Chapter 18 - Ex2: Car Sales

- Cho dữ liệu bán xe hơi 4 năm trong tập tin Retail2.xlsx.
- Thực hiện việc dự báo bán xe hơi sử dụng thuật toán ARIMA
- Cho biết trong 6 tháng sau 4 năm trên thì giá trị bán sản phẩm như thế nào?

In [2]:
```
import pandas as pd
```

In [3]:
```
data = pd.read_excel("Retail2.xlsx", index_col=0)
```

In [4]:
```
data.index = pd.to_datetime(data.index)
```

In [5]:
```
data.index
```

Out[5]:
```
DatetimeIndex(['2015-06-01', '2015-07-01', '2015-08-01', '2015-09-01',
               '2015-10-01', '2015-11-01', '2015-12-01', '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', '2019-01-01',
               '2019-02-01', '2019-03-01', '2019-04-01', '2019-05-01',
               '2019-06-01'],
              dtype='datetime64[ns]', name='Date', freq=None)
```

In [6]:
```
data.head()
```

Out[6]:

Date	Car sales
2015-06-01	67
2015-07-01	68
2015-08-01	66
2015-09-01	86
2015-10-01	84

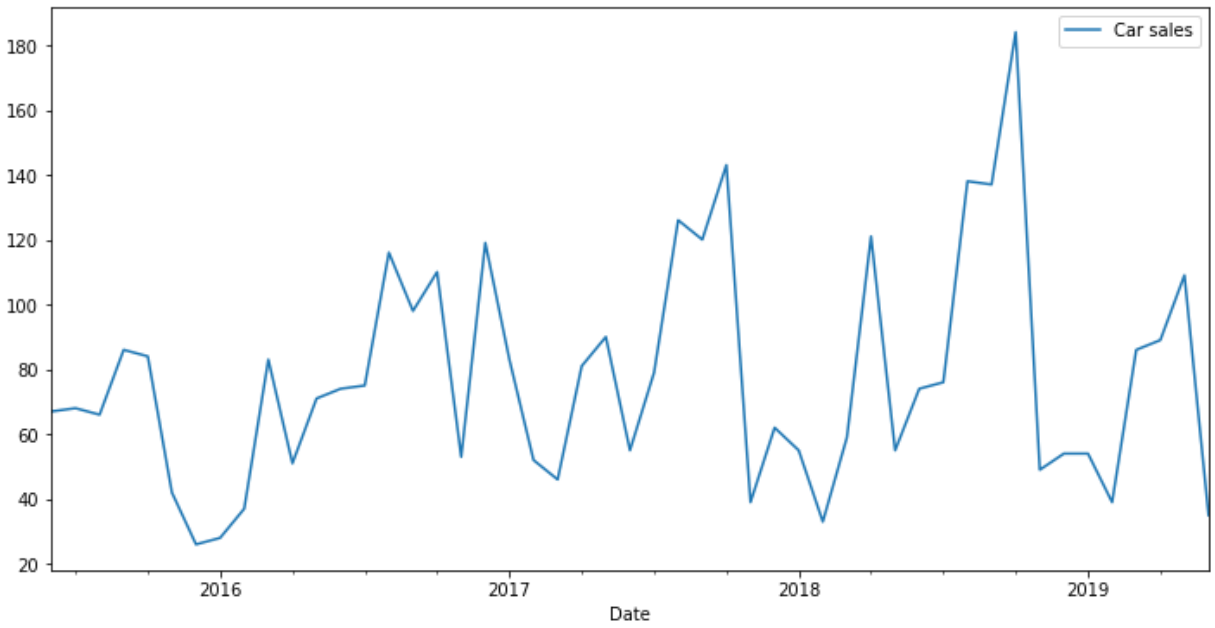
In [7]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 49 entries, 2015-06-01 to 2019-06-01
Data columns (total 1 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Car sales  49 non-null     int64
dtypes: int64(1)
memory usage: 784.0 bytes
```

In [8]:
```
import matplotlib.pyplot as plt
```

In [9]:
```
data.plot(figsize=(12,6))
```

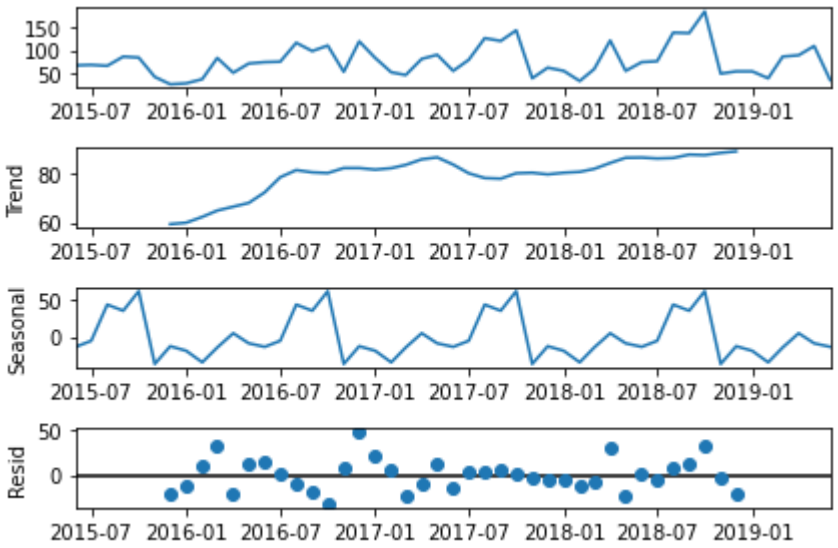
Out[9]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f78e3ed0978>
```



In [10]:
```
from statsmodels.tsa.seasonal import seasonal_decompose
result = seasonal_decompose(data, model='additive')
result
```

Out[10]:
```
<statsmodels.tsa.seasonal.DecomposeResult at 0x7f78e3ed0390>
```

In [11]:
```
fig = result.plot()
plt.show()
```



- Với kết quả trên, ta có thể thấy rõ tính seasonal component của data, và cũng có thể thấy xu hướng dữ liệu ở trên được tách riêng.
- Trend có thể lên hoặc xuống và có thể tuyến tính hoặc phi tuyến tính. Cần phải hiểu tập dữ liệu để biết liệu một khoảng thời gian đáng kể đã trôi qua có thể xác định xu hướng thực tế hay chưa.
- Cũng có thể có biến động bất thường (Irregular fluctuation) là những thay đổi đột ngột ngẫu nhiên và không thể đoán trước

Áp dụng auto_arima để xây dựng mô hình

In [13]:
```
from pmdarima import auto_arima
```

In [14]:
```
stepwise_model = auto_arima(data, start_p=2, start_q=2,
                            max_p=5, max_q=5, m=12,
                            start_P=1, seasonal=True,
                            d=1, D=1, trace=True,
                            error_action='ignore',
                            suppress_warnings=True,
                            stepwise=True)
```

```
Performing stepwise search to minimize aic
 ARIMA(2,1,2)(1,1,1)[12]             : AIC=inf, Time=0.90 sec
 ARIMA(0,1,0)(0,1,0)[12]             : AIC=371.830, Time=0.02 sec
 ARIMA(1,1,0)(1,1,0)[12]             : AIC=363.598, Time=0.15 sec
 ARIMA(0,1,1)(0,1,1)[12]             : AIC=353.684, Time=0.37 sec
 ARIMA(0,1,1)(0,1,0)[12]             : AIC=355.305, Time=0.06 sec
 ARIMA(0,1,1)(1,1,1)[12]             : AIC=355.684, Time=0.57 sec
 ARIMA(0,1,1)(0,1,2)[12]             : AIC=355.683, Time=1.12 sec
 ARIMA(0,1,1)(1,1,0)[12]             : AIC=354.150, Time=0.27 sec
 ARIMA(0,1,1)(1,1,2)[12]             : AIC=inf, Time=2.05 sec
 ARIMA(0,1,0)(0,1,1)[12]             : AIC=368.014, Time=0.17 sec
 ARIMA(1,1,1)(0,1,1)[12]             : AIC=355.567, Time=0.75 sec
 ARIMA(0,1,2)(0,1,1)[12]             : AIC=355.575, Time=0.44 sec
 ARIMA(1,1,0)(0,1,1)[12]             : AIC=362.459, Time=0.25 sec
 ARIMA(1,1,2)(0,1,1)[12]             : AIC=357.552, Time=0.59 sec
 ARIMA(0,1,1)(0,1,1)[12] intercept   : AIC=inf, Time=0.67 sec

Best model:  ARIMA(0,1,1)(0,1,1)[12]
Total fit time: 8.409 seconds
```

In [15]:
```
print(stepwise_model.aic())
```

```
353.68422726240806
```

In [16]:
```
train = data.loc['2015-06-01':'2018-02-01']
test = data.loc['2018-02-01':]
```

In [17]:
```
test.head()
```

Out[17]:

	Car sales
Date	
2018-02-01	33
2018-03-01	59
2018-04-01	121
2018-05-01	55
2018-06-01	74

In [18]:
```
len(test)
```

Out[18]: 17

Bước 2: Fit mô hình

In [19]:
```
stepwise_model.fit(train)
```

Out[19]:
```
ARIMA(maxiter=50, method='lbfgs', order=(0, 1, 1), out_of_sample_size=0,
      scoring='mse', scoring_args={}, seasonal_order=(0, 1, 1, 12),
      start_params=None, suppress_warnings=True, trend=None,
      with_intercept=False)
```

Bước 3: Dự đoán kết quả

In [22]:
```
future_forecast = stepwise_model.predict(n_periods=len(test)) # so khoang thoi g
```

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

Out[23]:
```
array([ 68.17928646,  69.68709326,  84.1672112 ,  75.47203818,
        84.13934054, 112.79966058, 111.47171382, 122.45976331,
        54.81572387,  79.13174188,  65.47015201,  50.80872709,
        75.85973745,  77.36754425,  91.84766219,  83.15248917,
        91.81979153])
```

In [24]:
```
from sklearn.metrics import mean_absolute_error
```

In [25]:
```
mean_absolute_error(test['Car sales'], future_forecast)
```

Out[25]:
```
29.186078315622016
```

In [26]:
```
future_forecast = pd.DataFrame(future_forecast,
                               index = test.index,
                               columns=['Prediction'])
df_merge = test.join(future_forecast)
df_merge.tail()
```

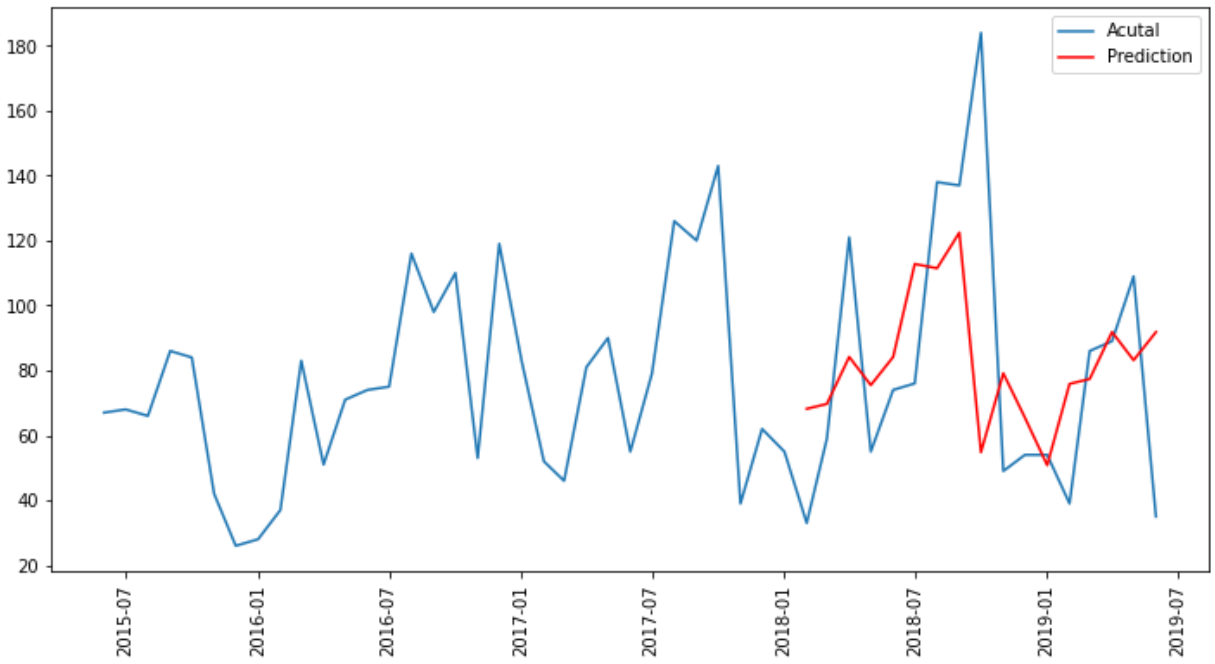
Out[26]:

Date	Car sales	Prediction
2019-02-01	39	75.859737
2019-03-01	86	77.367544
2019-04-01	89	91.847662
2019-05-01	109	83.152489
2019-06-01	35	91.819792

Bước 4: Trực quan hóa dữ liệu

In [27]:
```
plt.plot(test, label='Actual')
plt.plot(future_forecast, label='Prediction')
plt.xticks(rotation='vertical')
plt.legend()
plt.show()
```

In [28]:
```
plt.figure(figsize=(12,6))
plt.plot(data, label='Acutal')
plt.plot(future_forecast, label='Prediction', color='red')
plt.xticks(rotation='vertical')
plt.legend()
plt.show()
```

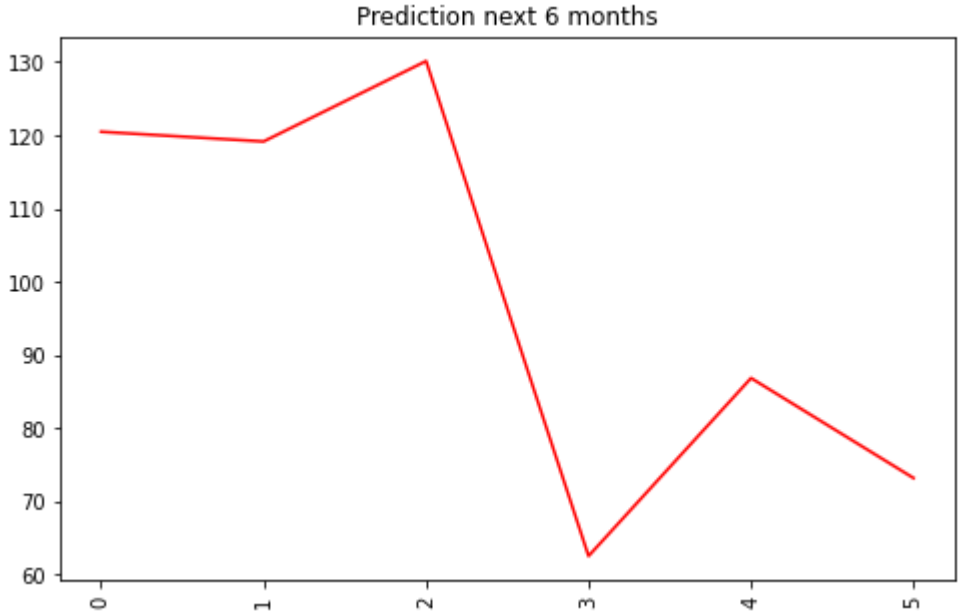


Dự đoán 6 tháng tiếp theo

In [29]:
```
future_forecast = stepwise_model.predict(n_periods=len(test)+6)
future_forecast
```

Out[29]:
```
array([ 68.17928646,  69.68709326,  84.1672112 ,  75.47203818,
        84.13934054, 112.79966058, 111.47171382, 122.45976331,
        54.81572387,  79.13174188,  65.47015201,  50.80872709,
        75.85973745,  77.36754425,  91.84766219,  83.15248917,
        91.81979153, 120.48011157, 119.15216481, 130.1402143 ,
        62.49617486,  86.81219287,  73.150603  ])
```

In [30]:
```
plt.figure(figsize=(8,5))
plt.plot(future_forecast[len(test):], color='red')
plt.xticks(rotation='vertical')
plt.title("Prediction next 6 months")
plt.show()
```



In [31]:
```
future_forecast[len(test):]
```

Out[31]:
```
array([120.48011157, 119.15216481, 130.1402143 ,  62.49617486,
        86.81219287,  73.150603  ])
```

In [32]:
```
months = pd.date_range('2019-07-01','2019-12-01',
              freq='MS').strftime("%Y-%m-%d").tolist()
```

In [33]:
```
months
```

Out[33]:
```
['2019-07-01',
 '2019-08-01',
 '2019-09-01',
 '2019-10-01',
 '2019-11-01',
 '2019-12-01']
```

In [34]:
```
new_predict = pd.DataFrame({
    'DATE' : months,
    'Energy Production': future_forecast[len(test):]}
    )
new_predict
```

Out[34]:

	DATE	Energy Production
0	2019-07-01	120.480112
1	2019-08-01	119.152165
2	2019-09-01	130.140214
3	2019-10-01	62.496175
4	2019-11-01	86.812193
5	2019-12-01	73.150603