**Exp no: 10 Develop vector auto regression model for multivariate time series data forecasting**

**Date: 15/4/25**

**Aim:**

To develop and evaluate a Vector AutoRegression (VAR) model for multivariate time series forecasting using the Netflix stock price dataset.

**Objectives:**

The key objectives are to preprocess and normalize the Netflix stock price dataset along with an additional time series variable, ensure stationarity of the data using statistical methods, build and fit a suitable VAR model, forecast future values (e.g., next 12 months), and finally visualize and evaluate the accuracy of the forecasts.

**Background/Scope:**

Time series forecasting is widely used across industries to predict future trends based on past data. The Vector AutoRegression (VAR) model is a powerful multivariate tool that captures the linear interdependencies among multiple time series. This project uses the classic Netflix dataset, extended with a simulated variable (e.g., weather index) to create a multivariate time series.

**Steps of Implementation:**

**Step 1: Load and Prepare the Multivariate Dataset**

Import the Netflix dataset and create a second variable (e.g., Weather Index) to make it multivariate.

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from statsmodels.tsa.api import VAR

from statsmodels.tsa.stattools import adfuller

from sklearn.preprocessing import MinMaxScaler

df\_nflx=pd.read\_csv("/content/drive/MyDrive/NFLX (1).csv")

# Preprocessing

df\_nflx['Date'] = pd.to\_datetime(df\_nflx['Date'])

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

df\_nflx = df\_nflx[['Close']]

**Step 2: Normalize the Data**

We normalize the data using **MinMaxScaler** to scale the values between 0 and 1. This is essential for neural networks to perform optimally.

np.random.seed(0)

df\_nflx['Market\_Sentiment'] = df\_nflx['Close'] + np.random.normal(0, 5, len(df\_nflx))

# Plot the original series

df\_nflx.plot(figsize=(10, 5))

plt.title("Netflix Close Price & Market Sentiment")

plt.show()

# Normalize the data

scaler = MinMaxScaler()

scaled\_data = scaler.fit\_transform(df\_nflx)

df\_scaled = pd.DataFrame(scaled\_data, columns=df\_nflx.columns, index=df\_nflx.index)

**Step 3: Test for Stationarity and Difference the Data**

Check for stationarity using the Augmented Dickey-Fuller test and apply differencing to make the data stationary.

df\_diff = df\_scaled.diff().dropna()

# Fit the VAR model

model = VAR(df\_diff)

results = model.fit(maxlags=15, ic='aic')

**Step 4: Fit the VAR Model**

Use the VAR model from statsmodels to fit the differenced multivariate time series.

from statsmodels.tsa.api import VAR

# Fit VAR model

model = VAR(df\_diff)

results = model.fit(maxlags=15, ic='aic')

**Step 5: Forecast Future Values**

Forecast future values using the trained VAR model by providing the required lagged values.

orecast\_steps = 12

forecast = results.forecast(df\_diff.values[-results.k\_ar:], steps=forecast\_steps)

**Step 6: Reverse Transform and Visualize Forecast**

Reverse the differencing and normalization to get forecasted values in the original scale, then visualize the results.

forecast\_df = pd.DataFrame(forecast, index=pd.date\_range(start=df\_nflx.index[-1], periods=forecast\_steps+1, freq='B')[1:],

                           columns=df\_nflx.columns)

last\_values = df\_scaled.iloc[-1]

forecast\_reversed = forecast\_df.cumsum() + last\_values

# Inverse transform to original scale

forecast\_original = pd.DataFrame(scaler.inverse\_transform(forecast\_reversed),

                                 columns=df\_nflx.columns, index=forecast\_reversed.index)

# Plot forecast

plt.figure(figsize=(10,5))

plt.plot(df\_nflx.index, df\_nflx['Close'], label='Original')

plt.plot(forecast\_original.index, forecast\_original['Close'], label='Forecast', linestyle='--')

plt.legend()

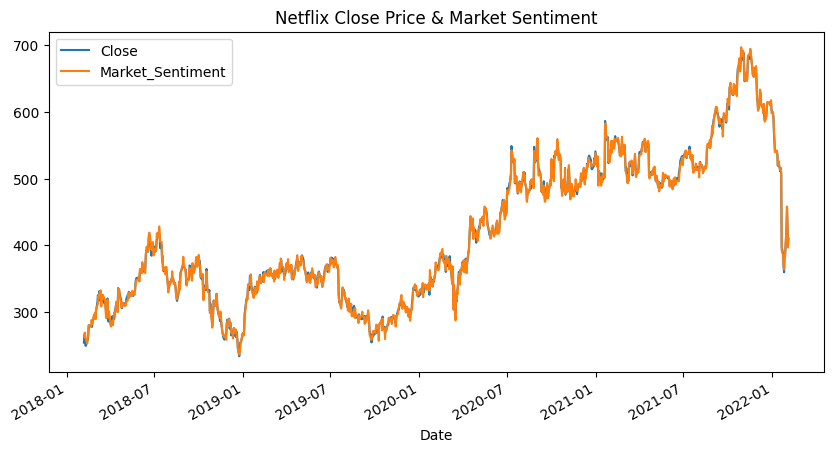
plt.title("Netflix Stock Forecast with VAR Model")

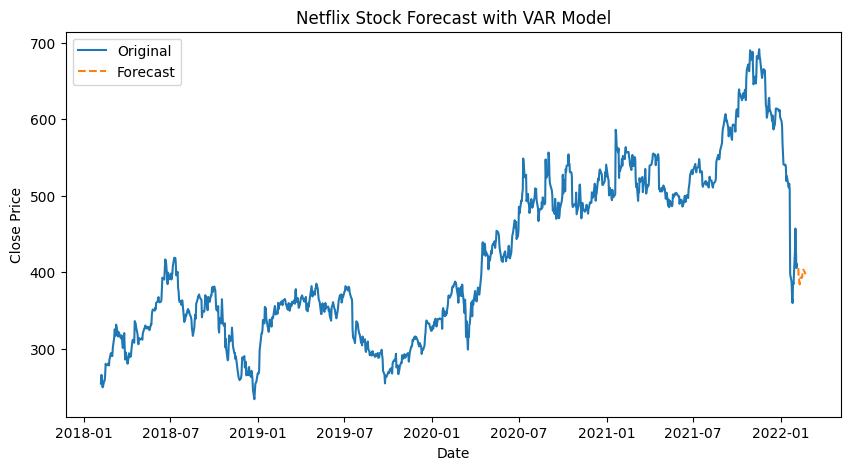
plt.xlabel("Date")

plt.ylabel("Close Price")

plt.show()

**Output:**





**Result:**

Vector auto regression model for multivariate time series data forecasting for netflix stock pricedataset has been successfully implemented.