**Develop a linear regression model for forecasting time series data**

**EX:No.3**

**DATE:28/02/25**

**AIM:**

To develop a Linear Regression model for forecasting time series data by leveraging lag features to predict future values.

**ALGORITHM:**

1. Load Data – Read the dataset and set the 'Date' column as the index.
2. Feature Engineering – Create lag features (Lag\_1, Lag\_2, etc.) from the 'Close' price.
3. Handle Missing Values – Remove NaN values created by lagging.
4. Split Data – Divide into training (80%) and testing (20%) sets.
5. Train Model – Fit a Linear Regression model using lag features as predictors.
6. Make Predictions – Use the trained model to forecast future stock prices.
7. Evaluate & Visualize – Compute MAE, MSE, R² score and plot actual vs. predicted values.

**CODE:**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_absolute\_error, mean\_squared\_error, r2\_score

# Load the dataset

file\_path = 'AAPL.csv' # Replace with your dataset path

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

# Convert 'Date' to datetime and set it as index

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

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

# Selecting 'Close' price for forecasting

ts = data[['Close']]

# Create lag features for time series forecasting

ts['Lag\_1'] = ts['Close'].shift(1) # Lag of 1 day

ts.dropna(inplace=True) # Remove NaN values from shifting

# Splitting into training and testing sets

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

train, test = ts[:train\_size], ts[train\_size:]

# Prepare X (features) and y (target)

X\_train, y\_train = train[['Lag\_1']], train['Close']

X\_test, y\_test = test[['Lag\_1']], test['Close']

# Train Linear Regression model

model = LinearRegression()

model.fit(X\_train, y\_train)

# Make predictions

y\_pred = model.predict(X\_test)

# Evaluate the model

mae = mean\_absolute\_error(y\_test, y\_pred)

mse = mean\_squared\_error(y\_test, y\_pred)

r2 = r2\_score(y\_test, y\_pred)

# Print evaluation metrics

print(f"Mean Absolute Error (MAE): {mae:.4f}")

print(f"Mean Squared Error (MSE): {mse:.4f}")

print(f"R-squared (R²): {r2:.4f}")

# Plot actual vs. predicted values

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

plt.plot(test.index, y\_test, label="Actual Prices", color='blue')

plt.plot(test.index, y\_pred, label="Predicted Prices", color='red', linestyle='dashed')

plt.xlabel("Date")

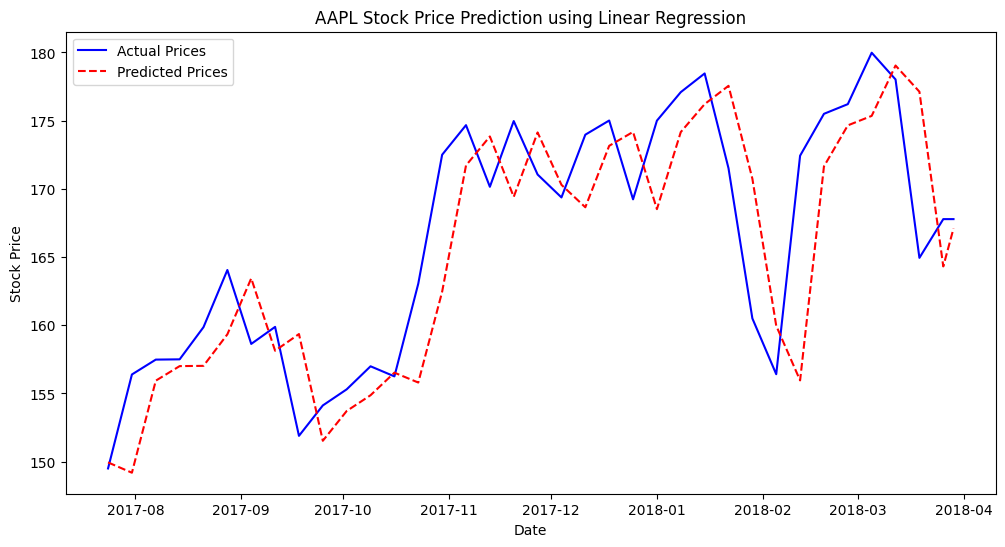
plt.ylabel("Stock Price")

plt.title("AAPL Stock Price Prediction using Linear Regression")

plt.legend()

plt.show()

**OUTPUT:**



Mean Absolute Error (MAE): 4.2930

Mean Squared Error (MSE): 30.6977

R-squared (R²): 0.5916

**RESULT:**

Thus the program has been completed and verified successfully.