EX:No.10

DATE:12/04/25

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

AIM:

To Develop vector auto regression model for multivariate time series data forecasting.

ALGORITHM:

1. **Load and preprocess data**: Import the dataset, convert 'Year' and 'Month' to datetime, set it as index, and drop unnecessary columns.
2. **Handle missing values**: Fill any missing values using forward fill to maintain continuity.
3. **Split dataset**: Divide the data into training (80%) and testing (20%) sets.
4. **Fit VAR model**: Initialize the VAR model on training data, select optimal lag order (e.g., using AIC), and fit the model.
5. **Forecast future values**: Use the fitted model to forecast the same number of steps as in the test set.
6. **Evaluate and visualize**: Plot actual vs forecasted values for each variable to assess model performance.

Code:

```
import pandas as pd
import numpy as np
from statsmodels.tsa.api import VAR
import matplotlib.pyplot as plt

# Step 1: Load the dataset
data = pd.read_csv("/content/us_air_pollution_2012_2021.csv")

# Step 2: Convert Year and Month to datetime
data['Date'] = pd.to_datetime(data[['Year', 'Month']].assign(DAY=1))
data.set_index('Date', inplace=True)

# Step 3: Drop unnecessary columns
data = data.drop(columns=['Year', 'Month'])

# Step 4: Handle missing values if any
data = data.fillna(method='ffill')  # Forward fill missing values

# Step 5: Split into train and test
train_size = int(len(data) * 0.8)
train = data.iloc[:train_size]
test = data.iloc[train_size:]

# Step 6: Fit VAR model
model = VAR(train)
```

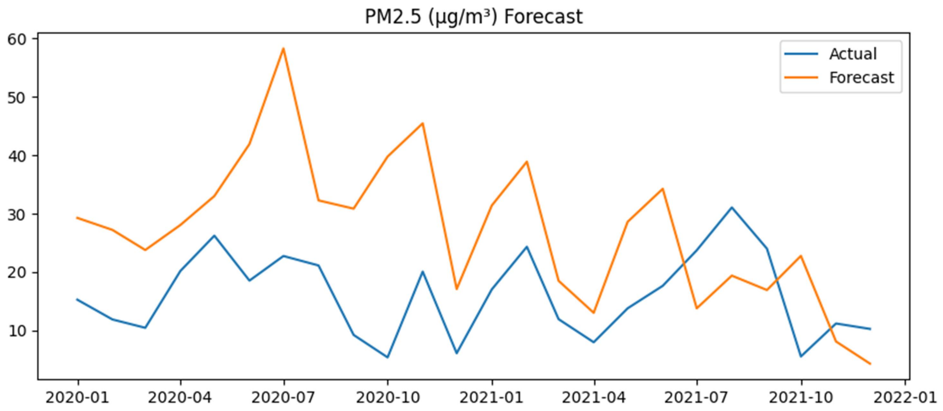
```
lag_order = model.select_order().aic  # You can use BIC, FPE, etc.
model_fitted = model.fit(lag_order)

# Step 7: Forecasting
forecast_input = train.values[-lag_order:]
forecast_steps = len(test)

forecast = model_fitted.forecast(y=forecast_input, steps=forecast_steps)
forecast_df = pd.DataFrame(forecast, index=test.index, columns=data.columns)

# Step 8: Plot predictions vs actual
for col in data.columns:
    plt.figure(figsize=(10, 4))
    plt.plot(test.index, test[col], label='Actual')
    plt.plot(forecast_df.index, forecast_df[col], label='Forecast')
    plt.title(f'{col} Forecast')
    plt.legend()
    plt.show()
```

OUTPUT:



RESULT:

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