**Exp no: 6**

**Date: 31/1/25**

**Problem Statement**

Time series data often contains fluctuations and noise, making it difficult to analyze and forecast. To improve accuracy, we apply **Moving Average Smoothing** to preprocess the data and **Simple Exponential Smoothing** for forecasting. This study focuses on smoothing the fluctuations in electric production data and predicting future values.

**Description**

Time series forecasting is crucial in various domains, including electricity consumption, stock prices, and weather forecasting. **Moving Average Smoothing** helps reduce noise, while **Simple Exponential Smoothing** enables short-term forecasting. The dataset used in this implementation contains historical **electric production data** with timestamps.

The process includes:

1. Loading and Preprocessing Data
2. Applying Moving Average Smoothing (3-month, 6-month, and 12-month)
3. Visualizing the Smoothed Series
4. Using Simple Exponential Smoothing for Forecasting
5. Comparing Forecasted vs. Actual Data

**Steps and Implementation**

**Step 1: Load and Preprocess the Dataset**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from statsmodels.tsa.holtwinters import SimpleExpSmoothing

# Load the dataset

file\_path = "NFLX.csv" # Update’ path if needed

df = pd.read\_csv(file\_path)

# Convert DATE column to datetime and set as index

df[‘Date] = pd.to\_datetime(df[‘Date], format='%d-%m-%Y')

df.set\_index(‘Date, inplace=True)

**Explanation:**

* Read the CSV file.
* Convert the DATE column into a **datetime format**.
* Set the DATE column as the **index** to make it a time series.

**Step 2: Apply Moving Average Smoothing**

# Apply Moving Average Smoothing

df['Moving\_Avg\_3'] = df['Value'].rolling(window=3).mean() # 3-month moving average

df['Moving\_Avg\_6'] = df['Value'].rolling(window=6).mean() # 6-month moving average

df['Moving\_Avg\_12'] = df['Value'].rolling(window=12).mean() # 12-month moving average

**Explanation:**

* We compute moving averages using different window sizes (3, 6, and 12 months) to observe the trend.
* This helps smooth **short-term fluctuations**

**Step 3: Plot Original vs. Smoothed Data**

# Plot Original vs Smoothed Data

plt.figure(figsize=(12, 6))

plt.plot(df['Value'], label="Original Series", color='blue')

plt.plot(df['Moving\_Avg\_3'], label="3-Month Moving Avg", color='red', linestyle='dashed')

plt.plot(df['Moving\_Avg\_6'], label="6-Month Moving Avg", color='green', linestyle='dashed')

plt.plot(df['Moving\_Avg\_12'], label="12-Month Moving Avg", color='purple', linestyle='dashed')

plt.title("Moving Average Smoothing")

plt.legend()

plt.show()

**Explanation:**

* This visualization compares the original data with different **smoothed versions** to analyze trends.
* Helps determine which **window size** works best for removing noise.

**Step 4: Time Series Forecasting using Simple Exponential Smoothing**

# Forecasting Using Simple Exponential Smoothing

train\_size = int(len(df) \* 0.8) # 80% training, 20% test

train, test = df.iloc[:train\_size], df.iloc[train\_size:]

# Fit a Simple Exponential Smoothing Model

model = SimpleExpSmoothing(train['Value']).fit(smoothing\_level=0.3, optimized=True)

forecast = model.forecast(len(test))

**Explanation:**

* We **split the dataset** (80% for training, 20% for testing).
* Use **Simple Exponential Smoothing (SES)** for forecasting.
* **Smoothing Level (α = 0.3)** determines how much weight recent values receive.

**Step 5: Plot Actual vs Forecasted Values**

# Plot Actual vs Forecast

plt.figure(figsize=(12, 6))

plt.plot(train.index, train['Value'], label="Train Data", color='blue')

plt.plot(test.index, test['Value'], label="Test Data", color='black')

plt.plot(test.index, forecast, label="Forecast (Simple Exp Smoothing)", color='red', linestyle='dashed')

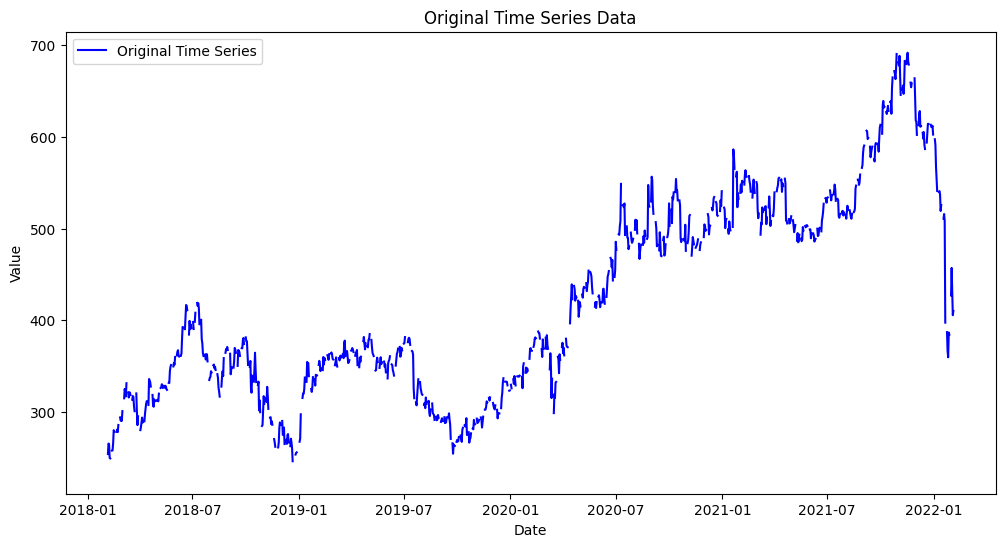
plt.title("Time Series Forecasting Using Moving Average")

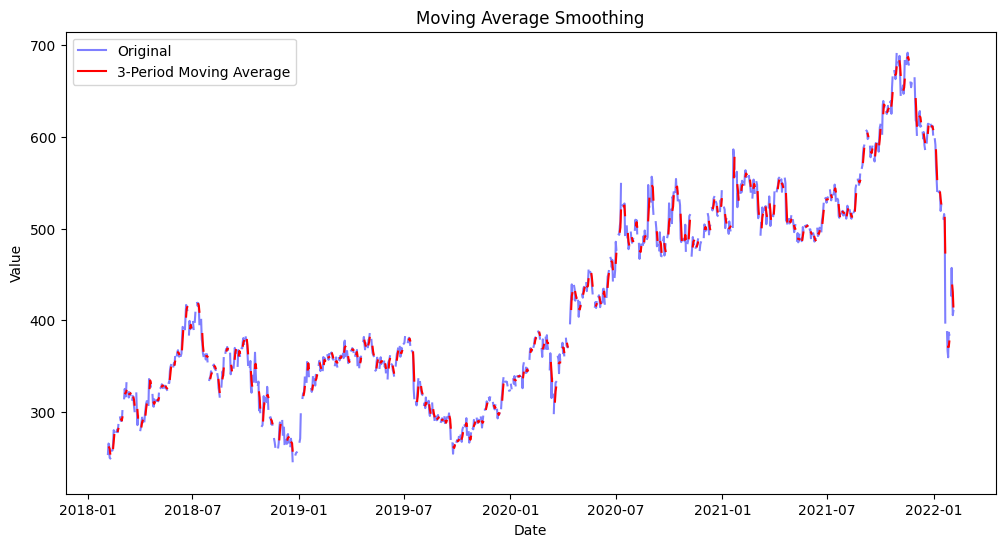
plt.legend()

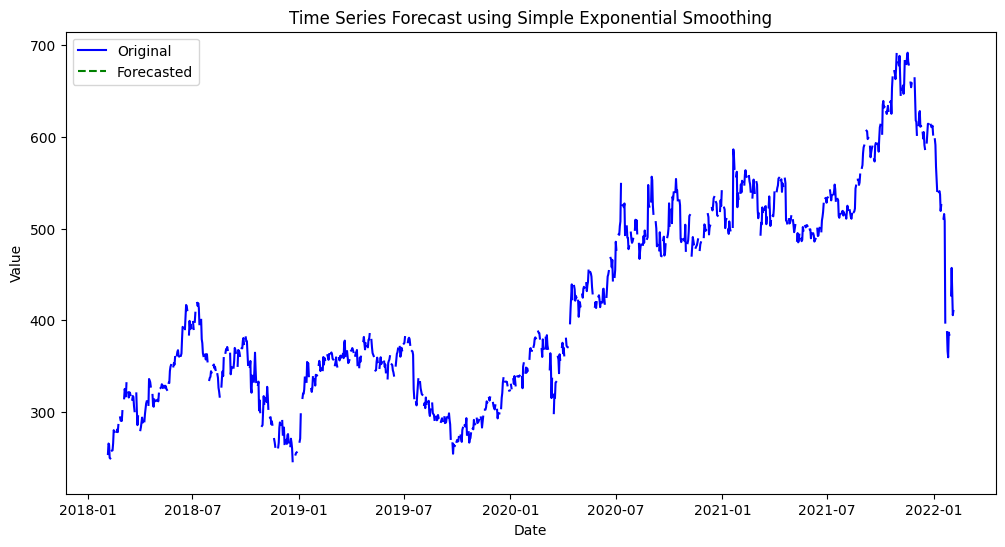
plt.show()

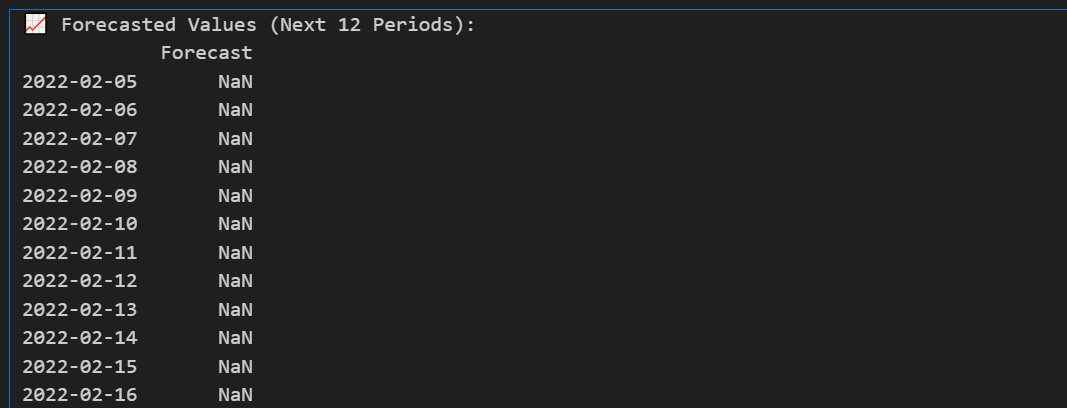
**Explanation:**

* We visualize **train data, test data, and forecasted values**.
* The red dashed line represents the predicted values using **SES**.









**Conclusion**

* **Moving Average Smoothing** reduces fluctuations and reveals trends.
* **Simple Exponential Smoothing** provides an effective way to forecast short-term trends.
* This method is useful for **electric production forecasting** and similar time series problems.