Chapter 18 - exercise 1: Sales of shampoo over a three year

Cho dữ liệu bán shampoo 3 năm trong tập tin sales-of-shampoo-over-a-three-year.csv.

- Thực hiện việc dự báo bán sản phẩm shampoo sử dụng thuật toán ARIMA
- Cho biết trong 3 tháng sau 3 năm trên thì giá trị bán sản phẩm như thế nào?

In [1]:
```
import pandas as pd
```

In [2]:
```
data = pd.read_csv("sales-of-shampoo-over-a-three-year.csv", index_col=0)
data.head()
```

Out[2]:

Month	Sales of shampoo over a three year period
Friday, January 1, 2016	266.0
Monday, February 1, 2016	145.9
Tuesday, March 1, 2016	183.1
Friday, April 1, 2016	119.3
Sunday, May 1, 2016	180.3

In [3]:
```
data.index = pd.to_datetime(data.index)
```

In [4]:
```
data.index
```

Out[4]:
```
DatetimeIndex(['2016-01-01', '2016-02-01', '2016-03-01', '2016-04-01',
               '2016-05-01', '2016-06-01', '2016-07-01', '2016-08-01',
               '2016-09-01', '2016-10-01', '2016-11-01', '2016-12-01',
               '2017-01-01', '2017-02-01', '2017-03-01', '2017-04-01',
               '2017-05-01', '2017-06-01', '2017-07-01', '2017-08-01',
               '2017-09-01', '2017-10-01', '2017-11-01', '2017-12-01',
               '2018-01-01', '2018-02-01', '2018-03-01', '2018-04-01',
               '2018-05-01', '2018-06-01', '2018-07-01', '2018-08-01',
               '2018-09-01', '2018-10-01', '2018-11-01', '2018-12-01'],
              dtype='datetime64[ns]', name='Month', freq=None)
```

In [5]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 36 entries, 2016-01-01 to 2018-12-01
Data columns (total 1 columns):
Sales of shampoo over a three year period    36 non-null float64
dtypes: float64(1)
memory usage: 576.0 bytes
```

In [6]:
```
data.columns = ['Sales_of_shampoo']
```

In [7]:
```
data.head()
```

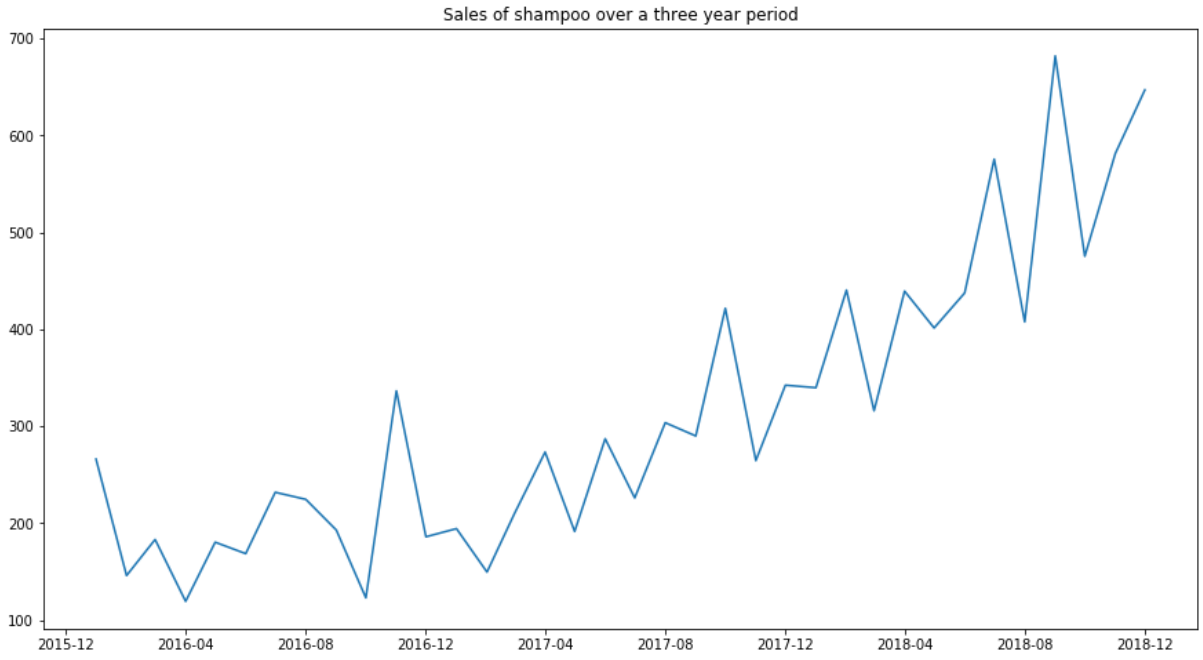
Out[7]:

Month	Sales_of_shampoo
2016-01-01	266.0
2016-02-01	145.9
2016-03-01	183.1
2016-04-01	119.3
2016-05-01	180.3

In [8]:
```
from datetime import datetime
```

In [9]:
```
import matplotlib.pyplot as plt
```

In [10]:
```
plt.figure(figsize=(15,8))
plt.plot(data)
plt.title("Sales of shampoo over a three year period")
plt.show()
```



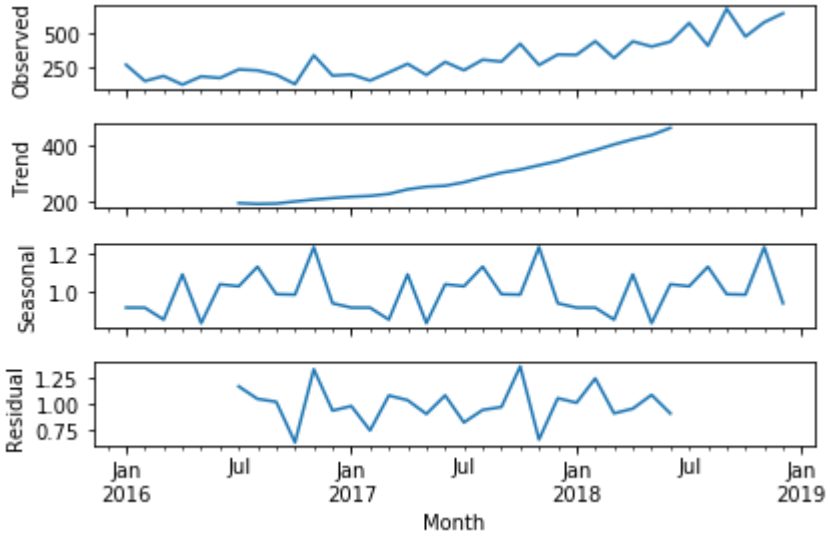
In [11]:
```
type(data)
```

Out[11]:
```
pandas.core.frame.DataFrame
```

In [12]:
```
from statsmodels.tsa.seasonal import seasonal_decompose
result = seasonal_decompose(x = data, model='multiplicative')
result
```

Out[12]:
```
<statsmodels.tsa.seasonal.DecomposeResult at 0x19ad1de0a20>
```

In [13]:
```
result.plot()
plt.show()
```



In [14]:
```
from pmdarima.arima import auto_arima
```

In [15]:
```
stepwise_model = auto_arima(data, start_p=1, start_q=1,
                            max_p=3, max_q=3, m=12,
                            start_P=0, seasonal=True,
                            d=1, D=1, trace=True,
                            error_action='ignore',
                            suppress_warnings=True,
                            stepwise=True)
```

```
Fit ARIMA: order=(1, 1, 1) seasonal_order=(0, 1, 1, 12); AIC=nan, BIC=nan, Fit
time=nan seconds
Fit ARIMA: order=(0, 1, 0) seasonal_order=(0, 1, 0, 12); AIC=307.761, BIC=310.
032, Fit time=0.024 seconds
Fit ARIMA: order=(1, 1, 0) seasonal_order=(1, 1, 0, 12); AIC=287.966, BIC=292.
508, Fit time=0.387 seconds
Fit ARIMA: order=(0, 1, 1) seasonal_order=(0, 1, 1, 12); AIC=nan, BIC=nan, Fit
time=nan seconds
Fit ARIMA: order=(1, 1, 0) seasonal_order=(0, 1, 0, 12); AIC=288.331, BIC=291.
737, Fit time=0.067 seconds
Fit ARIMA: order=(1, 1, 0) seasonal_order=(2, 1, 0, 12); AIC=nan, BIC=nan, Fit
time=nan seconds
Fit ARIMA: order=(1, 1, 0) seasonal_order=(1, 1, 1, 12); AIC=nan, BIC=nan, Fit
time=nan seconds
Fit ARIMA: order=(1, 1, 0) seasonal_order=(2, 1, 1, 12); AIC=nan, BIC=nan, Fit
time=nan seconds
Fit ARIMA: order=(0, 1, 0) seasonal_order=(1, 1, 0, 12); AIC=305.077, BIC=308.
483, Fit time=0.317 seconds
Fit ARIMA: order=(2, 1, 0) seasonal_order=(1, 1, 0, 12); AIC=284.983, BIC=290.
660, Fit time=0.827 seconds
Fit ARIMA: order=(2, 1, 1) seasonal_order=(1, 1, 0, 12); AIC=nan, BIC=nan, Fit
time=nan seconds
Fit ARIMA: order=(3, 1, 1) seasonal_order=(1, 1, 0, 12); AIC=274.080, BIC=282.
029, Fit time=1.737 seconds
Fit ARIMA: order=(3, 1, 1) seasonal_order=(0, 1, 0, 12); AIC=274.003, BIC=280.
816, Fit time=0.759 seconds
Fit ARIMA: order=(3, 1, 1) seasonal_order=(0, 1, 1, 12); AIC=nan, BIC=nan, Fit
time=nan seconds
Fit ARIMA: order=(3, 1, 1) seasonal_order=(1, 1, 1, 12); AIC=nan, BIC=nan, Fit
time=nan seconds
Fit ARIMA: order=(2, 1, 1) seasonal_order=(0, 1, 0, 12); AIC=nan, BIC=nan, Fit
time=nan seconds
Fit ARIMA: order=(3, 1, 0) seasonal_order=(0, 1, 0, 12); AIC=285.401, BIC=291.
079, Fit time=0.379 seconds
Fit ARIMA: order=(3, 1, 2) seasonal_order=(0, 1, 0, 12); AIC=nan, BIC=nan, Fit
time=nan seconds
Fit ARIMA: order=(2, 1, 0) seasonal_order=(0, 1, 0, 12); AIC=288.620, BIC=293.
162, Fit time=0.199 seconds
Total fit time: 4.729 seconds
```

In [16]:
```
print(stepwise_model.aic())
```

```
274.0028295021545
```

In [17]:
```
train = data.loc['2016-01-01':'2018-02-01']
test = data.loc['2018-01-01':]
```

In [18]:
```
test
```

Out[18]:

Month	Sales_of_shampoo
2018-01-01	339.7
2018-02-01	440.4
2018-03-01	315.9
2018-04-01	439.3
2018-05-01	401.3
2018-06-01	437.4
2018-07-01	575.5
2018-08-01	407.6
2018-09-01	682.0
2018-10-01	475.3
2018-11-01	581.3
2018-12-01	646.9

In [19]:
```
len(test)
```

Out[19]:
```
12
```

In [20]:
```
len(train)
```

Out[20]:
```
26
```

In [21]:
```
stepwise_model.fit(train)
```

Out[21]:
```
ARIMA(callback=None, disp=0, maxiter=None, method=None, order=(3, 1, 1),
   out_of_sample_size=0, scoring='mse', scoring_args={},
   seasonal_order=(0, 1, 0, 12), solver='lbfgs', start_params=None,
   suppress_warnings=True, transparams=True, trend=None,
   with_intercept=True)
```

In [22]:
```
future_forecast = stepwise_model.predict(n_periods=len(test))
```

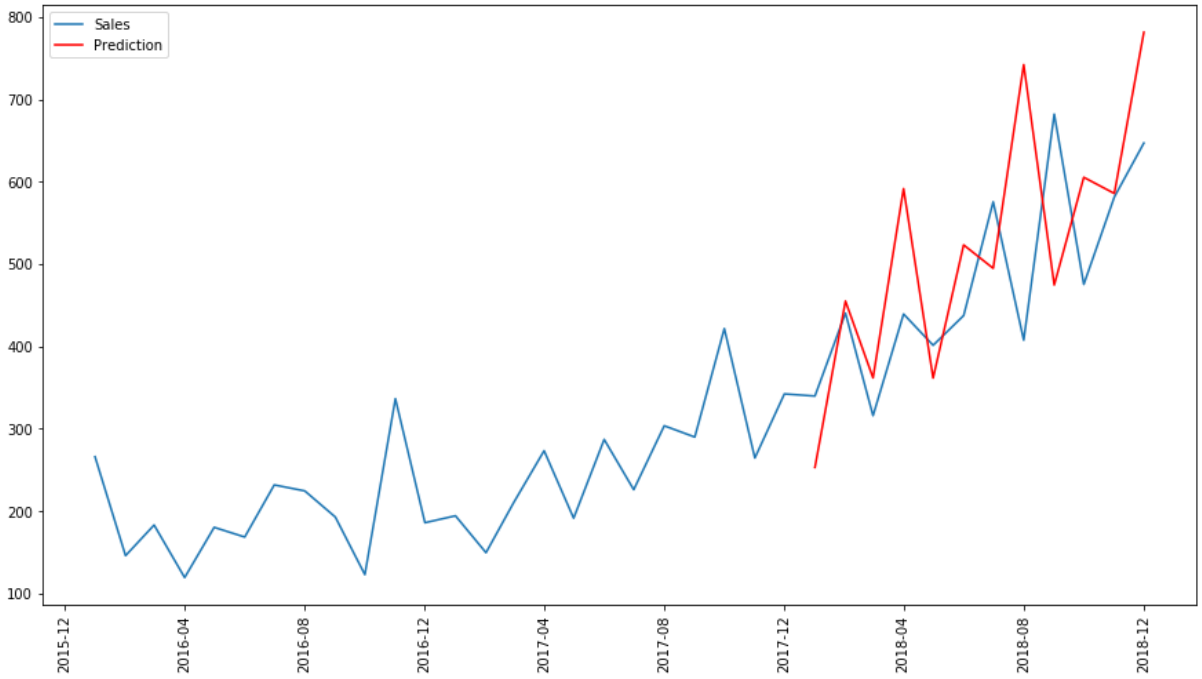
In [23]:
```
future_forecast
```

Out[23]:
```
array([253.0810698 , 455.01424285, 361.75011438, 591.56014563,
       361.53999986, 523.17147598, 494.78154287, 742.08900299,
       474.5548315 , 605.26881957, 585.66914232, 781.35313332])
```

In [24]:
```
future_forecast = pd.DataFrame(future_forecast,
                               index = test.index,
                               columns=['Prediction'])
```

In [25]:
```
plt.plot(test, label='Sales')
plt.plot(future_forecast, label='Prediction')
plt.xticks(rotation='vertical')
plt.legend()
plt.show()
```

In [26]:
```
plt.figure(figsize=(15,8))
plt.plot(data, label='Sales')
plt.plot(future_forecast, label='Prediction', color='red')
plt.xticks(rotation='vertical')
plt.legend()
plt.show()
```

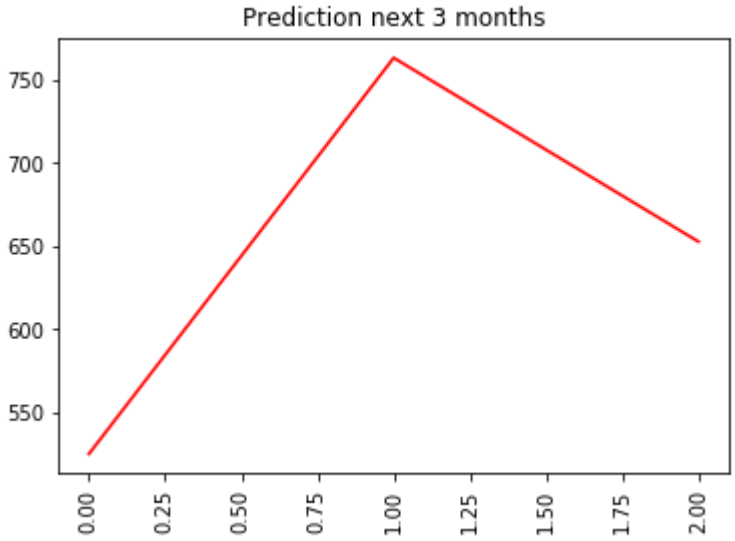


In [27]:
```
# Dự đoán 3 tháng sau
```

In [28]:
```
future_forecast = stepwise_model.predict(n_periods=len(test)+3)
future_forecast
```

Out[28]:
```
array([253.0810698 , 455.01424285, 361.75011438, 591.56014563,
       361.53999986, 523.17147598, 494.78154287, 742.08900299,
       474.5548315 , 605.26881957, 585.66914232, 781.35313332,
       525.22848133, 763.13328636, 652.71303393])
```

In [29]:
```
plt.plot(future_forecast[len(test):], color='red')
plt.xticks(rotation='vertical')
plt.title("Prediction next 3 months")
plt.show()
```



In [ ]:
```

```