EX:No.10 221501060

12/04/25

**Develop vector auto regression model for multivariate time series data forecasting**

**Aim:**

ToDevelop vector auto regression model for multivariate time series data forecasting

**Algorithm:**

1. **Load the Data**:
   * Read the CSV file containing the weather data.
   * Parse the date column as a datetime index.
2. **Clean the Data**:
   * Handle missing values by performing forward and backward filling.
   * Drop any remaining NaN values.
3. **Normalize the Data**:
   * Apply **Min-Max Scaling** to normalize each column's values between 0 and 1.
4. **Add Time-Based Features**:
   * Extract additional features from the datetime index: day, month and year
5. **Visualize the Data**:
   * Plot the time series for a specific column (e.g., temperature T) over time.
6. **Execute the Program**:
   * Sequentially call the functions to load, clean, normalize, add features, and visualize the data.

**Code:**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from statsmodels.tsa.api import VAR

from sklearn.metrics import mean\_squared\_error

np.random.seed(42)

date\_range = pd.date\_range(start='2018-01-01', periods=100, freq='M')

energy = np.cumsum(np.random.normal(300, 10, size=100))

temperature = np.sin(np.linspace(0, 20, 100)) \* 10 + 25

humidity = np.random.normal(50, 5, size=100)

df = pd.DataFrame({

'Date': date\_range,

'Energy': energy,

'Temperature': temperature,

'Humidity': humidity

})

df.set\_index('Date', inplace=True)

df.plot(subplots=True, figsize=(10, 6), title='Simulated Energy Data')

plt.tight\_layout()

plt.show()

n\_obs = 10 # forecast horizon

train = df[:-n\_obs]

test = df[-n\_obs:]

model = VAR(train)

lag\_order = model.select\_order(maxlags=15)

print("Selected Lags:\n", lag\_order.summary()) # optional

selected\_lag = lag\_order.selected\_orders['aic'] # use AIC-selected lag

fitted\_model = model.fit(selected\_lag)

forecast\_input = train.values[-selected\_lag:]

forecast = fitted\_model.forecast(y=forecast\_input, steps=n\_obs)

forecast\_df = pd.DataFrame(forecast, index=test.index, columns=test.columns)

for col in df.columns:

plt.figure(figsize=(8, 3))

plt.plot(test.index, test[col], label='Actual')

plt.plot(test.index, forecast\_df[col], label='Forecast')

plt.title(f'Forecast vs Actual - {col}')

plt.legend()

plt.tight\_layout()

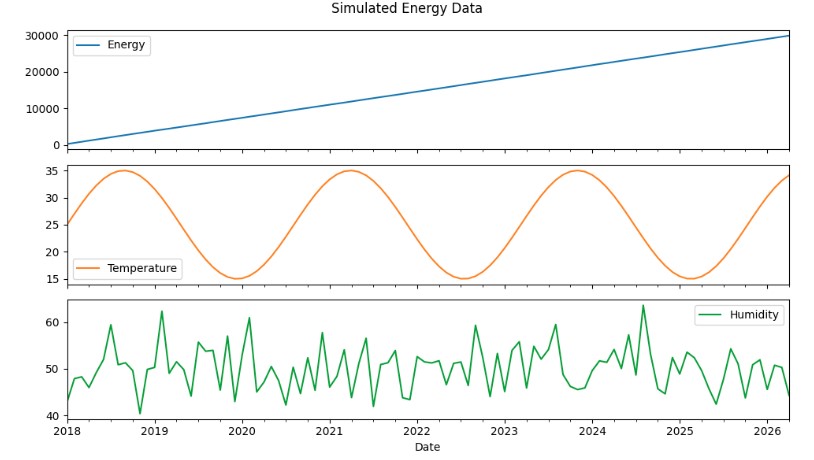
plt.show()

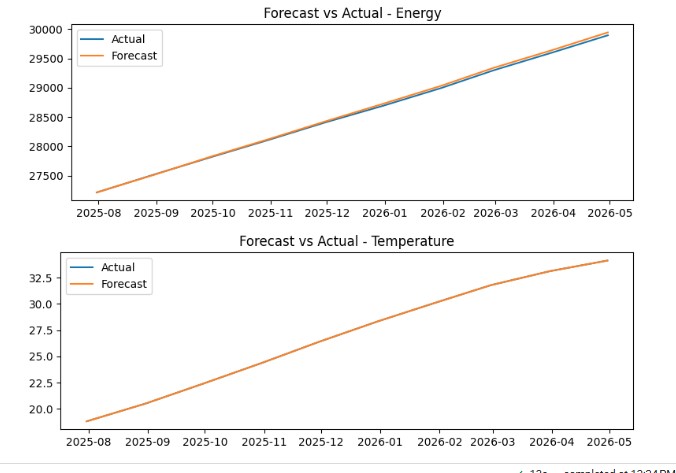
for col in df.columns:

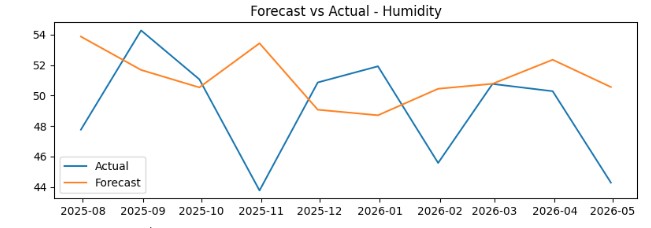
mse = mean\_squared\_error(test[col], forecast\_df[col])

print(f'{col} - Mean Squared Error: {mse:.2f}')

**Output:**

****

****



**Result:**

Thus, the program using the time series data implementation has been done successfully.