Chapter 18: Demo Time Series với PMDARIMA

```
from google.colab import drive
drive.mount("/content/gdrive", force_remount=True)

path = '/content/gdrive/My Drive/LDS6_MachineLearning/'
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=9473189

Enter your authorization code:
··········
Mounted at /content/gdrive
```

```
import pandas as pd

data = pd.read_csv(path +"practice/Chapter18_ARIMA/electric_production.csv",index_col=0)
data.head()
```

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

```
data.index = pd.to_datetime(data.index)

data.index
```

```
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)
```

```
data.info()
```

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

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

```
data.head()
```

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

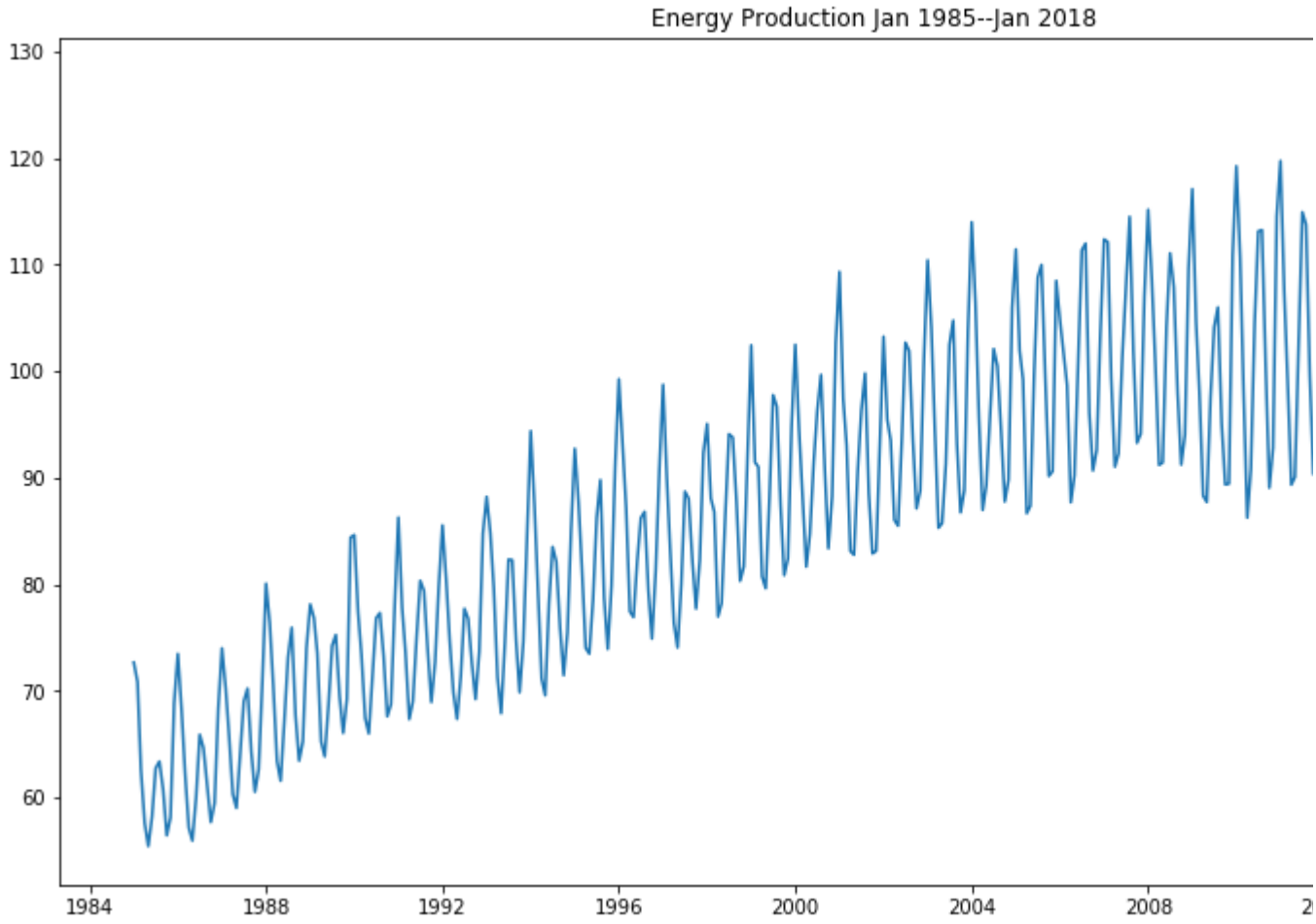
```
import matplotlib.pyplot as plt
```

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

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

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

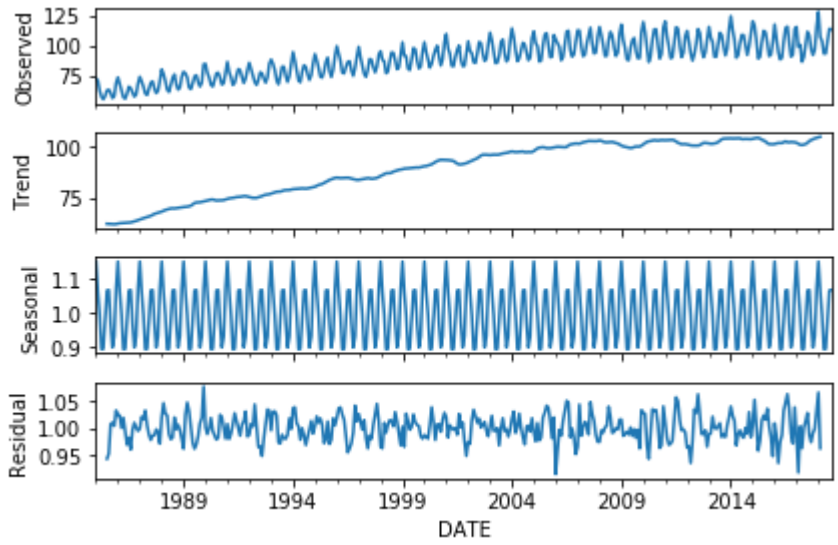
```
/usr/local/lib/python3.6/dist-packages/pandas/plotting/_matplotlib/converter.py:103: Fut

To register the converters:
        >>> from pandas.plotting import register_matplotlib_converters
        >>> register_matplotlib_converters()
  warnings.warn(msg, FutureWarning)
```




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

```
<statsmodels.tsa.seasonal.DecomposeResult at 0x7fdce2489358>
```

```
result.plot()
plt.show()
```



- Với kết quả trên, ta có thể thấy rõ tính seasonal component của data, và cũng có thể thấy xu hướ

- Trend có thể lên hoặc xuống và có thể tuyến tính hoặc phi tuyến tính. Cần phải hiểu tập dữ liệu đ 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 nhi

Áp dụng auto_arima để xây dựng mô hình

Cài pip install pmdarima

```
! pip install pmdarima
```

```
Collecting pmdarima
  Downloading https://files.pythonhosted.org/packages/d8/b7/708f4c8c0aef0761ab2d1d638b2a
     |████████████████████████████████| 1.4MB 2.7MB/s
Requirement already satisfied: numpy>=1.16 in /usr/local/lib/python3.6/dist-packages (fr
Requirement already satisfied: Cython>=0.29 in /usr/local/lib/python3.6/dist-packages (f
Requirement already satisfied: statsmodels>=0.10.0 in /usr/local/lib/python3.6/dist-pack
Requirement already satisfied: pandas>=0.19 in /usr/local/lib/python3.6/dist-packages (f
Requirement already satisfied: pathlib in /usr/local/lib/python3.6/dist-packages (from p
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.6/dist-packages (from
Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.6/dist-packages (f
Requirement already satisfied: scipy>=1.3 in /usr/local/lib/python3.6/dist-packages (fro
Requirement already satisfied: scikit-learn>=0.19 in /usr/local/lib/python3.6/dist-packa
Requirement already satisfied: patsy>=0.4.0 in /usr/local/lib/python3.6/dist-packages (f
Requirement already satisfied: pytz>=2017.2 in /usr/local/lib/python3.6/dist-packages (f
Requirement already satisfied: python-dateutil>=2.6.1 in /usr/local/lib/python3.6/dist-p
Installing collected packages: pmdarima
Successfully installed pmdarima-1.5.1
```

```
from pmdarima.arima import auto_arima
```

```
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)
```

```
Fit ARIMA: order=(2, 1, 2) seasonal_order=(1, 1, 1, 12); AIC=3729.019, BIC=3767.811, Fit
Fit ARIMA: order=(0, 1, 0) seasonal_order=(0, 1, 0, 12); AIC=4238.962, BIC=4248.660, Fit
Fit ARIMA: order=(1, 1, 0) seasonal_order=(1, 1, 0, 12); AIC=4058.517, BIC=4077.914, Fit
Fit ARIMA: order=(0, 1, 1) seasonal_order=(0, 1, 1, 12); AIC=3859.889, BIC=3879.286, Fit
Fit ARIMA: order=(0, 1, 0) seasonal_order=(0, 1, 0, 12); AIC=4236.967, BIC=4241.816, Fit
Fit ARIMA: order=(2, 1, 2) seasonal_order=(0, 1, 1, 12); AIC=3729.631, BIC=3763.574, Fit
Fit ARIMA: order=(2, 1, 2) seasonal_order=(1, 1, 0, 12); AIC=3889.170, BIC=3923.113, Fit
Fit ARIMA: order=(2, 1, 2) seasonal_order=(2, 1, 1, 12); AIC=3707.297, BIC=3750.938, Fit
Fit ARIMA: order=(2, 1, 2) seasonal_order=(2, 1, 0, 12); AIC=3776.649, BIC=3815.442, Fit
Fit ARIMA: order=(2, 1, 2) seasonal_order=(2, 1, 2, 12); AIC=3698.517, BIC=3747.007, Fit
Fit ARIMA: order=(2, 1, 2) seasonal_order=(1, 1, 2, 12); AIC=3723.199, BIC=3766.840, Fit
Fit ARIMA: order=(1, 1, 2) seasonal_order=(2, 1, 2, 12); AIC=3699.443, BIC=3743.084, Fit
Fit ARIMA: order=(2, 1, 1) seasonal_order=(2, 1, 2, 12); AIC=3700.355, BIC=3743.997, Fit
Fit ARIMA: order=(3, 1, 2) seasonal_order=(2, 1, 2, 12); AIC=3698.900, BIC=3752.240, Fit
Fit ARIMA: order=(2, 1, 3) seasonal_order=(2, 1, 2, 12); AIC=3699.510, BIC=3752.849, Fit
Fit ARIMA: order=(1, 1, 1) seasonal_order=(2, 1, 2, 12); AIC=3700.373, BIC=3739.166, Fit
Fit ARIMA: order=(1, 1, 3) seasonal_order=(2, 1, 2, 12); AIC=3697.520, BIC=3746.011, Fit
Fit ARIMA: order=(2, 1, 4) seasonal_order=(2, 1, 2, 12); AIC=3725.147, BIC=3783.336, Fit
Fit ARIMA: order=(1, 1, 3) seasonal_order=(1, 1, 2, 12); AIC=3728.673, BIC=3772.314, Fit
Near non-invertible roots for order (1, 1, 3)(1, 1, 2, 12); setting score to inf (at lea
Fit ARIMA: order=(1, 1, 3) seasonal_order=(2, 1, 1, 12); AIC=3706.714, BIC=3750.356, Fit
Fit ARIMA: order=(1, 1, 3) seasonal_order=(1, 1, 1, 12); AIC=3729.035, BIC=3767.827, Fit
Fit ARIMA: order=(0, 1, 3) seasonal_order=(2, 1, 2, 12); AIC=3705.653, BIC=3749.294, Fit
Fit ARIMA: order=(1, 1, 4) seasonal_order=(2, 1, 2, 12); AIC=3699.637, BIC=3752.976, Fit
Fit ARIMA: order=(0, 1, 2) seasonal_order=(2, 1, 2, 12); AIC=3712.344, BIC=3751.136, Fit
Fit ARIMA: order=(0, 1, 4) seasonal_order=(2, 1, 2, 12); AIC=3699.765, BIC=3748.256, Fit
Total fit time: 876.869 seconds
```

```
print(stepwise_model.aic())
```

```
3697.5203076307916
```

```
train = data.loc['1985-01-01':'2016-12-01']
test = data.loc['2015-01-01':]
```

```
test.head()
```

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

```
len(test)
```

```
44
```

Bước 2: Fit mô hình

```
stepwise_model.fit(train)
```

```
ARIMA(maxiter=50, method='lbfgs', order=(1, 1, 3), out_of_sample_size=0,
      scoring='mse', scoring_args=None, seasonal_order=(2, 1, 2, 12),
      start_params=None, suppress_warnings=True, trend=None,
      with_intercept=True)
```

Bước 3: Dự đoán kết quả

```
future_forecast = stepwise_model.predict(n_periods=len(test))
```

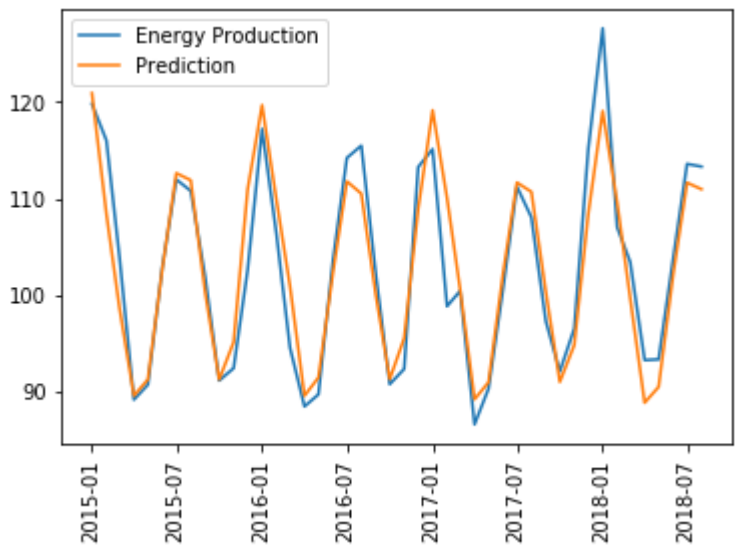
```
future_forecast
```

```
array([120.93236162, 108.39298529,  98.62944148,  89.49007019,
        91.18146281, 102.66662183, 112.6205039 , 111.88447352,
       100.22780387,  91.17875156,  95.06294083, 110.927545  ,
       119.69167182, 109.809806  , 100.87030382,  89.49000186,
        91.37698344, 102.22549745, 111.76577606, 110.49593295,
       100.00798446,  91.19900321,  95.47994631, 108.87655922,
       119.1567875 , 110.02918554, 100.57975752,  89.12318671,
        90.88524119, 102.12727277, 111.64649487, 110.67972937,
       100.29609669,  90.92590671,  94.75527157, 108.36238676,
       119.07489077, 109.57407639,  99.6420484 ,  88.76447797,
        90.40623397, 102.01188241, 111.63042831, 110.9545988 ])
```

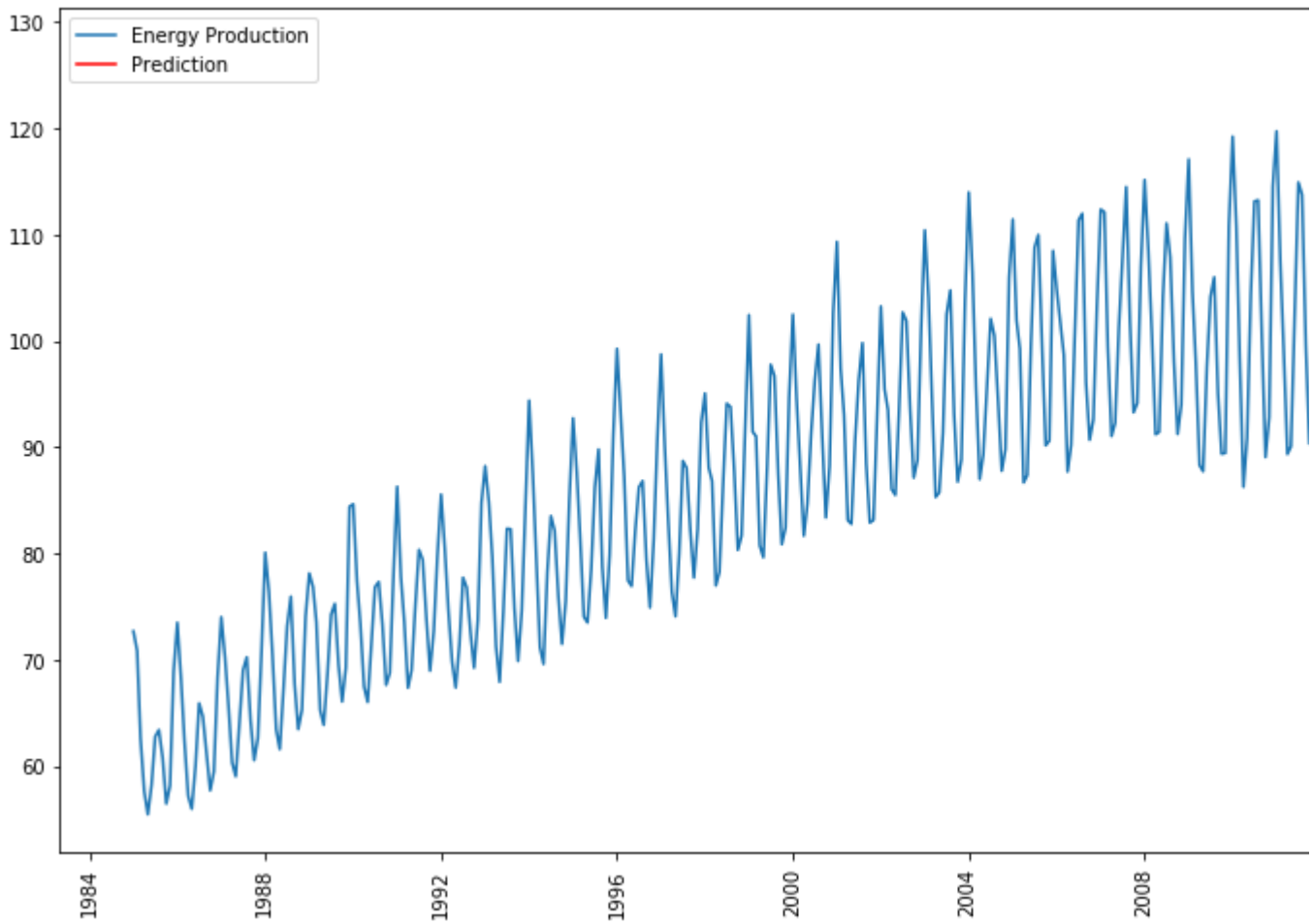
```
future_forecast = pd.DataFrame(future_forecast,index = test.index,columns=['Prediction'])
```

Bước 4: Trực quan hóa dữ liệu

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




```
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()
```



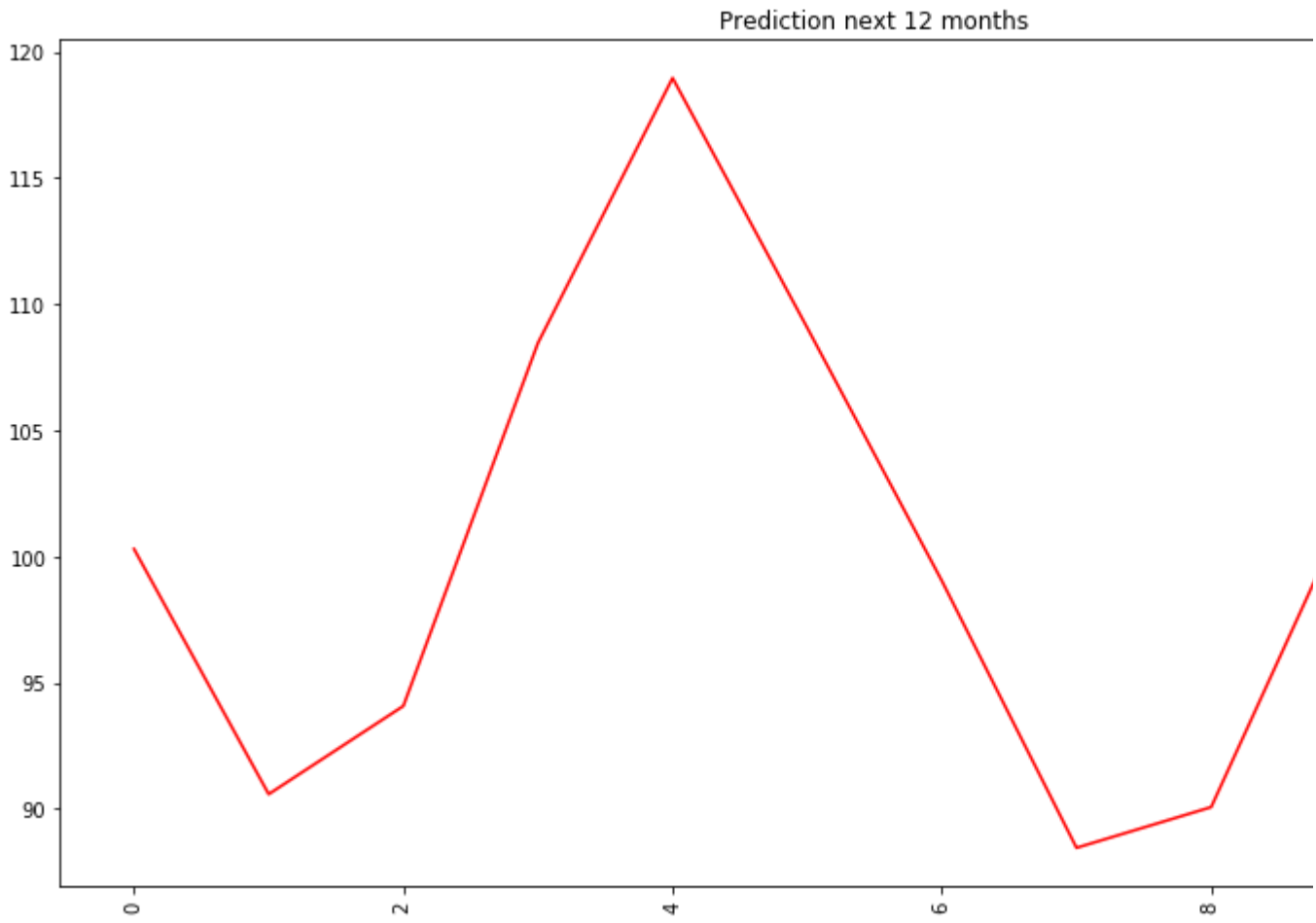
Dự đoán 12 tháng tiếp theo

```
future_forecast = stepwise_model.predict(n_periods=len(test)+12)
future_forecast
```

```
array([120.93236162, 108.39298529,  98.62944148,  89.49007019,
        91.18146281, 102.66662183, 112.6205039 , 111.88447352,
       100.22780387,  91.17875156,  95.06294083, 110.927545  ,
       119.69167182, 109.809806  , 100.87030382,  89.49000186,
        91.37698344, 102.22549745, 111.76577606, 110.49593295,
       100.00798446,  91.19900321,  95.47994631, 108.87655922,
       119.1567875 , 110.02918554, 100.57975752,  89.12318671,
        90.88524119, 102.12727277, 111.64649487, 110.67972937,
       100.29609669,  90.92590671,  94.75527157, 108.36238676,
       119.07489077, 109.57407639,  99.6420484 ,  88.76447797,
        90.40623397, 102.01188241, 111.63042831, 110.9545988 ,
       100.29286742,  90.5770431 ,  94.07086949, 108.45990025,
       118.95815124, 109.06095719,  99.02959086,  88.45084663,
        90.06344853, 101.73251784, 111.40035694, 110.77752961])
```

```
plt.figure(figsize=(15,8))
plt.plot(future_forecast[len(test):], color='red')
plt.xticks(rotation='vertical')
```

```
plt.title("Prediction next 12 months")
plt.show()
```




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

```
array([100.29286742,  90.5770431 ,  94.07086949, 108.45990025,
       118.95815124, 109.06095719,  99.02959086,  88.45084663,
        90.06344853, 101.73251784, 111.40035694, 110.77752961])
```

```
# Source: https://medium.com/@josemarcialportilla/using-python-and-auto-arima-to-forecast-sea
```