

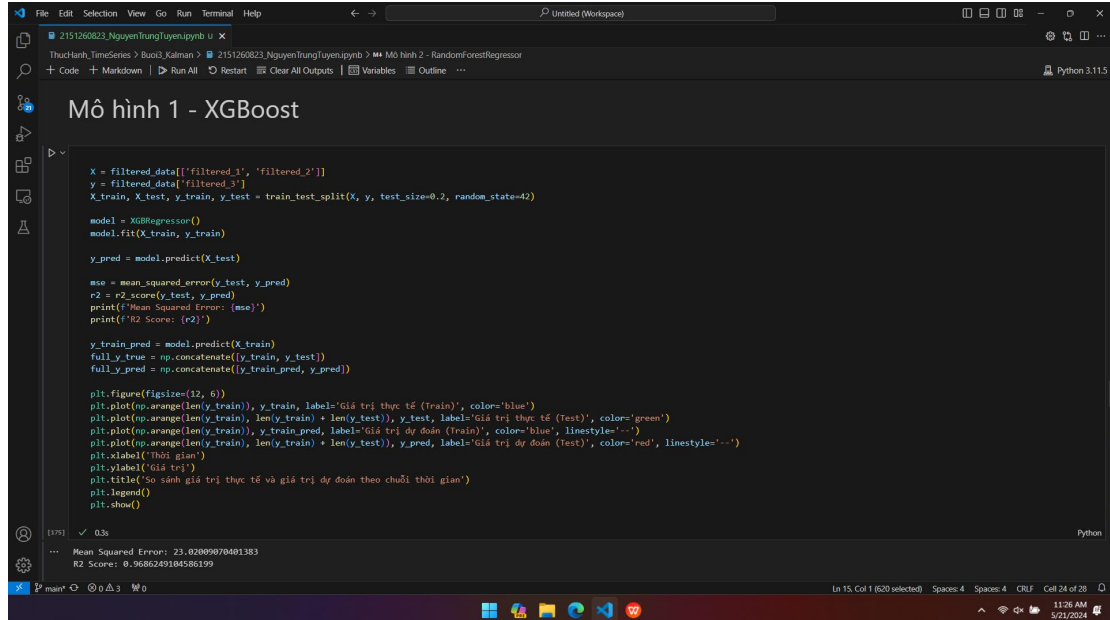
# Nguyễn Trung Tuyền - 2151260823 - Thực hành ngày 21/05/2024

## 1. Kịch bản chia dữ liệu

Các kịch bản chia dữ liệu được sử dụng trong code mô hình XGBoost, RandomForestRegressor, Sarima kết hợp với LinearRegression được chia theo tỉ lệ 80:20. Riêng mô hình Sarima thì sử dụng chu kì 12 ngày.

## 2. Ảnh Training

### 2.1. Mô hình XGBoost



```
File Edit Selection View Go Run Terminal Help
2151260823_NguyenTrungTuyen.ipynb
Thuchanh_TimeSeries > Buoi3_Kalman > 2151260823_NguyenTrungTuyen.ipynb > Mô hình 2 - RandomForestRegressor
+ Code + Markdown | ▶ Run All | ⏮ Restart | 🧹 Clear All Outputs | 📄 Variables | 📖 Outline | ...
Python 3.11.5

Mô hình 1 - XGBoost

X = filtered_data[['filtered_1', 'filtered_2']]
y = filtered_data['filtered_3']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

model = XGBRegressor()
model.fit(X_train, y_train)

y_pred = model.predict(X_test)

mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print(f'Mean Squared Error: {mse}')
print(f'R2 Score: {r2}')

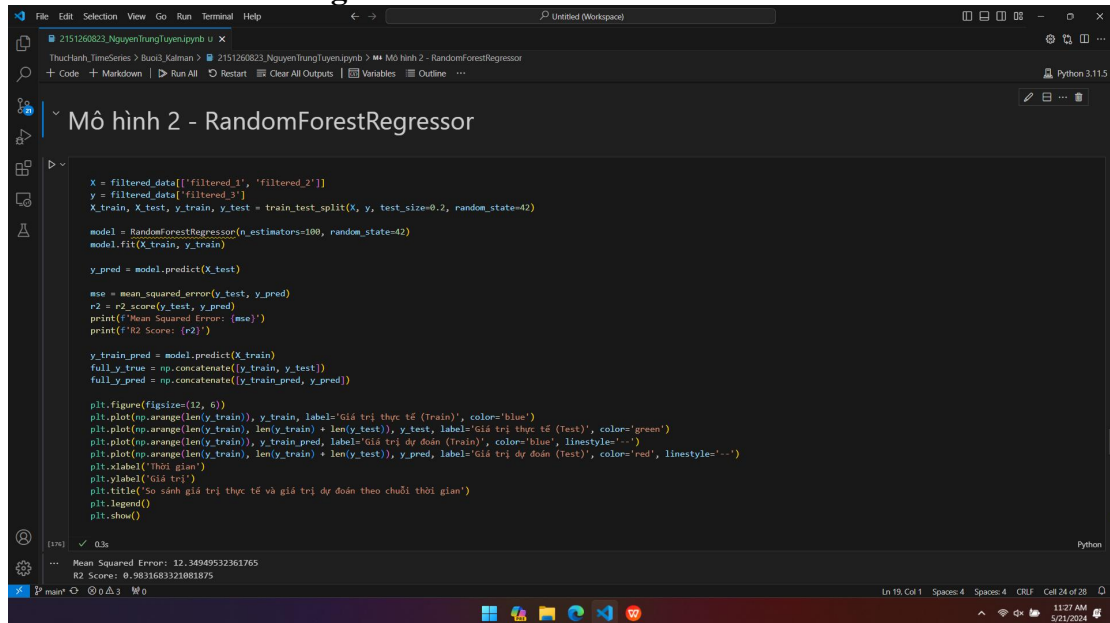
y_train_pred = model.predict(X_train)
full_y_true = np.concatenate([y_train, y_test])
full_y_pred = np.concatenate([y_train_pred, y_pred])

plt.figure(figsize=(12, 6))
plt.plot(np.arange(len(y_train)), y_train, label='Giá trị thực tế (Train)', color='blue')
plt.plot(np.arange(len(y_train), len(y_train) + len(y_test)), y_test, label='Giá trị thực tế (Test)', color='green')
plt.plot(np.arange(len(y_train), len(y_train) + len(y_test)), y_train_pred, label='Giá trị dự đoán (Train)', color='blue', linestyle='--')
plt.plot(np.arange(len(y_train), len(y_train) + len(y_test)), y_pred, label='Giá trị dự đoán (Test)', color='red', linestyle='--')
plt.xlabel('Thời gian')
plt.ylabel('Giá trị')
plt.title('So sánh giá trị thực tế và giá trị dự đoán theo chuỗi thời gian')
plt.legend()
plt.show()

(175) ✓ 0.3s
... Mean Squared Error: 23.02009070401383
R2 Score: 0.9686249104586199
Python

Ln 15, Col 1 (620 selected) Spaces: 4 Spaces: 4 CRLF Cell 24 of 28
11:26 AM 5/21/2024
```

### 2.2. RandomForestRegressor



```
File Edit Selection View Go Run Terminal Help
2151260823_NguyenTrungTuyen.ipynb
Thuchanh_TimeSeries > Buoi3_Kalman > 2151260823_NguyenTrungTuyen.ipynb > Mô hình 2 - RandomForestRegressor
+ Code + Markdown | ▶ Run All | ⏮ Restart | 🧹 Clear All Outputs | 📄 Variables | 📖 Outline | ...
Python 3.11.5

Mô hình 2 - RandomForestRegressor

X = filtered_data[['filtered_1', 'filtered_2']]
y = filtered_data['filtered_3']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

model = RandomForestRegressor(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

y_pred = model.predict(X_test)

mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print(f'Mean Squared Error: {mse}')
print(f'R2 Score: {r2}')

y_train_pred = model.predict(X_train)
full_y_true = np.concatenate([y_train, y_test])
full_y_pred = np.concatenate([y_train_pred, y_pred])

plt.figure(figsize=(12, 6))
plt.plot(np.arange(len(y_train)), y_train, label='Giá trị thực tế (Train)', color='blue')
plt.plot(np.arange(len(y_train), len(y_train) + len(y_test)), y_test, label='Giá trị thực tế (Test)', color='green')
plt.plot(np.arange(len(y_train), len(y_train) + len(y_test)), y_train_pred, label='Giá trị dự đoán (Train)', color='blue', linestyle='--')
plt.plot(np.arange(len(y_train), len(y_train) + len(y_test)), y_pred, label='Giá trị dự đoán (Test)', color='red', linestyle='--')
plt.xlabel('Thời gian')
plt.ylabel('Giá trị')
plt.title('So sánh giá trị thực tế và giá trị dự đoán theo chuỗi thời gian')
plt.legend()
plt.show()

(176) ✓ 0.3s
... Mean Squared Error: 12.34949532361765
R2 Score: 0.9831683321081875
Python

Ln 19, Col 1 Spaces: 4 Spaces: 4 CRLF Cell 24 of 28
11:27 AM 5/21/2024
```

### 2.3. Sarima kết hợp với LinearRegression

```
File Edit Selection View Go Run Terminal Help
2151260823_NguyenHungTuyen.ipynb
Thư viện TimeSeries > Built-in > 2151260823_NguyenHungTuyen.ipynb > Mô hình 2 - RandomForestRegressor
+ Code + Markdown + Run All + Restart + Clear All Outputs + Variables + Outline ...
Python 3.11.5

X = filtered_data.diff[['filtered_1', 'filtered_2']]
y = filtered_data.diff[['filtered_3']]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

lr_model = LinearRegression()
lr_model.fit(X_train, y_train)

y_train_pred_lr = lr_model.predict(X_train)
y_test_pred_lr = lr_model.predict(X_test)

train_size = int(len(y) * 0.8)
train, test = y[:train_size], y[train_size:]

sarima_model = SARIMAX(train, order=(1, 0, 3), seasonal_order=(1, 0, 3, 12))
sarima_model_fit = sarima_model.fit(dispatch=False)

y_train_pred_sarima = sarima_model_fit.predict(start=0, end=len(train)-1)
y_test_pred_sarima = sarima_model_fit.predict(start=len(train), end=len(train)+len(test)-1)

y_train_pred_combined = (y_train_pred_lr + y_train_pred_sarima[:len(y_train_pred_lr)]) / 2
y_test_pred_combined = (y_test_pred_lr + y_test_pred_sarima) / 2

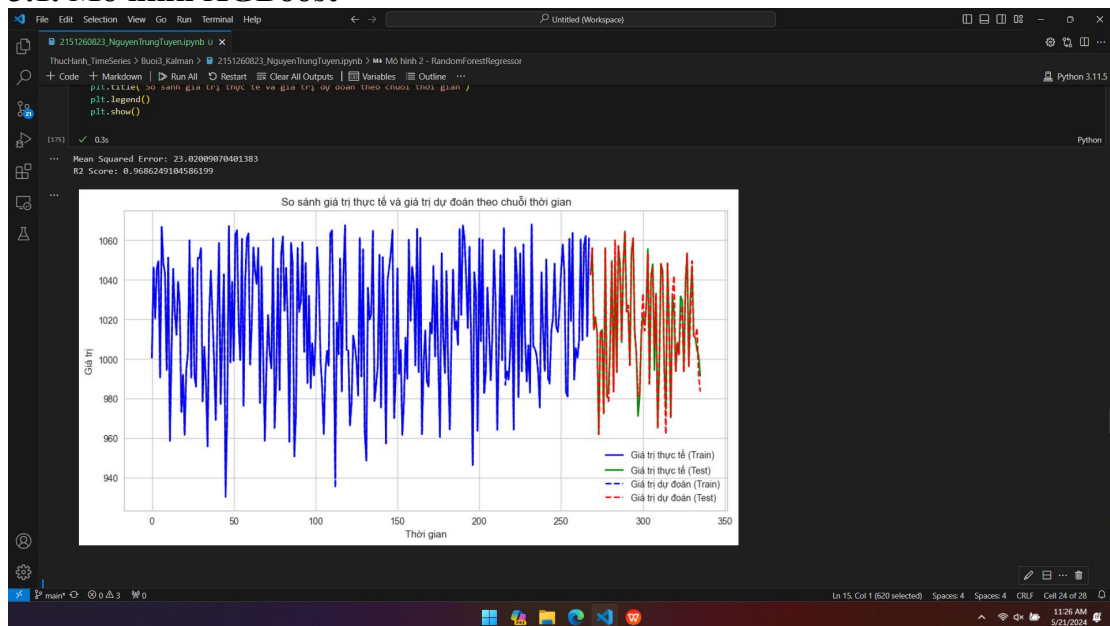
mse_combined = mean_squared_error(y_test, y_test_pred_combined)
r2_combined = r2_score(y_test, y_test_pred_combined)
print('Mean Squared Error: (mse_combined)')
print('R2 Score: (r2_combined)')

plt.figure(figsize=(12, 6))
plt.plot(np.arange(len(train)), train, label='Giá trị thực tế (Train)', color='blue')
plt.plot(np.arange(len(train), len(train) + len(test)), test, label='Giá trị thực tế (Test)', color='green')
plt.plot(np.arange(len(train)), y_train_pred_combined, label='Giá trị dự đoán (Train)', color='blue', linestyle='--')
plt.plot(np.arange(len(train), len(train) + len(test)), y_test_pred_combined, label='Giá trị dự đoán (Test)', color='red', linestyle='--')
plt.xlabel('Thời gian')
plt.ylabel('Giá trị')
plt.title('So sánh giá trị thực tế và giá trị dự đoán theo chuỗi thời gian (Mô hình kết hợp)')
plt.legend()
plt.show()

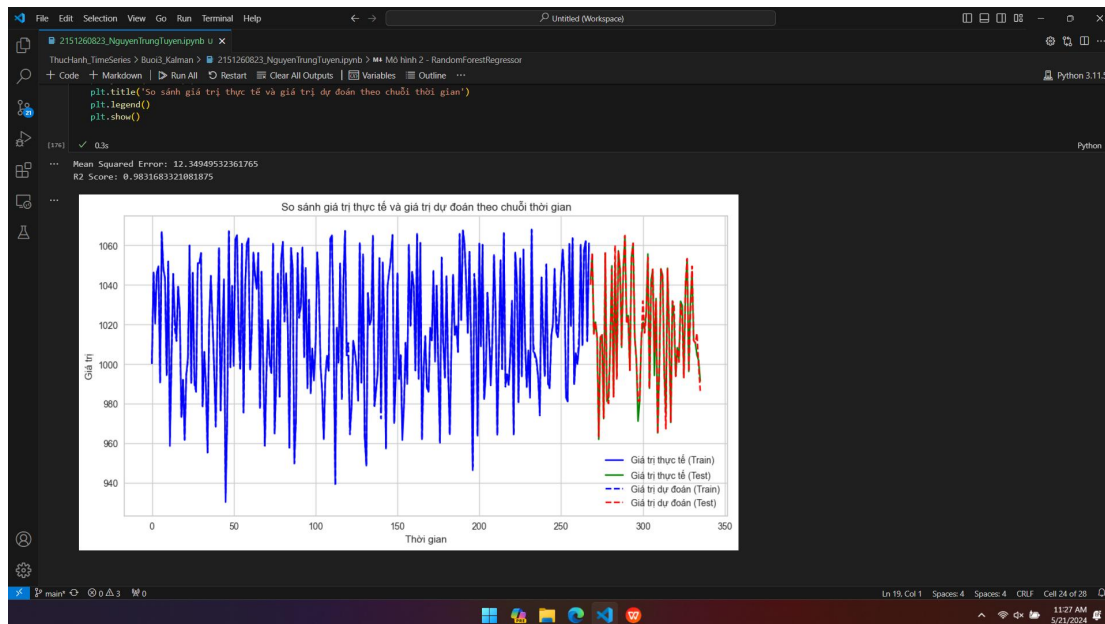
main
```

### 3. Ảnh kết quả chạy

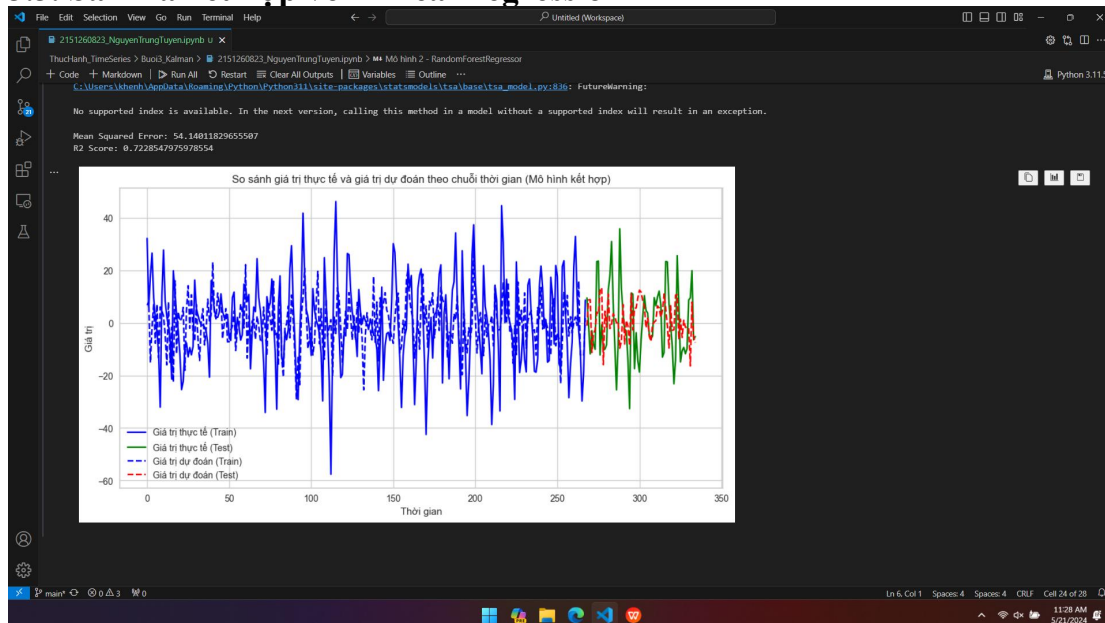
#### 3.1. Mô hình XGBoost



#### 3.2. RandomForestRegressor



### 3.3. Sarima kết hợp với LinearRegression



### 4. Nhận xét về kết quả chạy của 3 mô hình

- Với cùng tập dữ liệu, mô hình RandomForestRegressor cho ra kết quả R2 là tốt nhất (0.983).
- Mô hình kết hợp Sarima và LinearRegression có kết quả R2 thấp nhất = 0.7728.

### 5. Link code up trên github

[https://github.com/tuyennhe/Thuchanh\\_TimeSeries/tree/main/Buoi3\\_Kalman](https://github.com/tuyennhe/Thuchanh_TimeSeries/tree/main/Buoi3_Kalman)