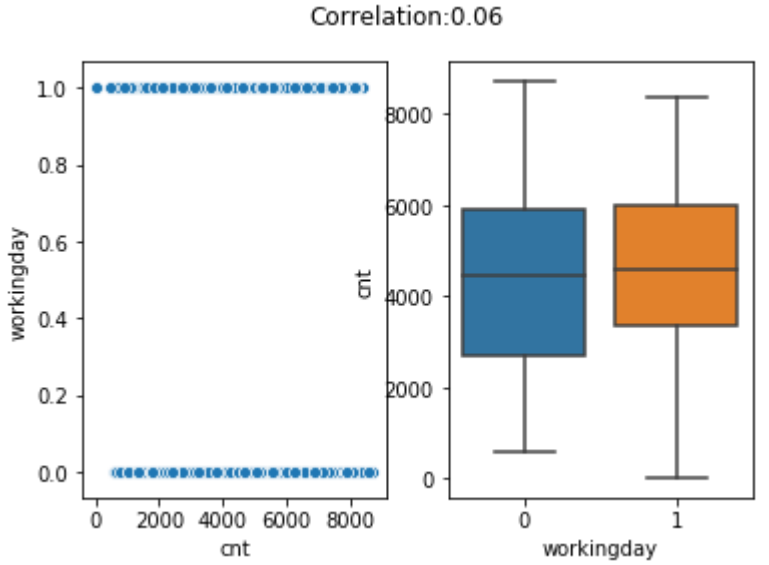

Correlation:0.41
season
cnt
cnt
season

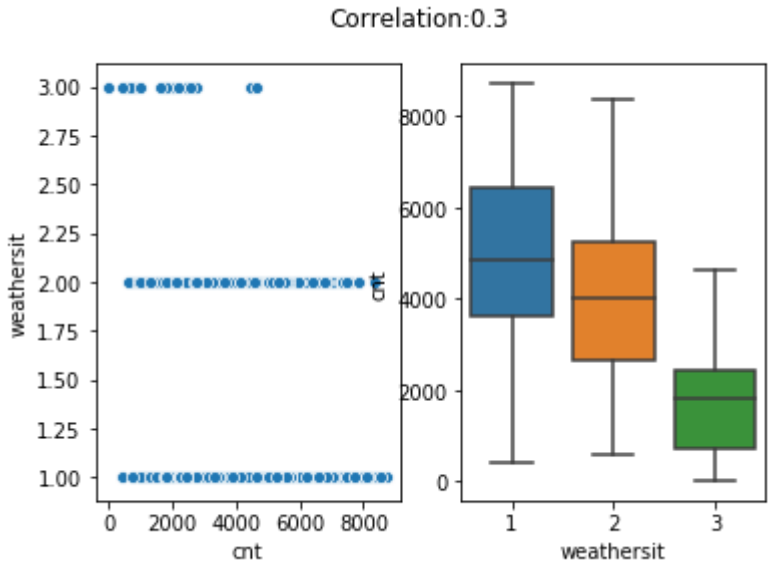

Correlation:0.57
yr
cnt
cnt
yr


Correlation:0.28
mnth
cnt
cnt
mnth


Correlation:0.07
holiday
cnt
cnt
holiday


Correlation:0.07
weekday
cnt
cnt
weekday





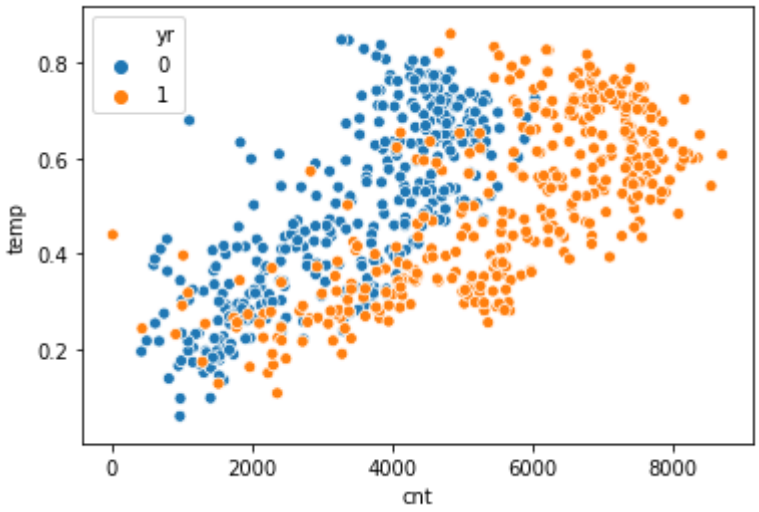
Graph one scatter plot using four features: x, y, size, and color

In [12]:

```
seaborn.scatterplot(data=df, x='cnt',y='temp',hue='yr')
```

Out[12]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x262eeb06448>
```

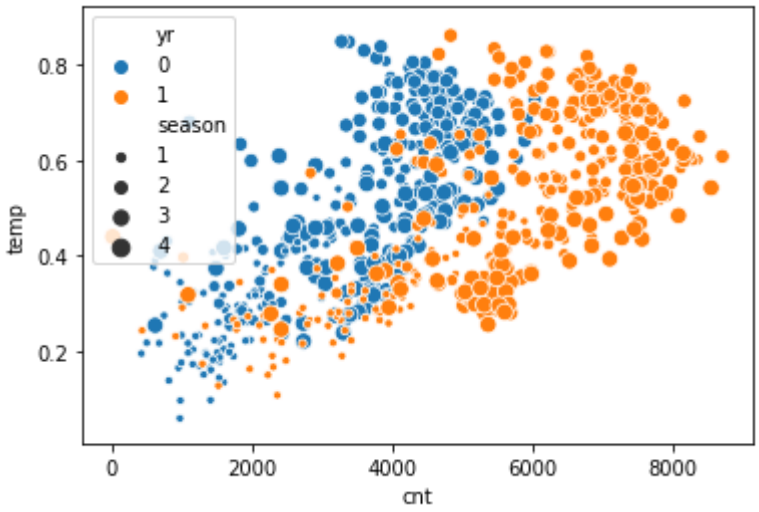


In [13]:

```
seaborn.scatterplot(data=df, x='cnt',y='temp',size='season',hue='yr')
```

Out[13]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x262eec8af48>
```

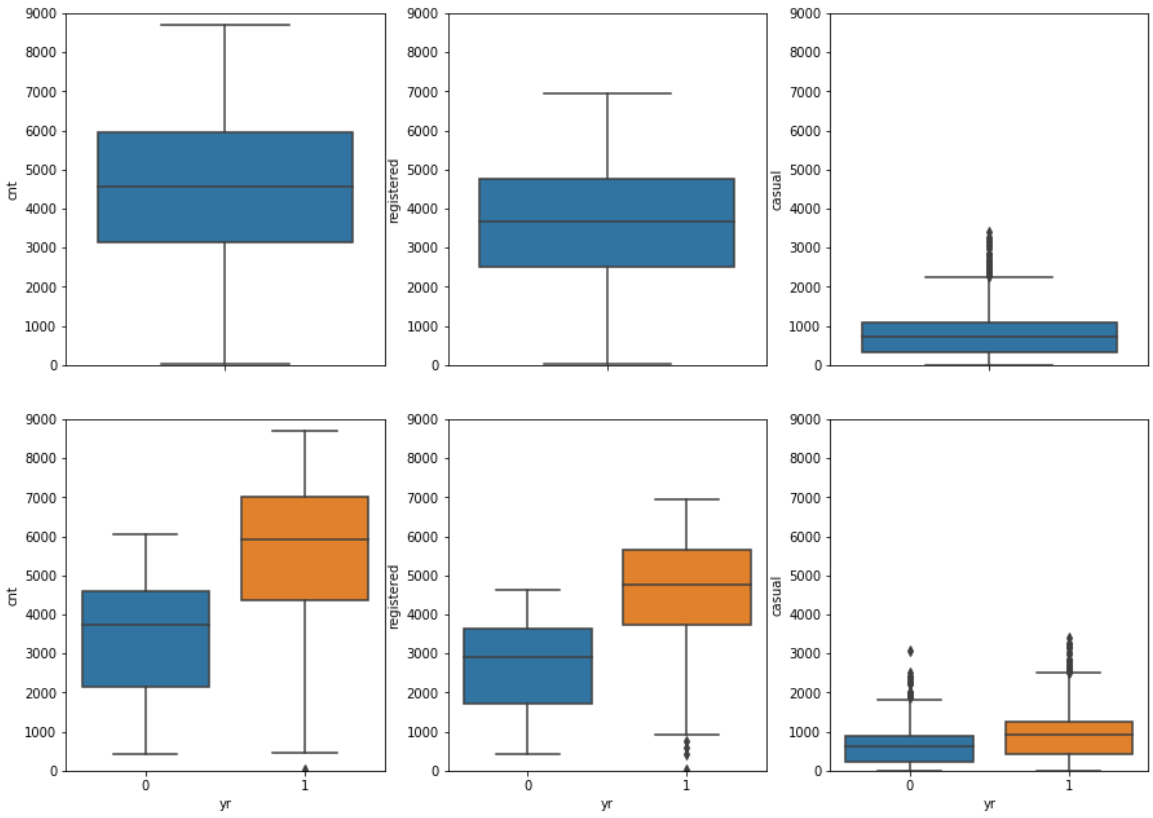


Write at least four insights from the data (justifying with metrics or graphs). Insights consist of some interesting facts that you discover from the Exploratory Data Analysis

1. The number of rents is significantly higher in the registered users than in casual users. In addition, the rents in the second year are higher than in the first year.

In [14]:

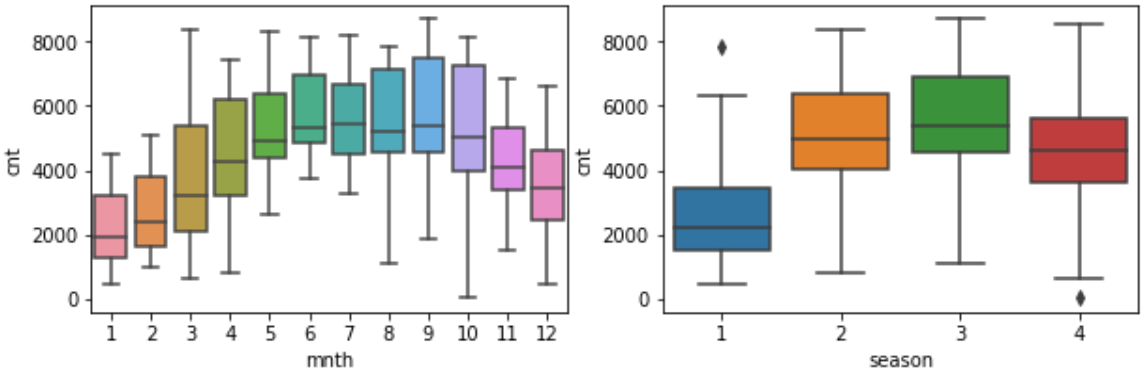
```
plt.figure(figsize=(15,5))
plt.subplot(1,3,1)
seaborn.boxplot(data=df,y='cnt')
plt.ylim([0,9000])
plt.subplot(1,3,2)
seaborn.boxplot(data=df,y='registered')
plt.ylim([0,9000])
plt.subplot(1,3,3)
seaborn.boxplot(data=df,y='casual')
plt.ylim([0,9000])
plt.show()

plt.figure(figsize=(15,5))
plt.subplot(1,3,1)
seaborn.boxplot(data=df, x='yr',y='cnt')
plt.ylim([0,9000])
plt.subplot(1,3,2)
seaborn.boxplot(data=df, x='yr', y='registered')
plt.ylim([0,9000])
plt.subplot(1,3,3)
seaborn.boxplot(data=df, x='yr', y='casual')
plt.ylim([0,9000])
plt.show()
```



1. Cnt is highly related to month and season.

In [15]:

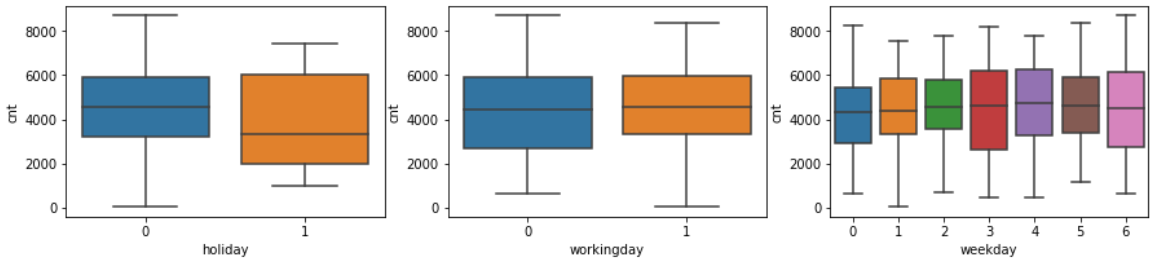
```
plt.figure(figsize=(10,3))
plt.subplot(1,2,1)
seaborn.boxplot(data=df, x='mnth',y='cnt')
plt.subplot(1,2,2)
seaborn.boxplot(data=df, x='season', y='cnt')
plt.show()
```

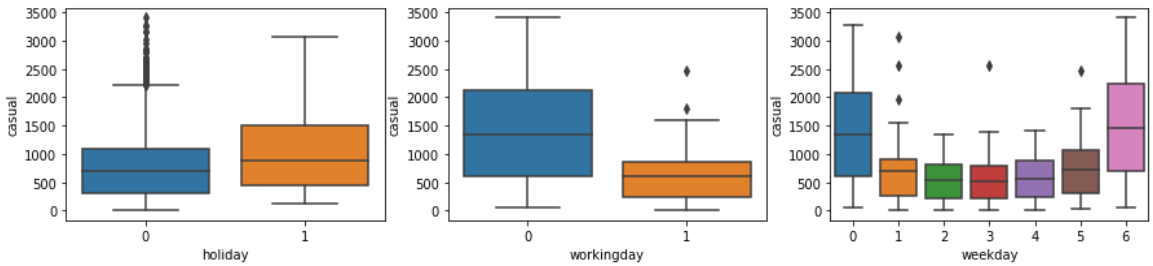


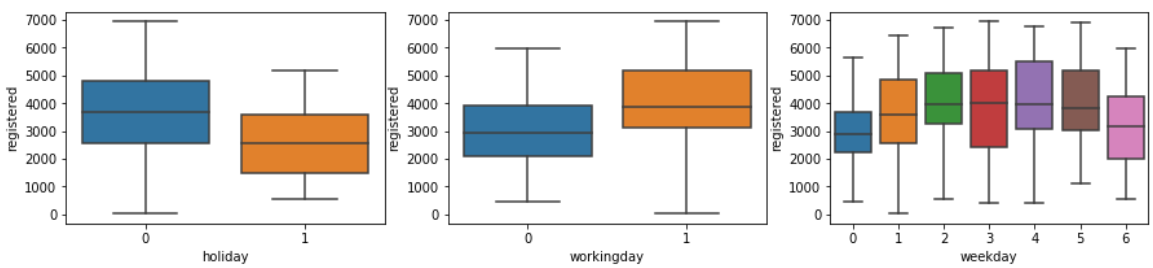
1. In general, the type of days does not affect the rents. However, registered users have more rents on working days. Opposite with casual users that rent bikes generally on weekends.

In [16]:

```
plt.figure(figsize=(15,3))
plt.subplot(1,3,1)
seaborn.boxplot(data=df, x='holiday',y='cnt')
plt.subplot(1,3,2)
seaborn.boxplot(data=df, x='workingday', y='cnt')
plt.subplot(1,3,3)
seaborn.boxplot(data=df, x='weekday', y='cnt')
plt.show()

plt.figure(figsize=(15,3))
plt.subplot(1,3,1)
seaborn.boxplot(data=df, x='holiday',y='casual')
plt.subplot(1,3,2)
seaborn.boxplot(data=df, x='workingday', y='casual')
plt.subplot(1,3,3)
seaborn.boxplot(data=df, x='weekday', y='casual')
plt.show()

plt.figure(figsize=(15,3))
plt.subplot(1,3,1)
seaborn.boxplot(data=df, x='holiday',y='registered')
plt.subplot(1,3,2)
seaborn.boxplot(data=df, x='workingday', y='registered')
plt.subplot(1,3,3)
seaborn.boxplot(data=df, x='weekday', y='registered')
plt.show()
```







1. Rents are correlated with temperature but not with humidity or wind speed.

In [17]:

```
plt.figure(figsize=(10,3))
plt.subplot(1,3,1)
plt.title( 'Correlation:'+ str(dfcabs.loc['cnt']['temp']) )
seaborn.scatterplot(data=df, x='cnt',y='temp')
plt.subplot(1,3,2)
plt.title( 'Correlation:'+ str(dfcabs.loc['cnt']['hum']) )
seaborn.scatterplot(data=df, x='cnt',y='hum')
plt.subplot(1,3,3)
plt.title( 'Correlation:'+ str(dfcabs.loc['cnt']['windspeed']) )
seaborn.scatterplot(data=df, x='cnt', y='windspeed')
plt.show()
```