lab5

December 4, 2019

[57]:
```
import pandas as pd
import numpy as np
import seaborn as sns
from statistics import mean
```

0.0.1 1.Visualize and interpret the pattern of this time-series

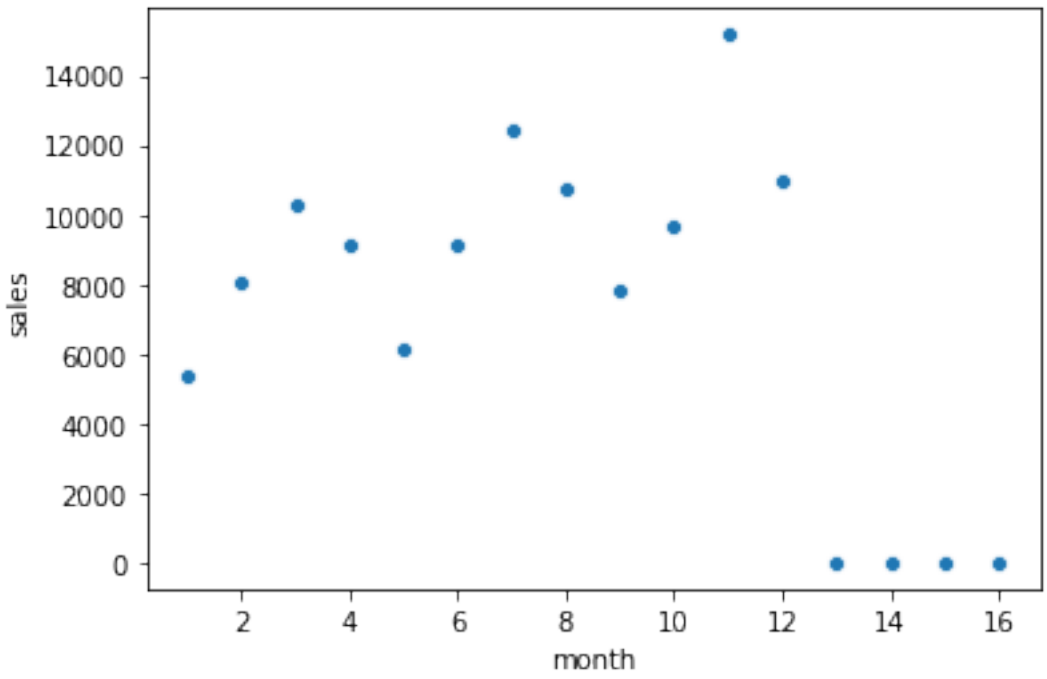
[3]:
```
sales_pd = pd.read_excel("./sales.xls",encoding="utf8")
sales_pd = sales_pd.set_index("month")
sales_pd
```

[3]:

month	sales
1	5384
2	8081
3	10282
4	9156
5	6118
6	9139
7	12460
8	10717
9	7825
10	9693
11	15177
12	10990
13	0
14	0
15	0
16	0

[4]:
```
sns.scatterplot(sales_pd.index,sales_pd['sales'])
```

[4]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1cdc9fd12b0>
```



[55]:
```
type(sales_pd.loc[10]['sales'])
```

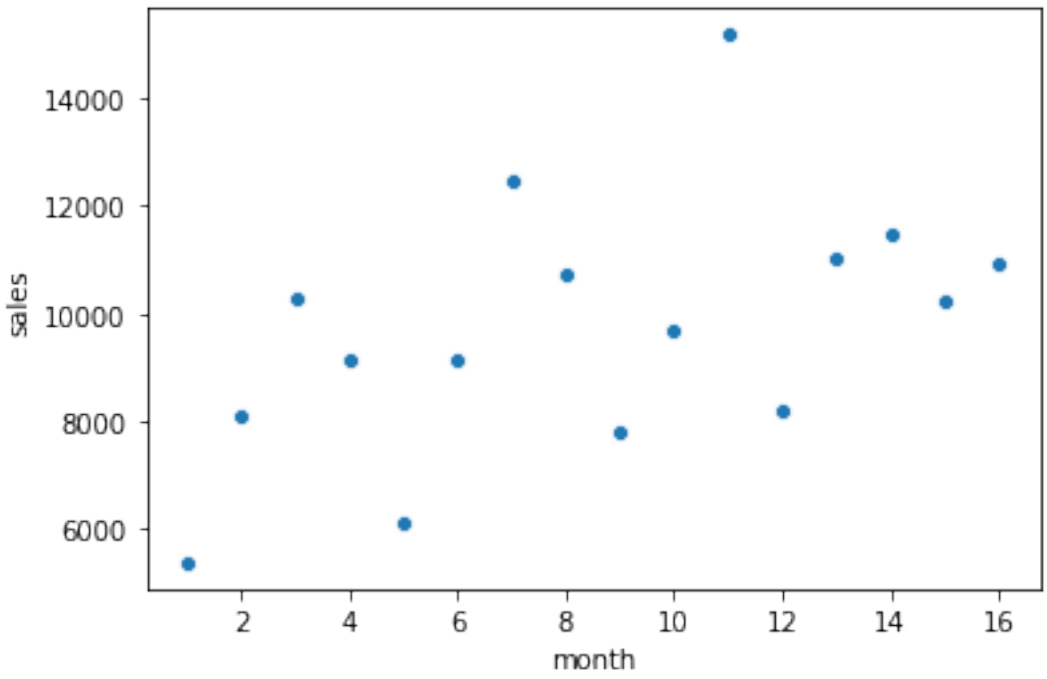
[55]:
```
numpy.int64
```

have seasonality

0.0.2 moving averge method

[54]:
```
#moving average method
for index in range(13,17):
    sales_pd.loc[index].sales = mean([sales_pd.loc[index-1].sales,sales_pd.
 ↪loc[index-2].sales,sales_pd.loc[index-3].sales])
sns.scatterplot(sales_pd.index,sales_pd['sales'])
```

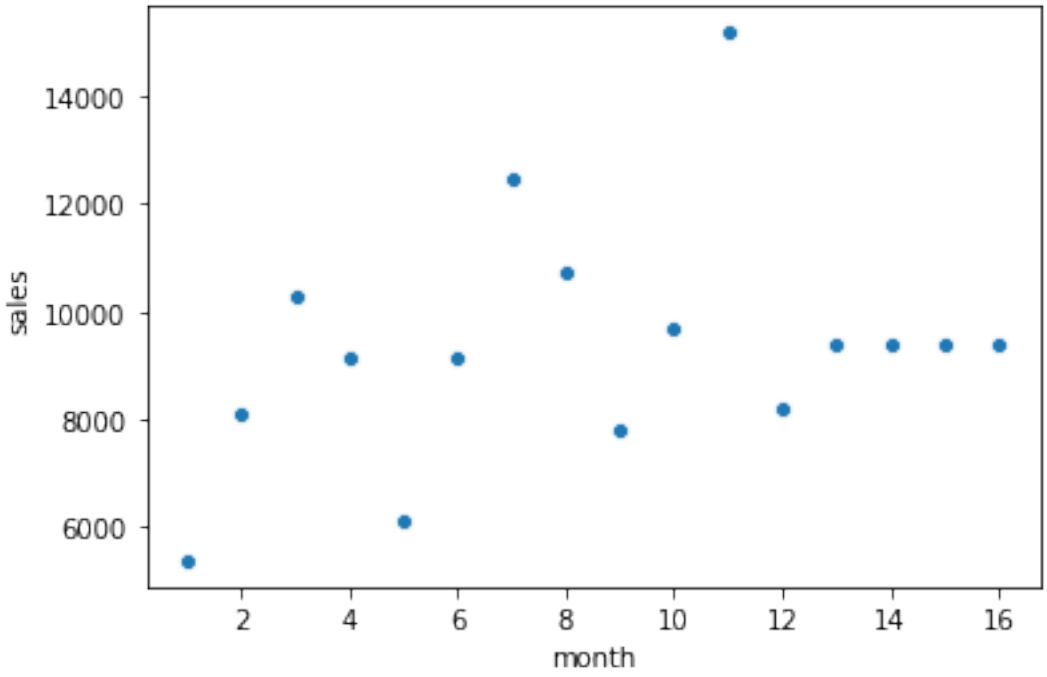
[54]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1cdcc0a94a8>
```



0.0.3 exponential smoothing

[53]:
```
alpha=0.8
for idx in range(13,17):
    value = 0
    # index is F_{t} t
    for pw in range(0,idx-1):
        #rint(alpha*(1-alpha)**pw*sales_pd.loc[idx-1-pw].sales)
        value = value + alpha*(1-alpha)**pw*sales_pd.loc[idx-1-pw].sales
        #rint(value)
    sales_pd.loc[idx].sales = value
    #rint(value)
    #rint("###############")

sns.scatterplot(sales_pd.index,sales_pd['sales'])
```

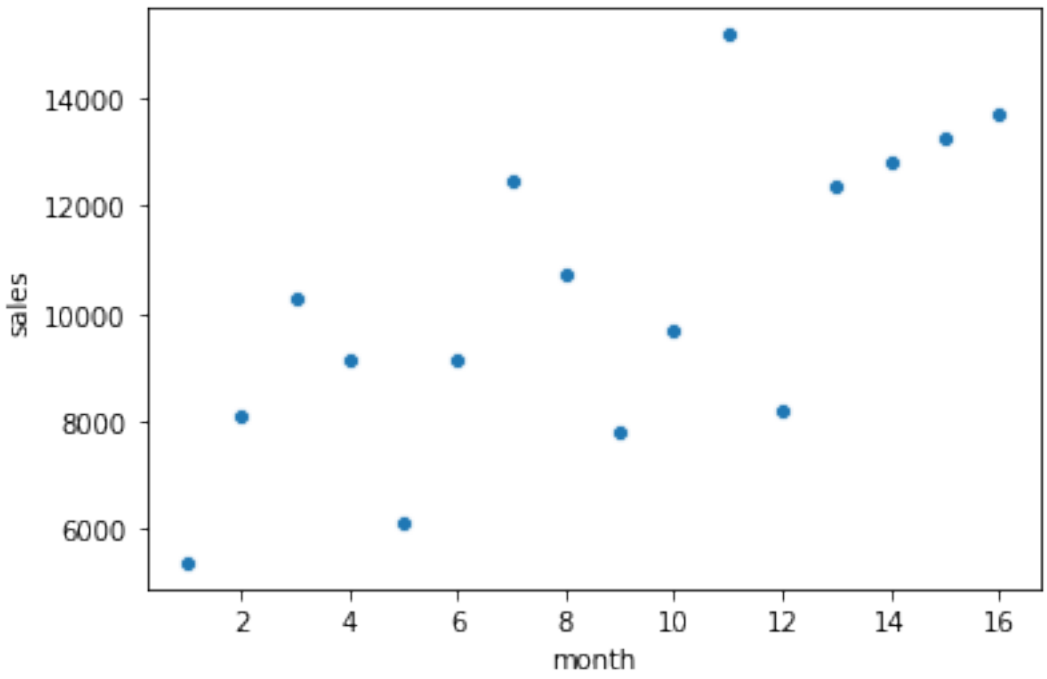
[53]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1cdcc0a4b70>
```



0.0.4 linear regression

[52]:
```
from sklearn.linear_model import LinearRegression
import numpy as np
lr = LinearRegression()
lr.fit(np.array(sales_pd.index).reshape(-1,1),sales_pd['sales'])
#lr.predict([[idx]])
for idx in range(13,17):
    sales_pd.loc[idx].sales = lr.predict([[idx]])[0]
sns.scatterplot(sales_pd.index,sales_pd['sales'])
```

[52]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1cdcbfe0cf8>
```

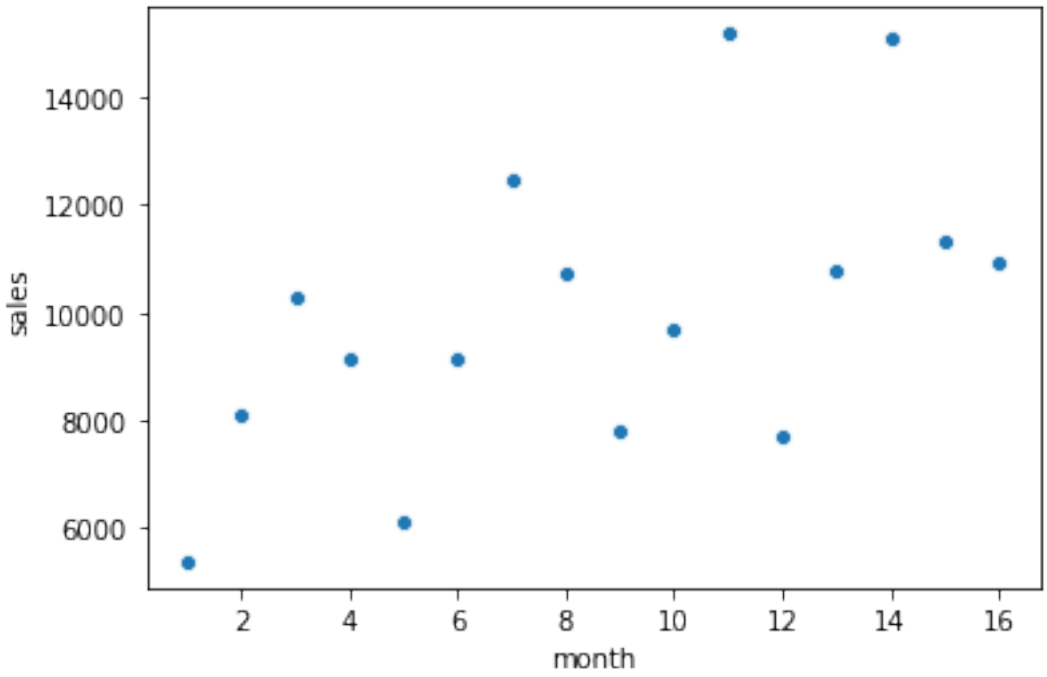


0.0.5 Predict future demand in month 13,14,15,16 with seasonality

[56]:
```
avg = []
for x in range(0,3):
    avg.append(mean([sales_pd.loc[x*4+1].sales,sales_pd.loc[x*4+2].
 ↪sales,sales_pd.loc[x*4+3].sales,sales_pd.loc[x*4+4].sales]))
avg

# array of season index
t_index=[]
for g in range(0,3):
    #season index for a group
    s_index= []
    for e in range(1,5):
        s_index.append(sales_pd.loc[g*4+e].sales/avg[g])
    t_index.append(s_index)

avg_4 = avg[2]+mean([avg[2]-avg[1],avg[1]-avg[0]])
avg_4
avg_index =[]

for e in range(0,4):
    avg_index.append(mean([t_index[0][e],t_index[1][e],t_index[2][e]]))
avg_index
```

```
for e in range(0,4):
    sales_pd.loc[12+e].sales = avg_4*avg_index[e]
sns.scatterplot(sales_pd.index,sales_pd['sales'])
```

[56]: `<matplotlib.axes._subplots.AxesSubplot at 0x1cdcc16b7f0>`



0.0.6 1. Basics of Recommendation Algorithm

[ ]: