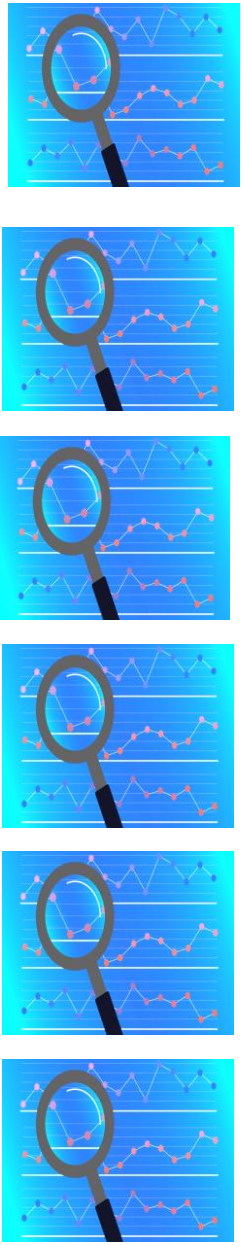


Chương 4: : Mô hình dự báo bằng phương pháp học máy

Thời gian: 5LT+5TH

GVHD: TS.Dương Thị Kim Chi

Nội Dung

1. **Chương 1**: Tổng quan (5 tiết)
2. **Chương 2:** Mô hình dự báo không có giám sát (5 tiết)
3. **Chương 3:** Mô hình dự báo đa biến (5 tiết)
4. **Chương 4: Mô hình dự báo bằng phương pháp học máy** (5 tiết)
5. **Chương 5:** mô hình dự báo bằng phương pháp học sâu (10 tiết
6. **Chương 6: C**ác bài toán ứng dụng (5 tiết)

Nội dung

4.1. • Giới thiệu

4.2. • Quy trình

4.3. • Các giải thích cơ bản của mô hình học máy

4.4. • Ứng dụng

1. Giới thiệu

4.1. Tổng quan so sánh

Phương pháp tiếp cận chuỗi thời gian

#	Category	Model	Handles Trend	Handles Seasonality	Computational Complexity	Interpretability	Hyperparameter Tuning	Python Package
1	Univariate	Simple Exponential Smoothing (SES)	No	No	Low	Easy	Low	`statsmodels`
2	Univariate	Holt's Linear Trend	Yes	No	Low	Easy	Low	`statsmodels`
3	Univariate	Holt-Winters Seasonal	Yes	Yes	Low	Moderate	Low	`statsmodels`
4	Univariate	ARIMA	Yes	No	Medium	Moderate	High	`statsmodels`
5	Univariate	SARIMA	Yes	Yes	Medium-High	Moderate	High	`statsmodels`
6	Multivariate	Vector Autoregressive (VAR)	Yes	No	Medium	Moderate	Medium	`statsmodels`
7	Machine Learning	Random Forest	Yes	No	High	Moderate-Difficult	Medium	`sklearn`
8	Machine Learning	XGBoost	Yes	No	High	Moderate-Difficult	High	`xgboost`
9	Deep Learning	LSTM	Yes	Yes	High	Difficult	High	`tensorflow`, `keras`
10	Deep Learning	GRU	Yes	Yes	High	Difficult	High	`tensorflow`, `keras`

4.1. Tổng quan so sánh

1. Phương pháp thống kê đơn biến 1.1. Làm mịn hàm mũ đơn giản 1.2. Xu hướng tuyến tính của Holt 1.3. Theo mùa Holt-Winters 1.4. Trung bình động tích hợp tự hồi quy (ARIMA) 1.5. Trung bình động tích hợp tự hồi quy theo mùa (SARIMA) **2. Phương pháp thống kê đa biến** 2.1. Tự hồi quy vectơ (VAR) **3. Các phương pháp học máy** 3.1. Rừng ngẫu nhiên 3.2. Tăng cường độ dốc cực đại (XGBoost) **4. Phương pháp học sâu** 4.1. Bộ nhớ dài hạn ngắn hạn (LSTM) 4.2 Đơn vị tuần hoàn có cổng (GRU)

4.1.Tổng quan

Các phương pháp tiếp cận học máy

Khi nói đến học máy, có nhiều phương pháp có thể giúp chúng ta dự báo theo chuỗi thời gian, bao gồm nhiều **phương pháp hồi quy** và **cây quyết định**, chẳng hạn như **rừng ngẫu nhiên** và **XGBoost**

So sánh với Phương pháp **Multivariate** Statistical Mô hình thống kê đa biến – Tự hồi quy vectơ (VAR) Models – Vector Autoregresssion (VAR)

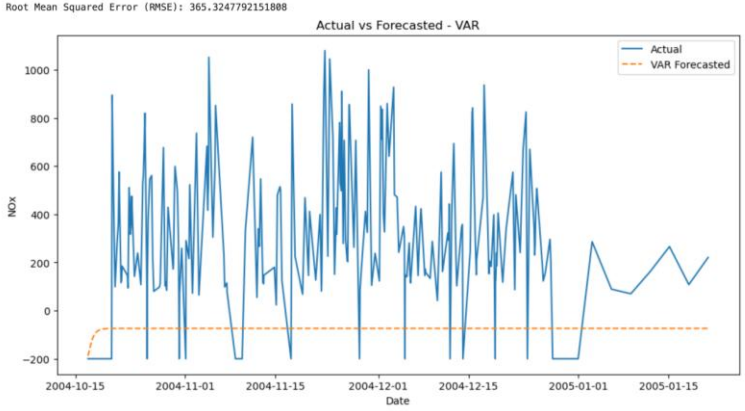
4.2. Các thuật toán học máy

Rừng ngẫu nhiên là một thuật toán học máy mạnh được sử dụng cho cả nhiệm vụ hồi quy và phân loại.

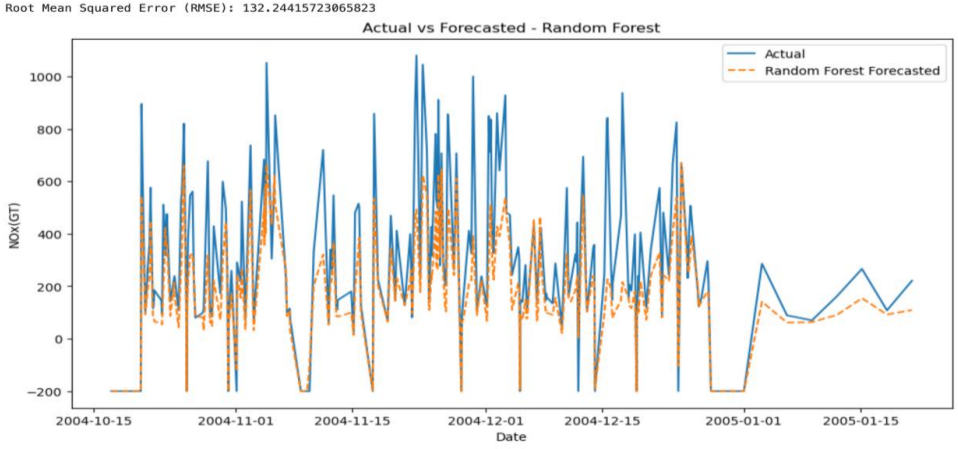
Rừng ngẫu nhiên dựa trên ý tưởng về học tập tổng hợp. Việc học tập tổng hợp trên nhiều cây quyết định tạo ra hiệu suất tốt hơn so với một mô hình duy nhất.

Khi triển khai VAR và xem xét các dự báo trên cùng một bộ dữ liệubộ dữ liệu chất lượng không khí nguồn mở từ kho dữ liệu học máy của UC Irvine

Kết quả:



Kết quả ban đầu và dự báo – Mô hình VAR



Kết quả ban đầu và dự đoán – Random Forest

4.3.Quy trình thực hiện

Quy trình

B1: Create lagged features

B2: separate X and y

B3: **Train**

B4: Forecast

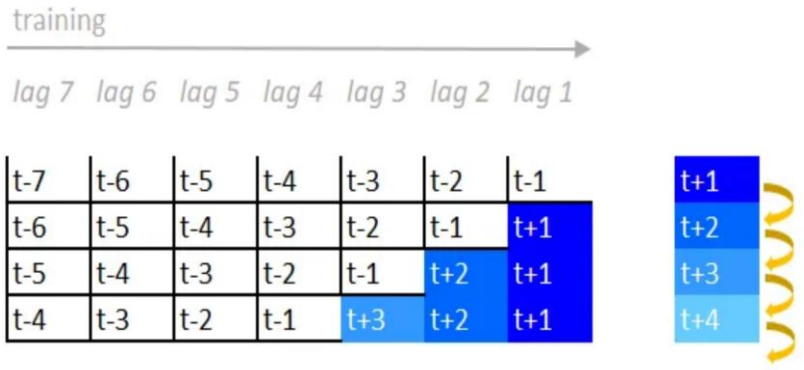
B5: Evaluate

B6: Visualize

Demo

4.3.Quy trình thực hiện_B1: Create lagged features

Lagged features: còn được gọi là Future Covariates được sử dụng để dự đoán bước thời gian tiếp theo. Future Covariates là các biến bổ sung, ngoài biến mục tiêu, được sử dụng làm các tính năng trong mô hình dự báo chuỗi thời gian.




```python
# import libraries
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error
import matplotlib.pyplot as plt
import numpy as np

# B1:create lagged features
def create_lagged_features(data, n_lags):
    lagged_data = data.copy()
    for i in range(1, n_lags + 1):
        lagged_data[f'NOx_lag_{i}'] = lagged_data['NOx(GT)'].shift(i)
    lagged_data = lagged_data.dropna()
    return lagged_data
```

4.3.Quy trình thực hiện_B2, B3: tạo mô hình học máy.

Các bước thực hiện B2 tạo tập huấn luyện; B3:Chọn **các thuật toán học máy** cho mô hình dự đoán

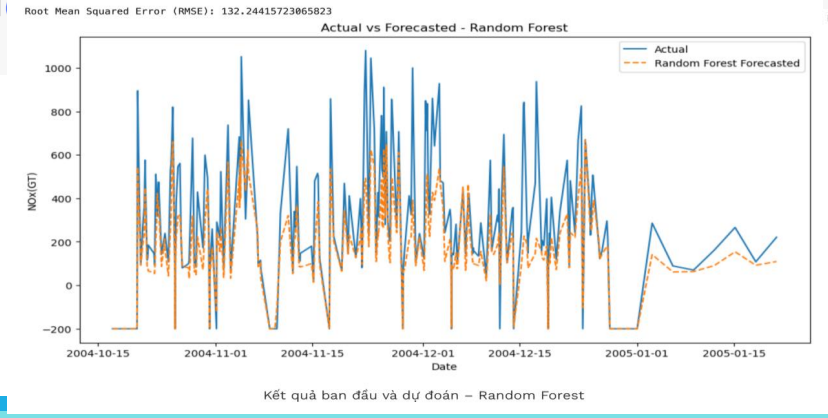
```
# apply lagged feature generation to both train and test data
train_data_lagged = create_lagged_features(train_data, n_lags)
test_data_lagged = create_lagged_features(test_data, n_lags)

#B2: separate X and y
X_train = train_data_lagged.drop(columns=['NOx(GT)'])
y_train = train_data_lagged['NOx(GT)']

X_test = test_data_lagged.drop(columns=['NOx(GT)'])
y_test = test_data_lagged['NOx(GT)']

#B3: Train
rf_model = RandomForestRegressor(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)
```

4.3.Quy trình thực hiện_B4; B5: Đánh giá kết quả và trực quan hóa kết quả

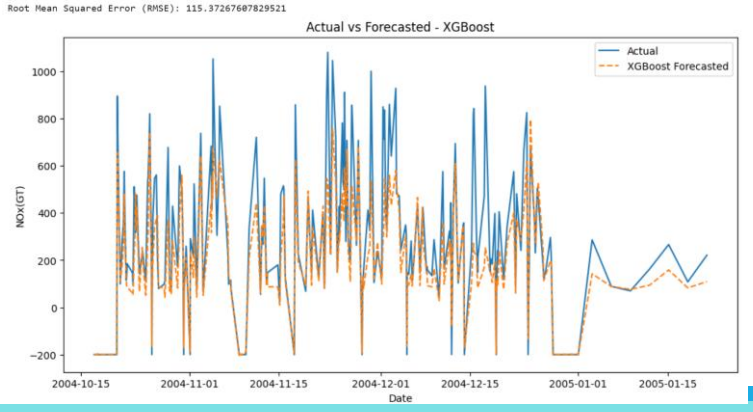
B4: Tạo dự báo; **B5**: đánh giá mô hình học máy : RMSE...R2..

```
#B4: Forecast
y_pred = rf_model.predict(X_test)
```

```
# B5:Evaluate
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
print(f'Root Mean Squared Error (RMSE): {rmse}')
# B6:Visualize
plt.figure(figsize=(12, 6))
plt.plot(y_test.index, y_test, label='Actual')
plt.plot(y_test.index, y_pred, label='Random Forest Forecasted', linestyle='--')
plt.title('Actual vs Forecasted - Random Forest')
plt.xlabel('Date')
plt.ylabel('NOx(GT)')
plt.legend
plt.show()
```



Kết quả ban đầu và dự đoán – Random Forest



Bài tập

Tìm hiểu Các mô hình học máy có thể sử dụng cho dự báo chuỗi thời gian: Tree-based Time Series; Probabilistic forecasting; LightGBM, XGBoost, CatBoost

4.4.Ứng dụng