

Thực hành 2 - 14/05/2024

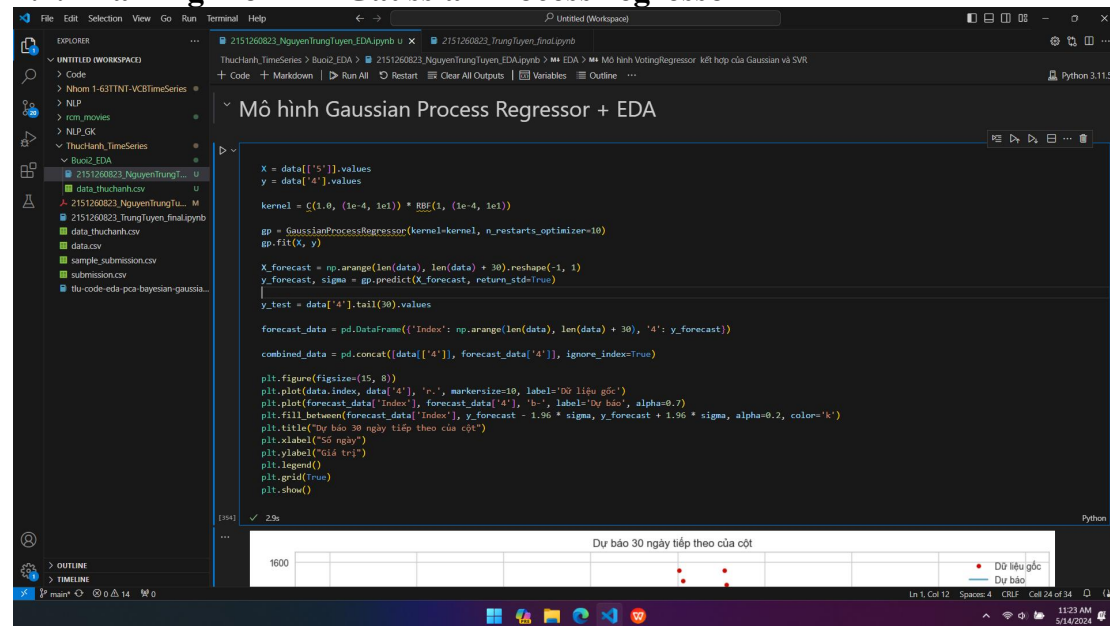
Nguyễn Trung Tuyển - 2151260823

1. Kịch bản chia data

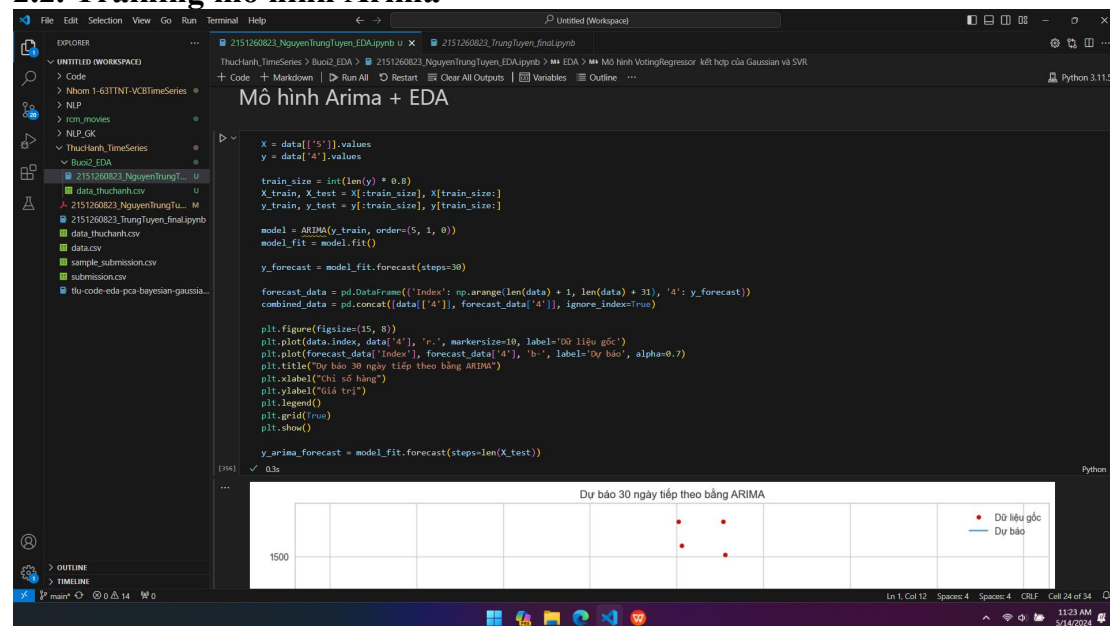
- Trong bài thực hành 02, em sử dụng 4 mô hình kết hợp với EDA: GaussianProcessRegressor, Arima, XGBoost, VotingRegressor(Gau nhưng bổ sung thêm SVR)
- Với các mô hình chia kịch bản: Sử dụng cột 5 làm đặc trưng, cột 4 làm nhãn. Dự báo ra 30 ngày tiếp theo.

2. Ảnh training

2.1. Training mô hình GaussianProcessRegressor



2.2. Training mô hình Arima



2.3. Training mô hình XGBoost

```
Thuchanh,TimeSeries> Buoi2_EDA> 2151260823_NguyenTrungTuyen_EDA.ipynb> MA EDA> Mô hình VotingRegressor kết hợp của Gaussian và SVR

Mô hình XGBoost + EDA

X = data[['5']].values
y = data[['4']].values

train_size = int((len(X) * 0.8))
X_train, X_test, y_train, y_test = X[:train_size], X[train_size:], y[:train_size], y[train_size:]

dtrain = xgb.DMatrix(X_train, label=y_train)
dtest = xgb.DMatrix(X_test, label=y_test)

params = {
    'objective': 'reg:squarederror',
    'colsample_bytree': 0.3,
    'learning_rate': 0.1,
    'max_depth': 5,
    'alpha': 10,
    'n_estimators': 100
}

model = xgb.train(params, dtrain)
y_forecast = model.predict(dtest)

forecast_data = pd.DataFrame({'Index': np.arange(train_size, len(X)), '4': y_forecast})
combined_data = pd.concat([data[['4']], forecast_data[['4']], ignore_index=True)

plt.figure(figsize=(15, 8))
plt.plot(data.index[:train_size], data['4'][:train_size], 'r.', markersize=10, label='Dữ liệu train')
plt.plot(data.index[train_size:], data['4'][:train_size], 'g.', markersize=10, label='Dữ liệu test')
plt.plot(forecast_data['Index'], forecast_data['4'], 'b-', label='Dự báo', alpha=0.7)
plt.title('Dự báo bằng XGBoost')
plt.xlabel('Chỉ số hàng')
plt.ylabel('Giá trị')
plt.legend()
...

```

2.4. Training mô hình Voting Regressor

```
Thuchanh,TimeSeries> Buoi2_EDA> 2151260823_NguyenTrungTuyen_EDA.ipynb> MA EDA> Mô hình VotingRegressor kết hợp của Gaussian và SVR

X = data[['5']].values
y = data[['4']].values

kernel = C(1.0, (1e-4, 1e1)) * RBF(1, (1e-4, 1e1))
gp = GaussianProcessRegressor(kernel=kernel, n_restarts_optimizer=10)

svr = SVR(kernel='rbf', C=100, gamma='auto')

ensemble_model = VotingRegressor([('gp', gp), ('svr', svr)])

ensemble_model.fit(X, y)

X_forecast = np.arange(len(data), len(data) + 30).reshape(-1, 1)
y_forecast = ensemble_model.predict(X_forecast)

forecast_data = pd.DataFrame({'Index': np.arange(len(data), len(data) + 30), '4': y_forecast})
combined_data = pd.concat([data[['4']], forecast_data[['4']], ignore_index=True)

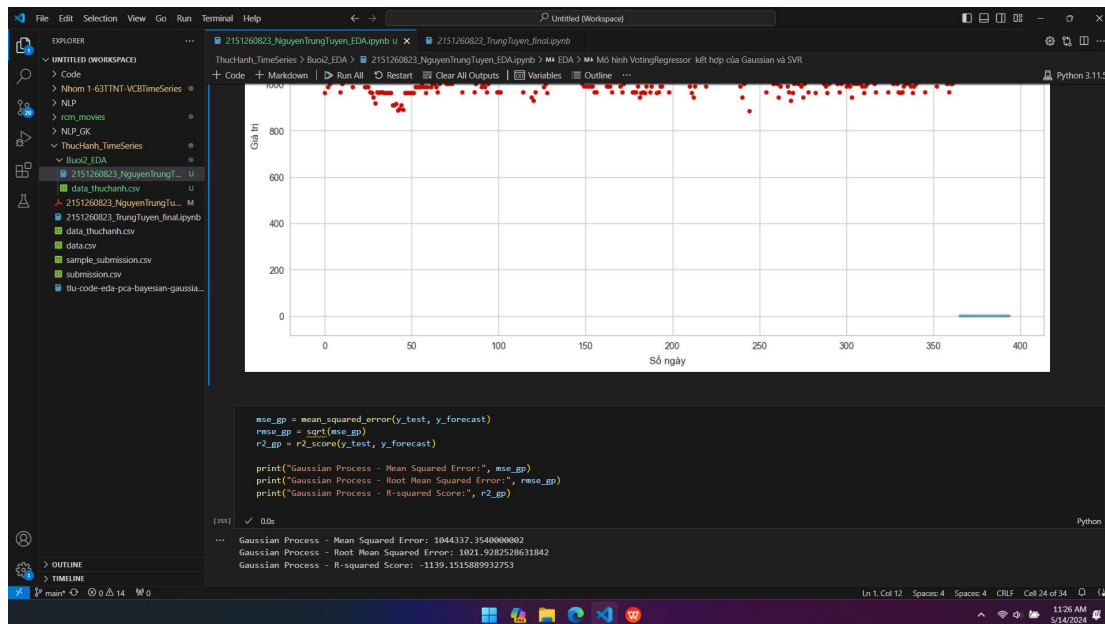
plt.figure(figsize=(15, 8))
plt.plot(data.index, data['4'], 'r.', markersize=10, label='Dữ liệu gốc')
plt.plot(forecast_data['Index'], forecast_data['4'], 'b-', label='Dự báo', alpha=0.7)
plt.title('Dự báo 30 ngày tiếp theo bằng Ensemble Learning')
plt.xlabel('Chỉ số hàng')
plt.ylabel('Giá trị')
plt.legend()
plt.grid(True)
plt.show()

y_train_forecast = ensemble_model.predict(X_train)
y_test_forecast = ensemble_model.predict(X_test)
...

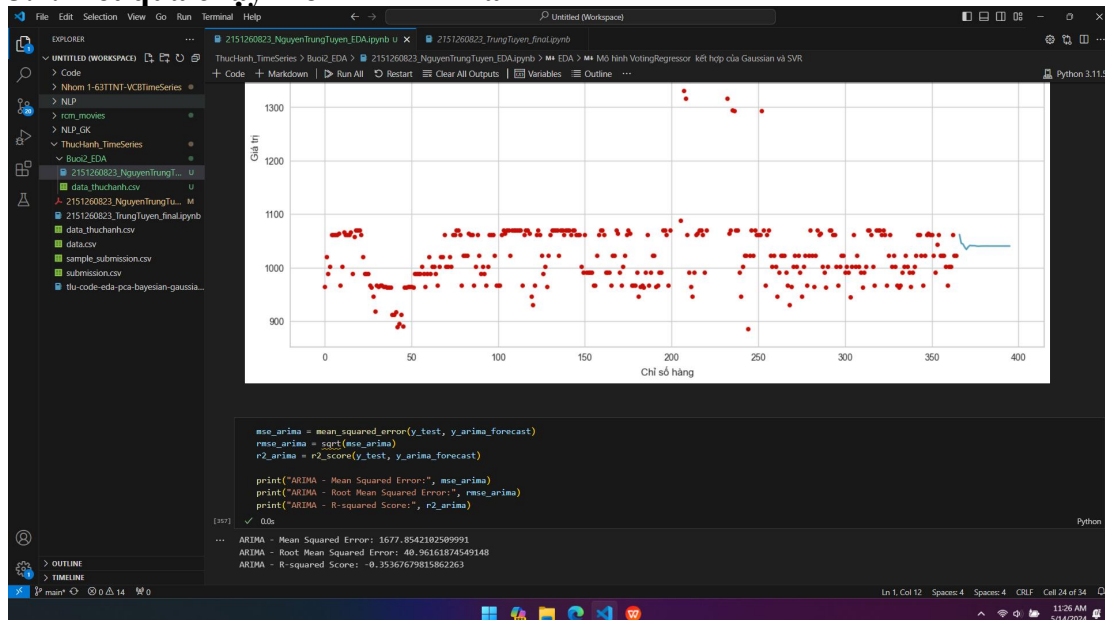
```

3. Ảnh kết quả chạy

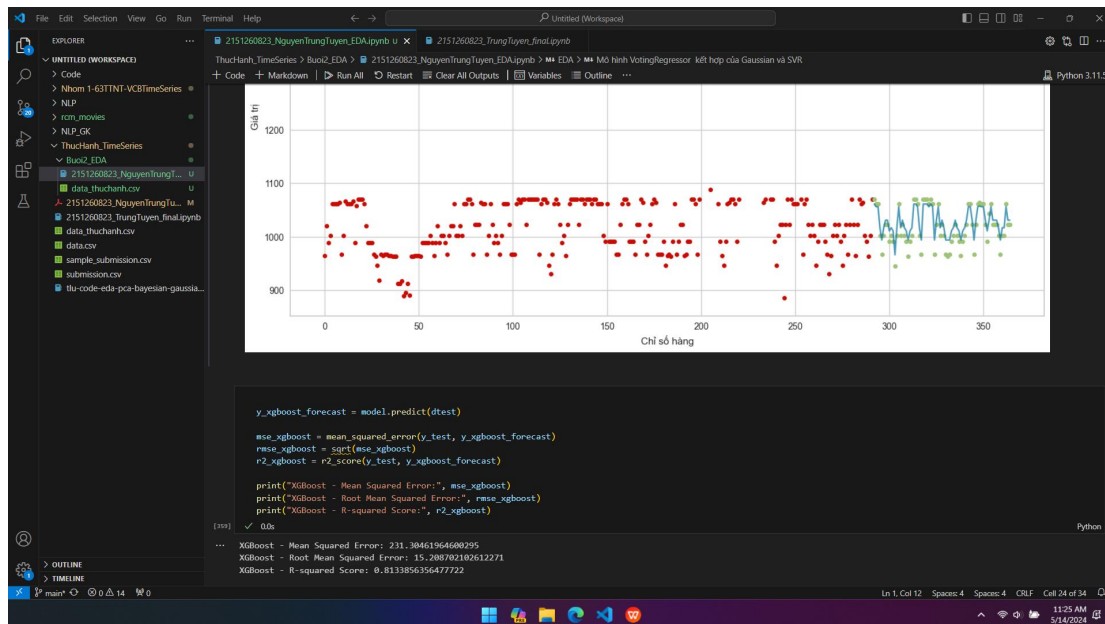
3.1. Kết quả chạy mô hình GaussianProcessRegressor



3.2. Kết quả chạy mô hình Arima



3.3. Kết quả chạy mô hình XGBoost



3.4. Kết quả chạy mô hình Voting Regressor



3.5. Nhận xét

- Với bộ dữ liệu và cách chia dữ liệu ở trên, mô hình XGBoost cho ra kết quả tốt nhất với $R^2 = 0.813$.

4. Link code trên github

https://github.com/tuyennhe/ThucHanh_TimeSeries/tree/main/Buoi2_EDA

File code thực hành 02 có tên: **2151260823_NguyenTrungTuyen_EDA.ipynb**