TRƯỜNG ĐẠI HỌC THUỶ LỢI
**KHOA CÔNG NGHỆ THÔNG TIN**

# Đề 3: Phân tích Sarima, Arimax

Đỗ Văn Luật - 2151264668

Lớp: 63 TTNT

I. Phân tích mô hình Sarima

1. Định nghĩa

- Sarima là một mô hình dự báo chuỗi thời gian linh hoạt và được sử dụng rộng rãi. Đây là mô hình cải mở rộng của mô hình ARIMA, khi được thiết kế để xử lý dữ liệu chuỗi thời gian có yếu tố mùa vụ. SARIMA nắm bắt cả sự phụ thuộc ngắn hạn và dài hạn trong dữ liệu, khiến nó trở thành một công cụ mạnh mẽ để dự báo. Nó kết hợp các khái niệm về mô hình tự hồi quy (AR), mô hình tích hợp (I) và đường trung bình động (MA) với các thành phần theo mùa.

2. Ứng dụng

Mô hình Sarima được ứng dụng trong nhiều lĩnh vực khác nhau, bao gồm:

- Kinh tế : Dự đoán các chỉ số kinh tế như lạm phát và GDP.
- Bán lẻ : Dự báo doanh thu và nhu cầu đối với các sản phẩm theo mùa.
- Năng lượng : Dự đoán mức tiêu thụ và nhu cầu năng lượng.
- Chăm sóc sức khỏe : Lập mô hình tiếp nhận bệnh nhân và bùng phát dịch bệnh.
- Tài chính : Dự đoán giá cổ phiếu và xu hướng thị trường.

3. Các thành phần của Sarima

- S (Season): thể hiện tính thời vụ, đề cập đến các mẫu lặp lại trong dữ liệu. Có thể là hàng ngày, hàng tháng, hàng năm hoặc bất kỳ khoảng thời gian thường xuyên nào khác. Điểm mạnh của nó là xác định và mô hình hóa thành phần theo mùa

- AR(Autoregressive): biểu thị thành phần tự hồi quy, mô hình hóa mối quan hệ giữa điểm dữ liệu hiện tại và các giá trị trong quá khứ của nó. Nó nắm bắt sự tự tương quan của dữ liệu, nghĩa là mức độ tương quan của dữ liệu với chính nó theo thời gian.

- I (Integrated): biểu thị sự khác biệt, giúp chuyển đổi dữ liệu không cố định thành dữ liệu cố định. Tính dừng là rất quan trọng đối với mô hình chuỗi thời gian. Thành phần tích hợp đo lường xem cần có bao nhiêu sự khác biệt để đạt được tính ổn định.
- MA (Moving Average): là đường trung bình động, mô hình hóa sự phụ thuộc giữa điểm dữ liệu hiện tại và các lỗi dự đoán trong quá khứ. Nó giúp thu được tiếng ồn ngắn hạn trong dữ liệu.

- Biểu diễn toán học

SARIMA(p, d, q)(P, D, Q, s) với:

- AR(p): Thành phần autoregressive của bậc p
- MA(q): Thành phần moving average của bậc q
- I(d): Thành phần tích hợp của bậc d
- Seasonal AR(P): Thành phần autoregressive theo mùa của bậc P
- MA(Q): Thành phần moving average theo mùa của bậc Q
- Seasonal I(D): Thành phần tích hợp theo mùa của bậc D
- s: Chu kỳ mùa

Công thức toán học:

$$(1-\phi_1 B)(1-\Phi_1 B_s)(1-B)(1-B_s)y_t = (1+\theta_1 B)(1+\Theta_1 B_s)\varepsilon_t$$

Trong đó,

- $y_t$ là chuỗi thời gian quan sát tại thời điểm $t$,
- $B$ là toán tử dịch chuyển ngược, đại diện cho toán tử độ trễ (tức là $By_t = y_{t-1}$),
- $\phi_1$ là hệ số autoregressive không theo mùa,
- $\Phi_1$ là hệ số autoregressive theo mùa,
- $\theta_1$ là hệ số moving average không theo mùa,
- $\Theta_1$ là hệ số moving average theo mùa,
- $s$ là chu kỳ mùa,
- $\varepsilon_t$ là thuật ngẫu nhiên trắng tại thời điểm $t$.

Phân tích các thành phần của mô hình SARIMA:

- Thành phần tự hồi quy phi AR nắm bắt mối quan hệ giữa quan sát hiện tại và một số quan sát trễ nhất định (các giá trị trước đó trong chuỗi thời gian). Toán tử dịch chuyển ngược được sử dụng trong phân tích chuỗi thời gian để dịch chuyển chuỗi thời gian về phía sau một khoảng thời gian nhất định. Bậc của thành phần tự hồi quy, ký hiệu là $p$: xác định số lượng các giá trị quá khứ được xem xét trong mô hình.

- Thành phần tự hồi quy theo mùa SAR nắm bắt mối quan hệ giữa quan sát hiện tại và một số quan sát trễ nhất định tại các khoảng thời gian theo mùa. Toán tử dịch chuyển ngược theo mùa được áp dụng cho các quan sát trễ theo mùa.

- Thành phần khác biệt phi mùa được sử dụng để làm cho chuỗi thời gian trở nên dừng bằng cách lấy hiệu của nó một số lần nhất định.d.

- Thành phần khác biệt theo mùa (Seasonal Differencing) được sử dụng để làm cho chuỗi thời gian trở nên dừng bằng cách lấy hiệu của nó tại các khoảng thời gian theo mùa

- Chuỗi thời gian quan sát (Observed Time Series) đại diện cho dữ liệu lịch sử mà chúng ta có và muốn dự báo.
- Thành phần trung bình động (MA - Moving Average) nắm bắt mối quan hệ giữa quan sát hiện tại và các sai số dự báo từ mô hình trung bình động áp dụng cho các quan sát trễ.
- Thành phần trung bình động theo mùa (SMA) nắm bắt mối quan hệ giữa quan sát hiện tại và các sai số dự báo từ mô hình trung bình động áp dụng cho các quan sát trễ theo mùa.
- Thuật ngẫu nhiên (Error Term) đại diện cho nhiễu ngẫu nhiên hoặc biến động không giải thích được trong chuỗi thời gian.

4. Cách xây dựng mô hình Arimax:
- Khám phá dữ liệu
- Kiểm tra tính dừng
- Lấy sai phân
- Lựa chọn các biến ngoại sinh
- Xác định thứ tự của mô hình
- Ước lượng các tham số
- Kiểm định mô hình
- Dự báo

II.Phân tích mô hình Arimax

1. Định nghĩa
- Là một dạng mở rộng của model ARIMA. Mô hình cũng dựa trên giải định về mối quan hệ tuyến tính giữa giá trị và phương sai trong quá khứ với giá trị hiện tại và sử dụng phương trình hồi qui tuyến tính được suy ra từ mối quan hệ trong quá khứ nhằm dự báo tương lai. Nhờ đó mà cải thiện được khả năng dự báo

2. Các thành phần của mô hình Arimax

- AR: Thành phần hồi quy tự hồi quy, thể hiện mối quan hệ giữa giá trị hiện tại và một số giá trị trong quá khứ của nó
- I : Thành phần tích phân, thể hiện được số lần lấy sai phân của chuỗi thời gian để làm cho nó trở nên tĩnh
- MA : Thành phần trung bình động, thể hiện mô hình hóa mối quan hệ giữa giá trị hiện tại và lỗi trong dự báo giá trị trước đó
- X : Các biến ngoại sinh là những biến số không phải là giá trị trễ của biến phụ thuộc nhưng có ảnh hưởng đến nó

3. Cách xây dựng mô hình Arimax:

- Khám phá dữ liệu
- Kiểm tra tính dừng
- Lấy sai phân
- Lựa chọn các biến ngoại sinh
- Xác định thứ tự của mô hình
- Ước lượng các tham số
- Kiểm định mô hình
- Dự báo

III.Áp dụng vào bài toán

Đầu tiên, ta tiền xử lý dữ liệu

- Đọc file data:

```
data = pd.read_csv('data-kiem-tra-2.csv')
data.head()
```

✓ 0.2s

| | date | truong_1 | truong_2 | truong_3 | truong_4 | truong_5 |
|---|---|---|---|---|---|---|
| 0 | 10.05.2013 | 4 | 58 | 3773 | 299.0 | 1 |
| 1 | 26.05.2013 | 4 | 58 | 3768 | 249.0 | 1 |
| 2 | 19.05.2013 | 4 | 58 | 4036 | 419.0 | 1 |
| 3 | 25.05.2013 | 4 | 58 | 12878 | 149.0 | 1 |
| 4 | 15.05.2013 | 4 | 58 | 12885 | 148.0 | 1 |

```
data.dtypes
```

✓ 0.0s

```
date          object
truong_1       int64
truong_2       int64
truong_3       int64
truong_4     float64
truong_5       int64
dtype: object
```

- Ở cột date, có dạng object, cần chuyển nó về dạng datetime

```
data['date'] = pd.to_datetime(data['date'], format='%d.%m.%Y')
```

✓ 0.0s

- Kiểm tra các Missing Values:

```
missing_values = data.isnull().sum()
print(missing_values)
```
✓ 0.0s

```
date        0
truong_1    0
truong_2    0
truong_3    0
truong_4    0
truong_5    0
dtype: int64
```

- Chuẩn hóa các cột số liệu bằng StandardScaler

```
scaler = StandardScaler()
data[['truong_1', 'truong_2', 'truong_3', 'truong_4', 'truong_5']] = scaler.fit_transform(
    data[['truong_1', 'truong_2', 'truong_3', 'truong_4', 'truong_5']]
)
```

- Kiểm tra trùng lặp ngày và  trung bình các giá trị cho mỗi ngày trùng lặp

```
duplicated_dates = data['date'].duplicated().sum()
print(f"Number of duplicated dates: {duplicated_dates}")

data_aggregated = data.groupby('date').mean().reset_index(
```
✓ 0.0s

```
Number of duplicated dates: 549824
```

- Kiểm tra lại dữ liệu sau khi loại bỏ các trùng lặp và tính trung bình

```
Data after aggregating duplicates:
        date  truong_1  truong_2  truong_3  truong_4  truong_5
0 2013-05-01 -1.785036 -0.224808 -0.007645 -0.078485 -0.074506
1 2013-05-02 -1.785036 -0.190042  0.037138 -0.072496 -0.070146
2 2013-05-03 -1.785036 -0.187567 -0.025286 -0.043853 -0.081470
3 2013-05-04 -1.785036 -0.126539 -0.022079 -0.092166 -0.086247
4 2013-05-05 -1.785036 -0.106864  0.107089 -0.124855 -0.092037
```

- Thiết lập cột 'date' làm chỉ số

```
# Thiết lập cột 'date' làm chỉ số
data_aggregated.set_index('date', inplace=True)
```
✓ 0.0s

- Thay các giá trị bị thiếu bằng phương pháp forward fill

```
data_filled = data_aggregated.reindex(all_dates).fillna(method='ffill')

print("Data after filling missing values:")
print(data_filled.head(10))
```
✓ 0.0s

```
Data after filling missing values:
            truong_1  truong_2  truong_3  truong_4  truong_5
2013-05-01 -1.785036 -0.224808 -0.007645 -0.078485 -0.074506
2013-05-02 -1.785036 -0.190042  0.037138 -0.072496 -0.070146
2013-05-03 -1.785036 -0.187567 -0.025286 -0.043853 -0.081470
2013-05-04 -1.785036 -0.126539 -0.022079 -0.092166 -0.086247
2013-05-05 -1.785036 -0.106864  0.107089 -0.124855 -0.092037
2013-05-06 -1.785036 -0.427208  0.021395 -0.036276 -0.095228
2013-05-07 -1.785036 -0.212068 -0.030525 -0.064728 -0.094957
2013-05-08 -1.785036 -0.232319  0.053717  0.004431 -0.083844
2013-05-09 -1.785036 -0.298793 -0.039356 -0.071782 -0.075685
2013-05-10 -1.785036 -0.168426  0.006531 -0.082714 -0.080281
```

- Chuẩn hóa lại các cột số liệu sau khi điền giá trị bị thiếu

```
# Chuẩn hóa lại các cột số liệu sau khi điền giá trị bị thiếu
data_filled[['truong_1', 'truong_2', 'truong_3', 'truong_4', 'truong_5']] = scaler.fit_transform(
    data_filled[['truong_1', 'truong_2', 'truong_3', 'truong_4', 'truong_5']]
)
print("Data after re-scaling:")
print(data_filled.head(10))
```
✓ 0.0s

```
Data after re-scaling:
            truong_1  truong_2  truong_3  truong_4  truong_5
2013-05-01 -1.497093 -1.552907 -0.259645 -1.050002 -0.785685
2013-05-02 -1.497093 -1.251577  0.364234 -0.980411 -0.740246
2013-05-03 -1.497093 -1.230127 -0.505398 -0.647600 -0.858254
2013-05-04 -1.497093 -0.701189 -0.460723 -1.208963 -0.908036
2013-05-05 -1.497093 -0.530660  1.338734 -1.588797 -0.968375
2013-05-06 -1.497093 -3.307142  0.144921 -0.559551 -1.001621
2013-05-07 -1.497093 -1.442485 -0.578384 -0.890149 -0.998799
2013-05-08 -1.497093 -1.618002  0.595200 -0.086568 -0.882996
2013-05-09 -1.497093 -2.194143 -0.701410 -0.972120 -0.797974
2013-05-10 -1.497093 -1.064235 -0.062158 -1.099140 -0.845862
```

- Chia dữ liệu với tập train và test

```
train = data_filled.iloc[:-30]
test = data_filled.iloc[-30:]
```

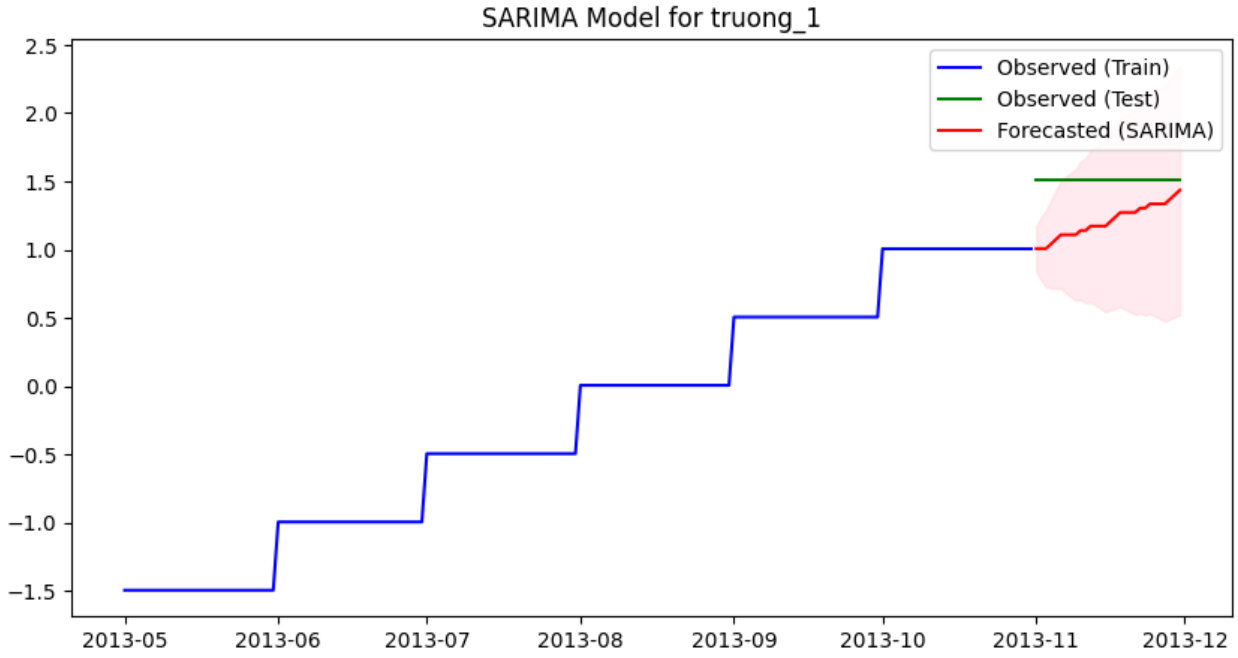
- **Dự báo cho mỗi cột với mô hình SARIMA**

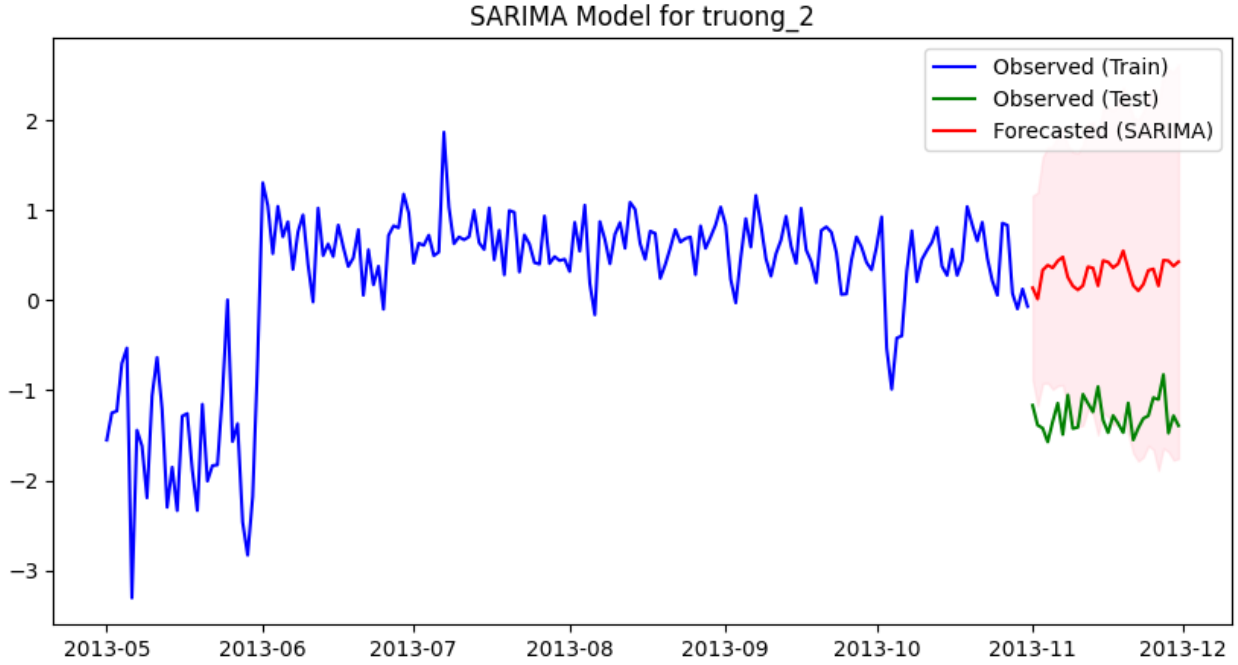
```
# Dự báo cho mỗi cột với mô hình SARIMA và ARIMAX
columns = ['truong_1', 'truong_2', 'truong_3', 'truong_4', 'truong_5']
for column in columns:
    # Mô hình
    sarima_model = sm.tsa.statespace.SARIMAX(train[column], order=(1, 1, 1), seasonal_order=(1, 1, 1, 12))
    sarima_result = sarima_model.fit()

    # Dự báo với mô hình SARIMA
    sarima_forecast = sarima_result.get_forecast(steps=30)
    sarima_predicted_mean = sarima_forecast.predicted_mean
    sarima_conf_int = sarima_forecast.conf_int()

    # Vẽ đồ thị kết quả dự báo SARIMA
    plt.figure(figsize=(10, 5))

    # Dữ liệu quan sát trong tập huấn luyện
    plt.plot(train.index, train[column], label='Observed (Train)', color='blue')

    # Dữ liệu quan sát trong tập kiểm tra
    plt.plot(test.index, test[column], label='Observed (Test)', color='green')

    # Dự báo của mô hình SARIMA
    plt.plot(test.index, sarima_predicted_mean, label='Forecasted (SARIMA)', color='red')

    # Tô màu cho khoảng tin cậy của dự báo
    plt.fill_between(test.index, sarima_conf_int.iloc[:, 0], sarima_conf_int.iloc[:, 1], color='pink', alpha=0.3)

    plt.legend()
    plt.title(f'SARIMA Model for {column}')
    plt.show()
```

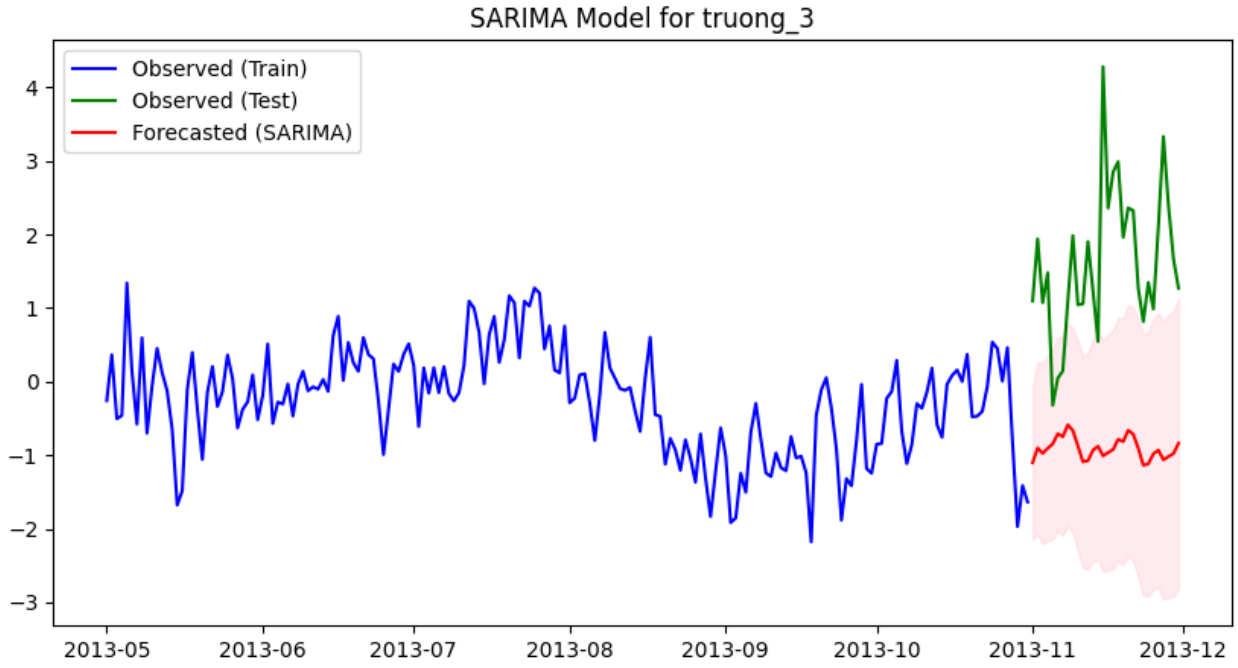
- Đầu tiên, tạo colums chứa các danh sách cột mà ta muốn dự đoán. Sau đó dùng vòng lặp for cho từng cột.

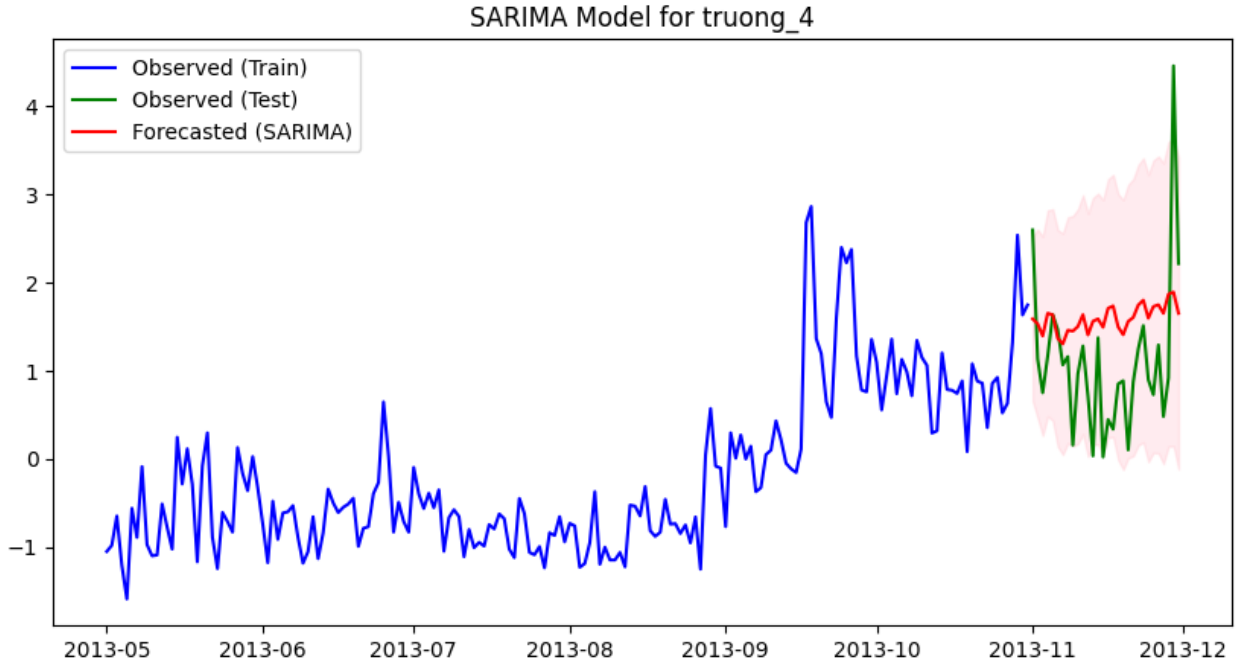
- Khởi tạo cho cột hiện tại với các thông số: order=(1, 1, 1) và seasonal_order=(1, 1, 1, 12)).
- train[column] là dữ liệu huấn luyện cho cột hiện tại. Tiếp theo, huấn luyện mô hình bằng cách gọi phương thức fit().
- Dự báo giá trị cho 30 bước tiếp theo bằng phương thức get_forecast().
- Lấy ra giá trị dự báo trung bình bằng: sarima_predicted_mean = sarima_forecast.predicted_mean
- Lấy ra khoảng tin cậy của dự báo bằng: sarima_conf_int = sarima_forecast.conf_int()

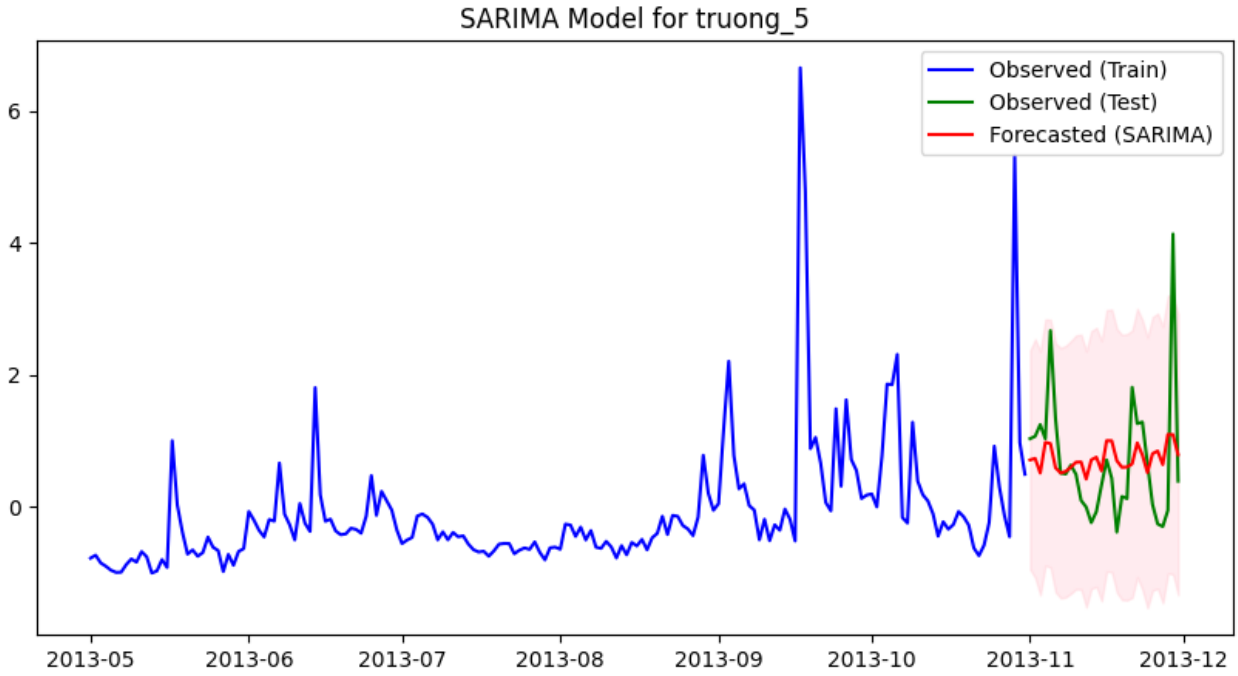
- Vẽ đồ thị


SARIMA Model for truong_1
Observed (Train)
Observed (Test)
Forecasted (SARIMA)
2.5
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
2013-05
2013-06
2013-07
2013-08
2013-09
2013-10
2013-11
2013-12


SARIMA Model for truong_2
Observed (Train)
Observed (Test)
Forecasted (SARIMA)
2
1
0
−1
−2
−3
2013-05
2013-06
2013-07
2013-08
2013-09
2013-10
2013-11
2013-12


SARIMA Model for truong_3
Observed (Train)
Observed (Test)
Forecasted (SARIMA)
4
3
2
1
0
−1
−2
−3
2013-05
2013-06
2013-07
2013-08
2013-09
2013-10
2013-11
2013-12


SARIMA Model for truong_4
Observed (Train)
Observed (Test)
Forecasted (SARIMA)
4
3
2
1
0
−1
2013-05
2013-06
2013-07
2013-08
2013-09
2013-10
2013-11
2013-12

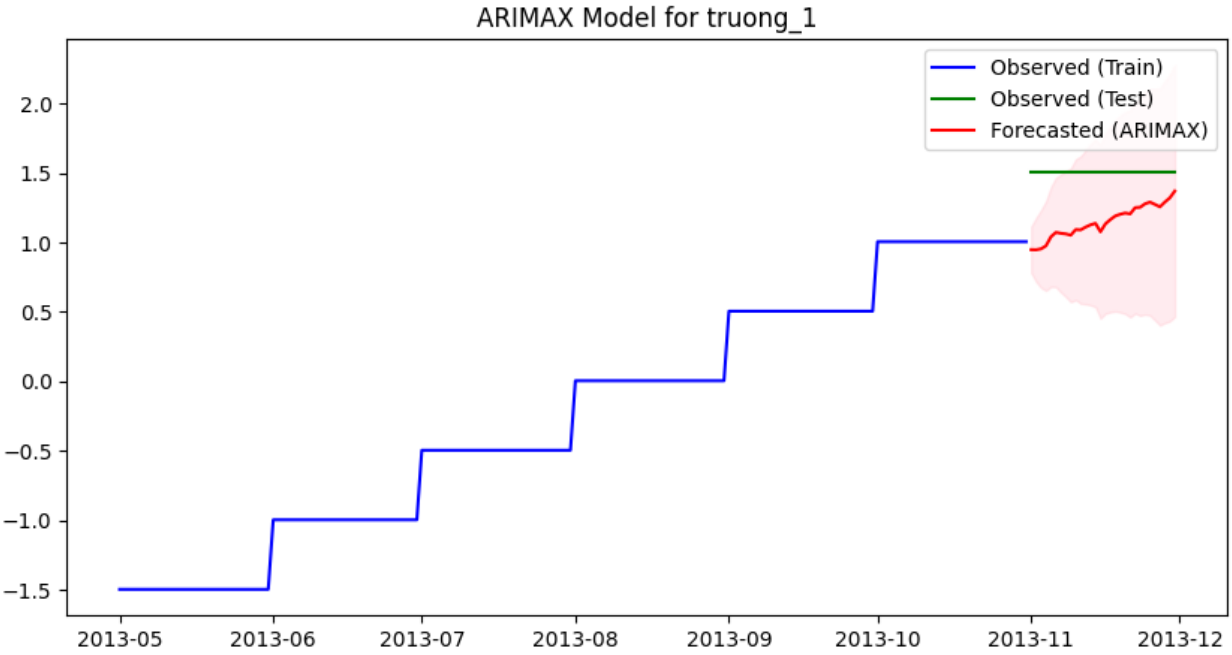


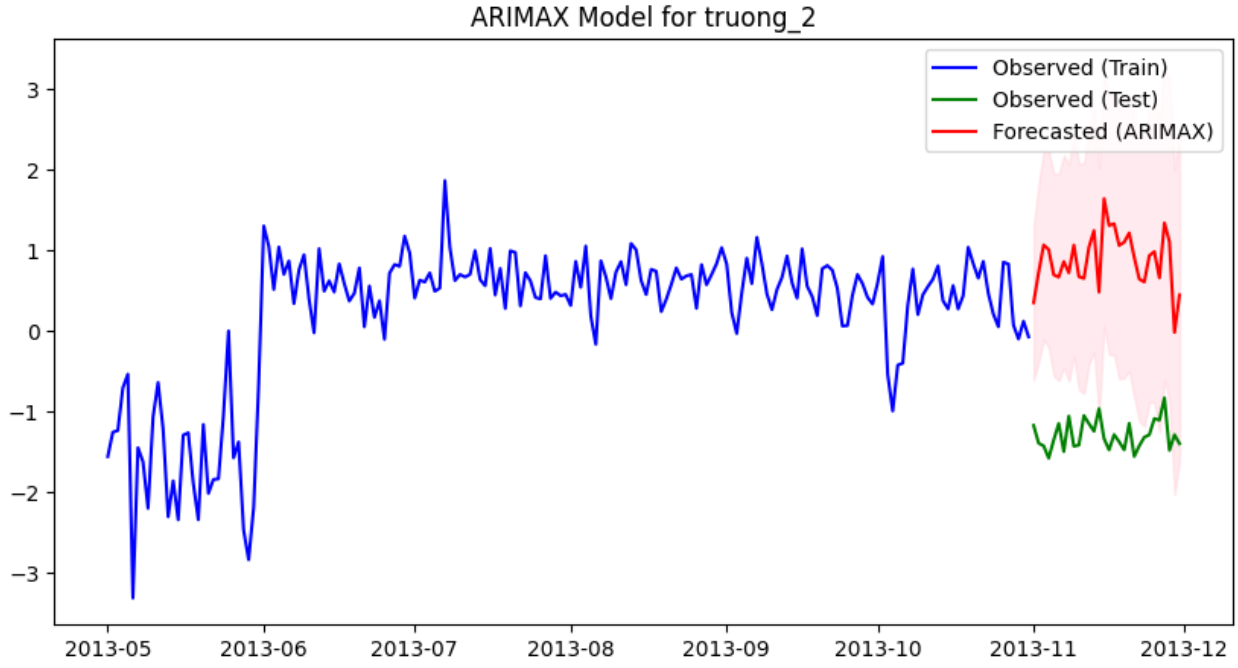

- Mô hình này chính xác trong việc dự báo tương đối tốt tuy nhiên vẫn còn nhiều đoạn dự đoán chưa chính xác

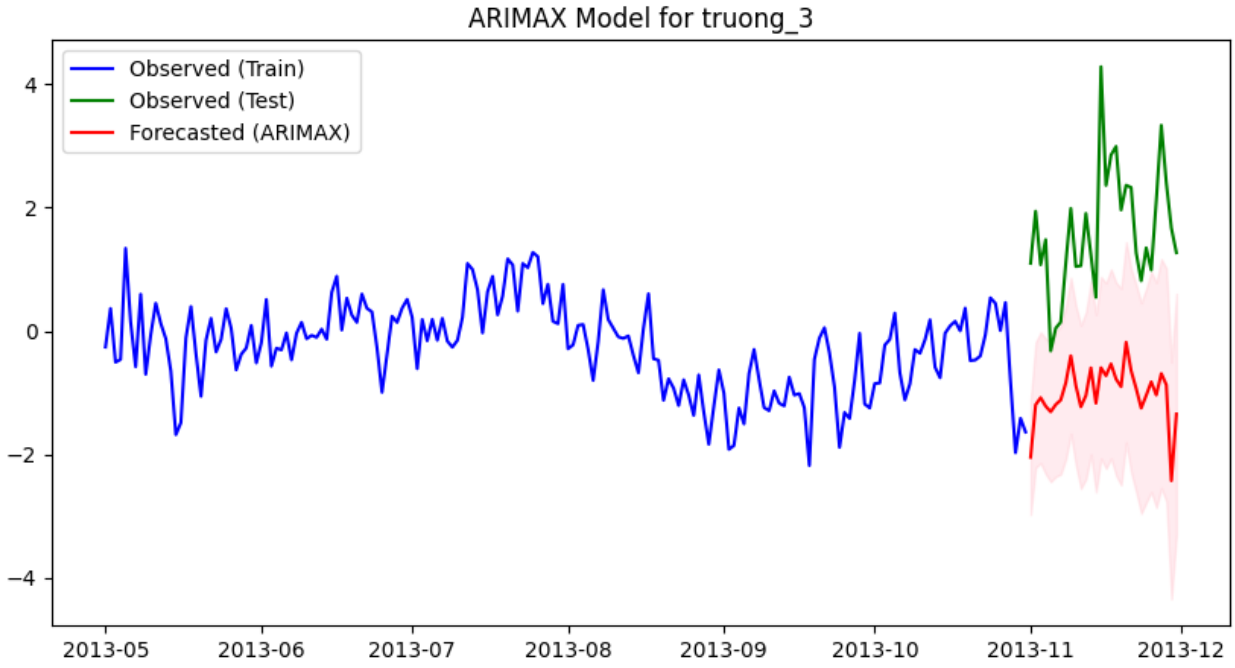
**Dự báo cho mỗi cột với mô hình ARIMAX**

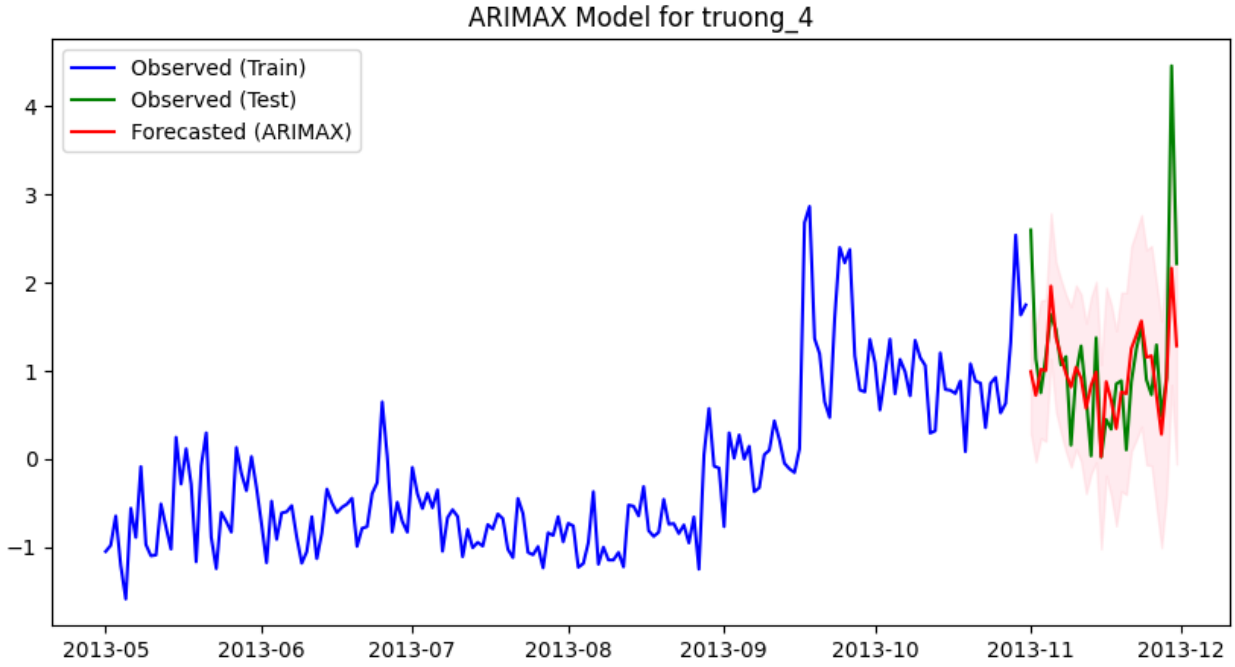
```python
# Dự báo cho mỗi cột với mô hình ARIMAX
columns = ['truong_1', 'truong_2', 'truong_3', 'truong_4', 'truong_5']
arimax_forecasts = {}

for column in columns:
    # Mô hình ARIMAX
    exog_train = train.drop(columns=[column])
    exog_test = test.drop(columns=[column])

    arimax_model = sm.tsa.statespace.SARIMAX(train[column], order=(1, 1, 1), seasonal_order=(1, 1, 1, 12), exog=exog_train)
    arimax_result = arimax_model.fit()

    # Dự báo với mô hình ARIMAX
    arimax_forecast = arimax_result.get_forecast(steps=30, exog=exog_test)
    arimax_predicted_mean = arimax_forecast.predicted_mean
    arimax_conf_int = arimax_forecast.conf_int()

    arimax_forecasts[column] = arimax_predicted_mean

    # Vẽ đồ thị kết quả dự báo ARIMAX
    plt.figure(figsize=(10, 5))

    # Dữ liệu quan sát trong tập huấn luyện
    plt.plot(train.index, train[column], label='Observed (Train)', color='blue')

    # Dữ liệu quan sát trong tập kiểm tra
    plt.plot(test.index, test[column], label='Observed (Test)', color='green')

    # Dự báo của mô hình ARIMAX
    plt.plot(test.index, arimax_predicted_mean, label='Forecasted (ARIMAX)', color='red')

    # Tô màu cho khoảng tin cậy của dự báo
    plt.fill_between(test.index, arimax_conf_int.iloc[:, 0], arimax_conf_int.iloc[:, 1], color='pink', alpha=0.3)

    plt.legend()
    plt.title(f'ARIMAX Model for {column}')
    plt.show()
```

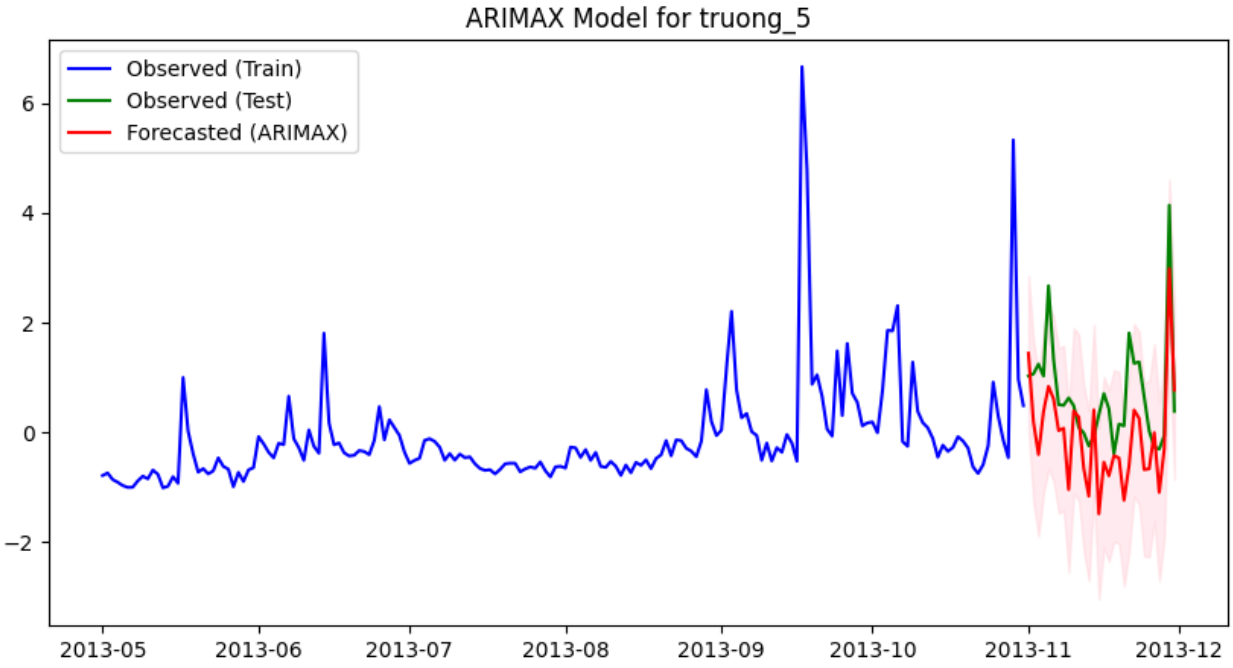
- Đầu tiên, tạo colums chứa các danh sách cột mà ta muốn dự đoán. Sau đó dùng vòng lặp for cho từng cột.
-
- Khởi tạo cho cột hiện tại với các thông số: order=(1, 1, 1) và seasonal_order=(1, 1, 1, 12)).
- train[column] là dữ liệu huấn luyện cho cột hiện tại. Tiếp theo, huấn luyện mô hình bằng cách gọi phương thức fit().
- Dự báo giá trị cho 30 bước tiếp theo bằng phương thức get_forecast().
- Lấy ra giá trị dự báo trung bình bằng: arimax_predicted_mean = arimax_forecast.predicted_mean
- Lấy ra khoảng tin cậy của dự báo bằng: arimax_conf_int = arimax_forecast.conf_int()
- Lưu trữ kết quả dự báo của mô hình: arimax_forecasts[column] = arimax_predicted_mean
- Vẽ biểu đồ cho từng nhãn:


ARIMAX Model for truong_1
Observed (Train)
Observed (Test)
Forecasted (ARIMAX)
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
2013-05
2013-06
2013-07
2013-08
2013-09
2013-10
2013-11
2013-12


ARIMAX Model for truong_2
Observed (Train)
Observed (Test)
Forecasted (ARIMAX)
3
2
1
0
−1
−2
−3
2013-05
2013-06
2013-07
2013-08
2013-09
2013-10
2013-11
2013-12


ARIMAX Model for truong_3
Observed (Train)
Observed (Test)
Forecasted (ARIMAX)
4
2
0
−2
−4
2013-05
2013-06
2013-07
2013-08
2013-09
2013-10
2013-11
2013-12


ARIMAX Model for truong_4
Observed (Train)
Observed (Test)
Forecasted (ARIMAX)
4
3
2
1
0
−1
2013-05
2013-06
2013-07
2013-08
2013-09
2013-10
2013-11
2013-12




- Đường dự đoán của mô hình Arimax thể hiện rằng nó đã khá tốt trong việc bắt chướng ngại về xu hướng của dữ liệu kiểm tra. Tuy nhiên, có một số điểm mà dự đoán của mô hình không hoàn toàn phù hợp với dữ liệu thực tế, đặc biệt là khi xuất hiện các đỉnh và đáy của chuỗi thời gian.
- Mô hình có khả năng dự đoán khá tốt xu hướng tổng quát của chuỗi thời gian cùng với các biến động nhỏ trong ngắn hạn. Tuy nhiên, mô hình vẫn còn hạn chế trong việc dự đoán chính xác các giá trị cụ thể tại các điểm có biến động lớn.