Chapter 18: Demo Time Series với PMDARIMA

In [ ]:
```
from google.colab import drive
drive.mount("/content/gdrive", force_remount=True)
%cd '/content/gdrive/My Drive/LDS6_MachineLearning/practice/Chapter18_ARIMA/'
```

```
Mounted at /content/gdrive
/content/gdrive/My Drive/LDS6_MachineLearning/practice/Chapter18_ARIMA
```

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

In [ ]:
```
data = pd.read_csv("electric_production.csv",index_col=0)
data.head()
```

Out[3]:

DATE	IPG2211A2N
1939-01-01	3.3842
1939-02-01	3.4100
1939-03-01	3.4875
1939-04-01	3.5133
1939-05-01	3.5133

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

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

Out[5]:
```
DatetimeIndex(['1939-01-01', '1939-02-01', '1939-03-01', '1939-04-01',
               '1939-05-01', '1939-06-01', '1939-07-01', '1939-08-01',
               '1939-09-01', '1939-10-01',
               ...
               '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'],
              dtype='datetime64[ns]', name='DATE', length=956, freq=None)
```

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

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 956 entries, 1939-01-01 to 2018-08-01
Data columns (total 1 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   IPG2211A2N  956 non-null    float64
dtypes: float64(1)
memory usage: 14.9 KB
```

In [ ]:
```
data.columns = ['Energy Production']
```

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

Out[8]:

DATE	Energy Production
1939-01-01	3.3842
1939-02-01	3.4100
1939-03-01	3.4875
1939-04-01	3.5133
1939-05-01	3.5133

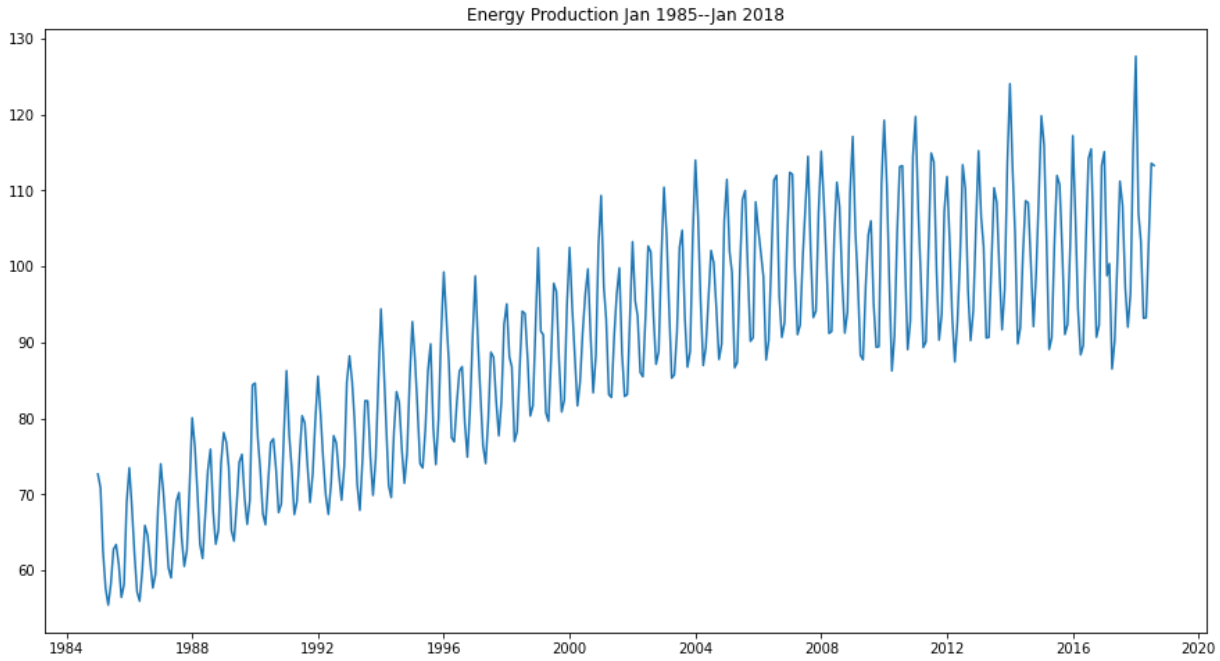
In [ ]:
```
import matplotlib.pyplot as plt
```

In [ ]:
```
data_1985 = data[data.index.year >=int(1985)]
data_1985.head()
```

Out[10]:

DATE	Energy Production
1985-01-01	72.6803
1985-02-01	70.8479
1985-03-01	62.6166
1985-04-01	57.6106
1985-05-01	55.4467

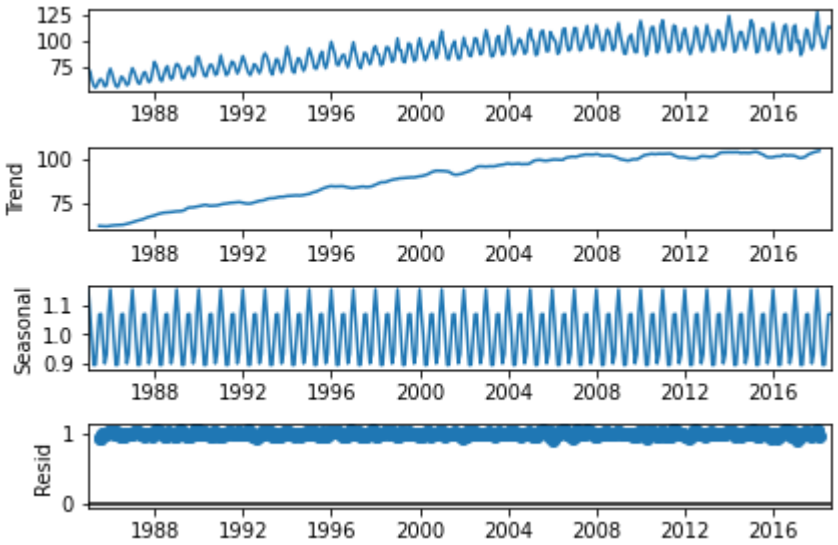
In [ ]:
```
plt.figure(figsize=(15,8))
plt.plot(data_1985)
plt.title("Energy Production Jan 1985--Jan 2018")
plt.show()
```



In [ ]:
```
from statsmodels.tsa.seasonal import seasonal_decompose
result = seasonal_decompose(data_1985, model='multiplicative')
result
```

Out[12]:
```
<statsmodels.tsa.seasonal.DecomposeResult at 0x7f6a44ffe4e0>
```

In [ ]:
```
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

Cài pip install pmdarima

In [ ]:
```
! pip install pmdarima
```

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

In [ ]:
```
stepwise_model = auto_arima(data_1985, 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=1827.308, Time=3.77 sec
 ARIMA(0,1,0)(0,1,0)[12]             : AIC=2055.116, Time=0.05 sec
 ARIMA(1,1,0)(1,1,0)[12]             : AIC=1980.491, Time=0.30 sec
 ARIMA(0,1,1)(0,1,1)[12]             : AIC=1876.298, Time=0.74 sec
 ARIMA(2,1,2)(0,1,1)[12]             : AIC=1826.789, Time=2.88 sec
 ARIMA(2,1,2)(0,1,0)[12]             : AIC=inf, Time=1.28 sec
 ARIMA(2,1,2)(0,1,2)[12]             : AIC=1826.372, Time=8.55 sec
 ARIMA(2,1,2)(1,1,2)[12]             : AIC=1825.064, Time=10.10 sec
 ARIMA(2,1,2)(2,1,2)[12]             : AIC=1812.125, Time=13.18 sec
 ARIMA(2,1,2)(2,1,1)[12]             : AIC=1815.973, Time=9.63 sec
 ARIMA(1,1,2)(2,1,2)[12]             : AIC=1810.891, Time=9.90 sec
 ARIMA(1,1,2)(1,1,2)[12]             : AIC=1824.617, Time=8.92 sec
 ARIMA(1,1,2)(2,1,1)[12]             : AIC=1815.075, Time=5.85 sec
 ARIMA(1,1,2)(1,1,1)[12]             : AIC=1827.134, Time=2.80 sec
 ARIMA(0,1,2)(2,1,2)[12]             : AIC=1815.740, Time=7.54 sec
 ARIMA(1,1,1)(2,1,2)[12]             : AIC=1809.104, Time=9.20 sec
 ARIMA(1,1,1)(1,1,2)[12]             : AIC=1824.331, Time=6.14 sec
 ARIMA(1,1,1)(2,1,1)[12]             : AIC=1814.650, Time=4.99 sec
 ARIMA(1,1,1)(1,1,1)[12]             : AIC=1826.608, Time=1.84 sec
 ARIMA(0,1,1)(2,1,2)[12]             : AIC=1850.176, Time=7.53 sec
 ARIMA(1,1,0)(2,1,2)[12]             : AIC=1878.105, Time=6.46 sec
 ARIMA(2,1,1)(2,1,2)[12]             : AIC=1810.967, Time=11.37 sec
 ARIMA(0,1,0)(2,1,2)[12]             : AIC=1901.687, Time=4.60 sec
 ARIMA(2,1,0)(2,1,2)[12]             : AIC=1850.887, Time=7.69 sec
 ARIMA(1,1,1)(2,1,2)[12] intercept   : AIC=1811.125, Time=18.21 sec

Best model:  ARIMA(1,1,1)(2,1,2)[12]
Total fit time: 163.562 seconds
```

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

```
1809.1044498637566
```

In [ ]:
```
train = data.loc['1985-01-01':'2015-01-01'] # 1/1985 => 12/2014
test = data.loc['2015-01-01':] # 1/2015 -> het
```

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

Out[19]:

DATE	Energy Production
2015-01-01	119.8260
2015-02-01	116.0253
2015-03-01	103.9265
2015-04-01	89.0847
2015-05-01	90.6408

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

Out[20]:
```
44
```

Bước 2: Fit mô hình

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

Out[21]:
```
ARIMA(maxiter=50, method='lbfgs', order=(1, 1, 1), out_of_sample_size=0,
      scoring='mse', scoring_args={}, seasonal_order=(2, 1, 2, 12),
      start_params=None, suppress_warnings=True, trend=None,
      with_intercept=False)
```

Bước 3: Dự đoán kết quả

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

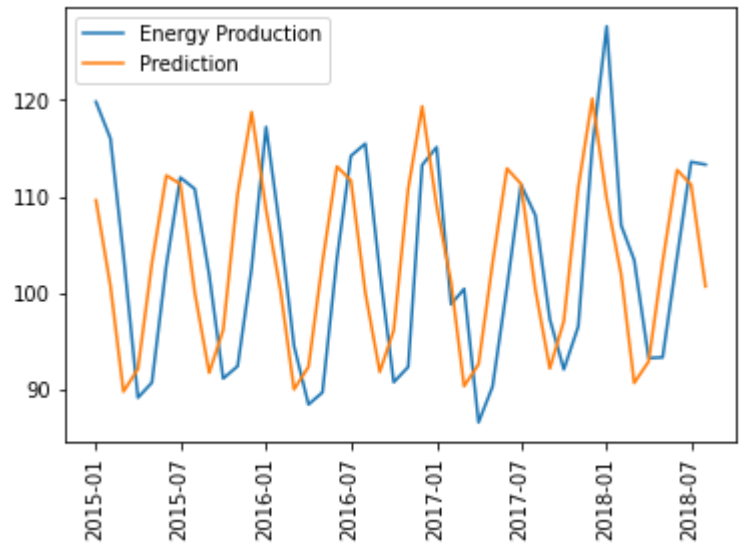
In [ ]:
```
future_forecast
```

Out[23]:
```
array([109.60539041, 100.70625686,  89.71135078,  92.06894014,
       103.03753549, 112.17579692, 111.30475581, 100.01379224,
        91.6640978 ,  96.12871534, 110.27753208, 118.76801355,
       108.64360461, 100.21331396,  89.93910629,  92.28890669,
       103.14679204, 113.09871384, 111.6633647 ,  99.99294187,
        91.7564248 ,  96.10083262, 110.79628686, 119.35702155,
       109.05568117, 101.0755135 ,  90.27862081,  92.55438726,
       103.04897326, 112.89399336, 111.30884567, 100.28326835,
        92.1216138 ,  97.03966078, 110.92733435, 120.15301699,
       109.7300375 , 101.87582216,  90.60724194,  92.84446357,
       103.16853737, 112.74569206, 111.2236608 , 100.648906  ])
```

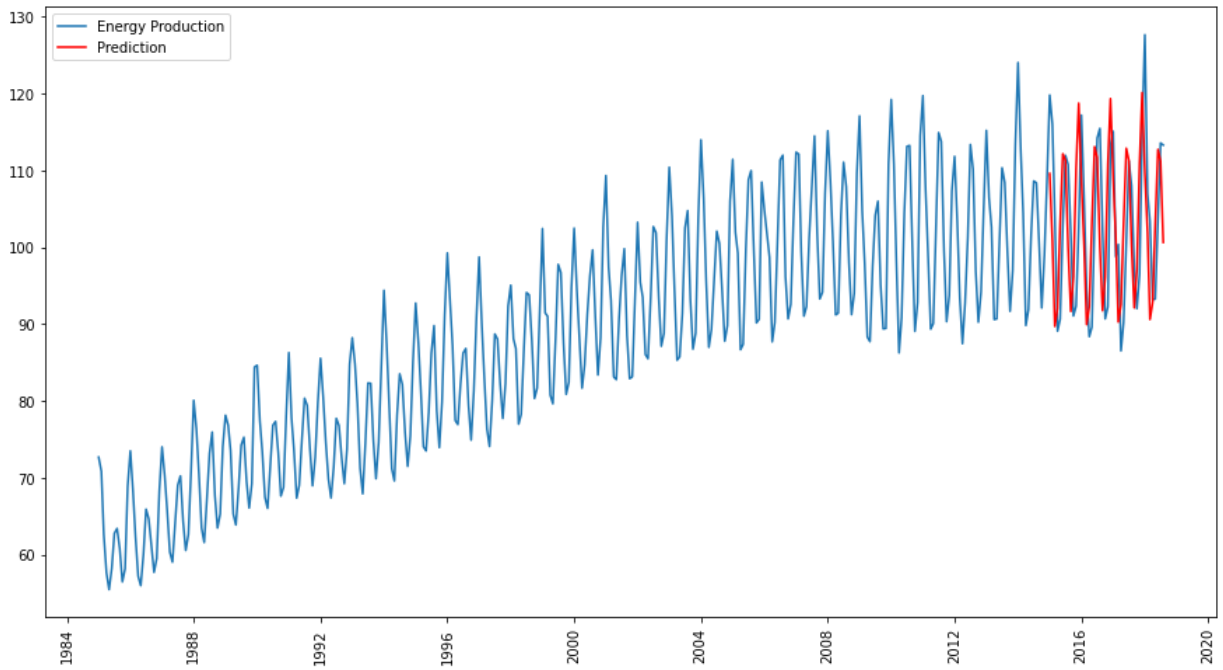
In [ ]:
```
future_forecast = pd.DataFrame(future_forecast,index = test.index,columns=['Pred
```

Bước 4: Trực quan hóa dữ liệu

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



In [ ]:
```
plt.figure(figsize=(15,8))
plt.plot(data_1985, label='Energy Production')
plt.plot(future_forecast, label='Prediction', color='red')
plt.xticks(rotation='vertical')
plt.legend()
plt.show()
```

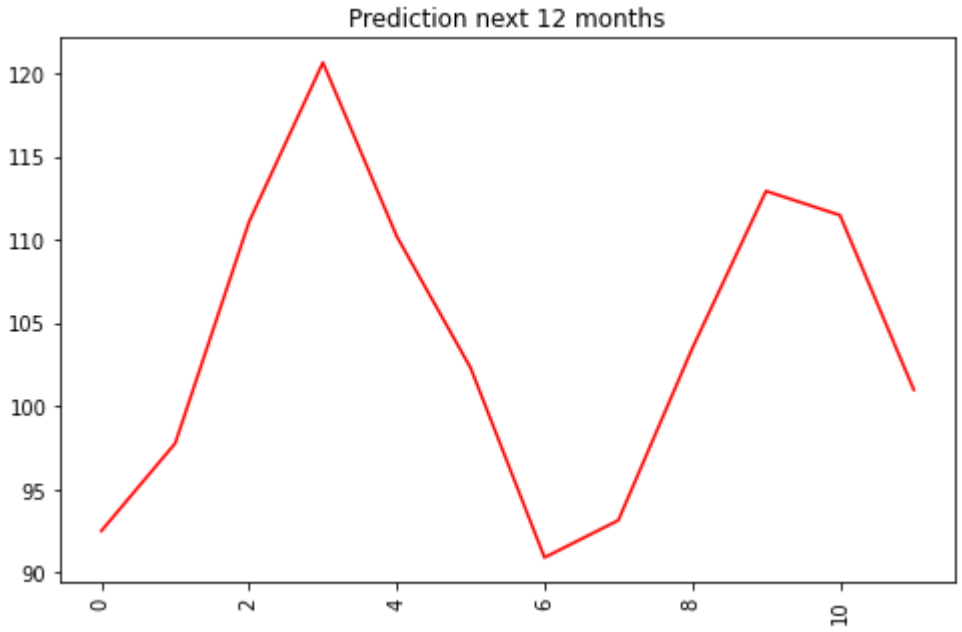


Dự đoán 12 tháng tiếp theo

In [ ]:
```
future_forecast = stepwise_model.predict(n_periods=len(test)+12) # 44 thang + 12
future_forecast
```

Out[27]:
```
array([109.60539041, 100.70625686,  89.71135078,  92.06894014,
       103.03753549, 112.17579692, 111.30475581, 100.01379224,
        91.6640978 ,  96.12871534, 110.27753208, 118.76801355,
       108.64360461, 100.21331396,  89.93910629,  92.28890669,
       103.14679204, 113.09871384, 111.6633647 ,  99.99294187,
        91.7564248 ,  96.10083262, 110.79628686, 119.35702155,
       109.05568117, 101.0755135 ,  90.27862081,  92.55438726,
       103.04897326, 112.89399336, 111.30884567, 100.28326835,
        92.1216138 ,  97.03966078, 110.92733435, 120.15301699,
       109.7300375 , 101.87582216,  90.60724194,  92.84446357,
       103.16853737, 112.74569206, 111.2236608 , 100.648906  ,
        92.49914218,  97.76231778, 111.06691685, 120.63874545,
       110.19465452, 102.2936354 ,  90.90115245,  93.13630657,
       103.46044181, 112.91837614, 111.46945151, 100.97574542])
```

In [ ]:
```
plt.figure(figsize=(8,5))
plt.plot(future_forecast[len(test):], color='red')
plt.xticks(rotation='vertical')
plt.title("Prediction next 12 months")
plt.show()
```



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

Out[29]:
```
array([ 92.49914218,  97.76231778, 111.06691685, 120.63874545,
       110.19465452, 102.2936354 ,  90.90115245,  93.13630657,
       103.46044181, 112.91837614, 111.46945151, 100.97574542])
```

In [ ]:
```
months = pd.date_range('2018-09-01','2019-08-01',
              freq='MS').strftime("%Y-%m-%d").tolist()
```

In [ ]:
```
months
```

Out[32]:
```
['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',
 '2019-07-01',
 '2019-08-01']
```

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

Out[33]:

	DATE	Energy Production
0	2018-09-01	92.499142
1	2018-10-01	97.762318
2	2018-11-01	111.066917
3	2018-12-01	120.638745
4	2019-01-01	110.194655
5	2019-02-01	102.293635
6	2019-03-01	90.901152
7	2019-04-01	93.136307
8	2019-05-01	103.460442
9	2019-06-01	112.918376
10	2019-07-01	111.469452
11	2019-08-01	100.975745

In [ ]:
```
# Source: https://medium.com/@josemarcialportilla/using-python-and-auto-arima-to
```