**STOCK PRICE PREDICTION**

**Data Science – PHASE – 3**

**Development - 1**

***Introduction:***

Stock price prediction is a common problem in financial analytics and machine learning. It involves using historical stock price data to build models that can forecast future stock prices. This script provides a basic example of stock price prediction using a linear regression model. It loads a dataset from a CSV file, preprocesses the data, builds a model, and makes predictions.

***Prerequisites:***

Before using this script, ensure you have the following libraries installed:

* **pandas:** For data manipulation and analysis.
* **scikit-learn:** For machine learning tasks.

**Step-by-Step Explanation:**

***1. Import Required Libraries***

|  |
| --- |
| import pandas as pd  from sklearn.model\_selection import train\_test\_split  from sklearn.linear\_model import LinearRegression  import matplotlib.pyplot as plt |

* pandas is used for data manipulation and handling the CSV dataset.
* train\_test\_split from sklearn.model\_selection is used to split the dataset into training and testing sets.
* LinearRegression from sklearn.linear\_model is used to create a linear regression model for prediction.
* matplotlib.pyplot is used for plotting the actual vs. predicted stock prices.

***2. Load the CSV Dataset***

|  |
| --- |
| data = pd.read\_csv('stock\_data.csv') |

Load the stock price dataset from a CSV file. Replace 'stock\_data.csv' with the path to your dataset.

***3. Preprocess the Dataset***

|  |
| --- |
| print(data.head())  data = data[['Date', 'Close']]  data['Date'] = pd.to\_datetime(data['Date'])  data.set\_index('Date', inplace=True) |

* Display the first few rows of the dataset to understand its structure.
* Select the 'Date' and 'Close' columns for analysis.
* Convert the 'Date' column to datetime format.
* Set 'Date' as the index for time series data.

***4. Split the Dataset***

|  |
| --- |
| X = data.index.values.astype(int).reshape(-1, 1)  y = data['Close']  X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42) |

* Extract the feature ('X') and target ('y') data. 'X' contains timestamp values, and 'y' contains closing prices.
* Split the dataset into training (80%) and testing (20%) sets.

***5. Create a Linear Regression Model***

|  |
| --- |
| model = LinearRegression()  model.fit(X\_train, y\_train) |

* Create a linear regression model.
* Fit the model to the training data to learn the relationship between timestamps and closing prices.

***6. Make Predictions***

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

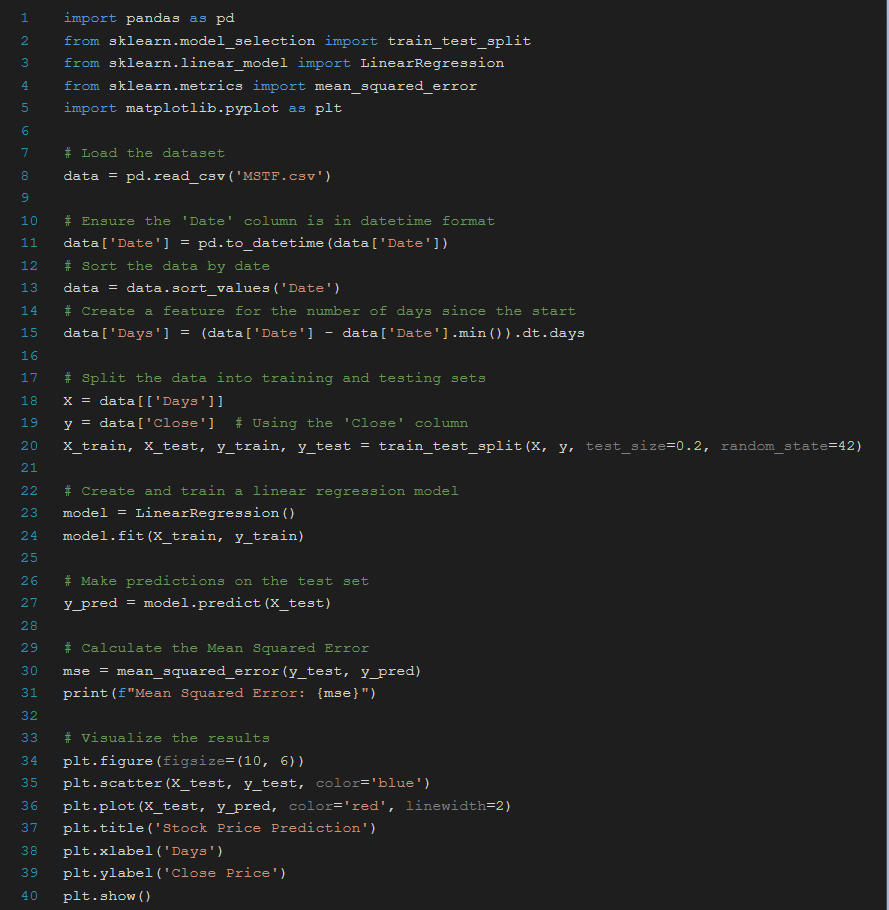
Use the trained model to make predictions on the test data.

***7. Plot Actual vs. Predicted Prices***

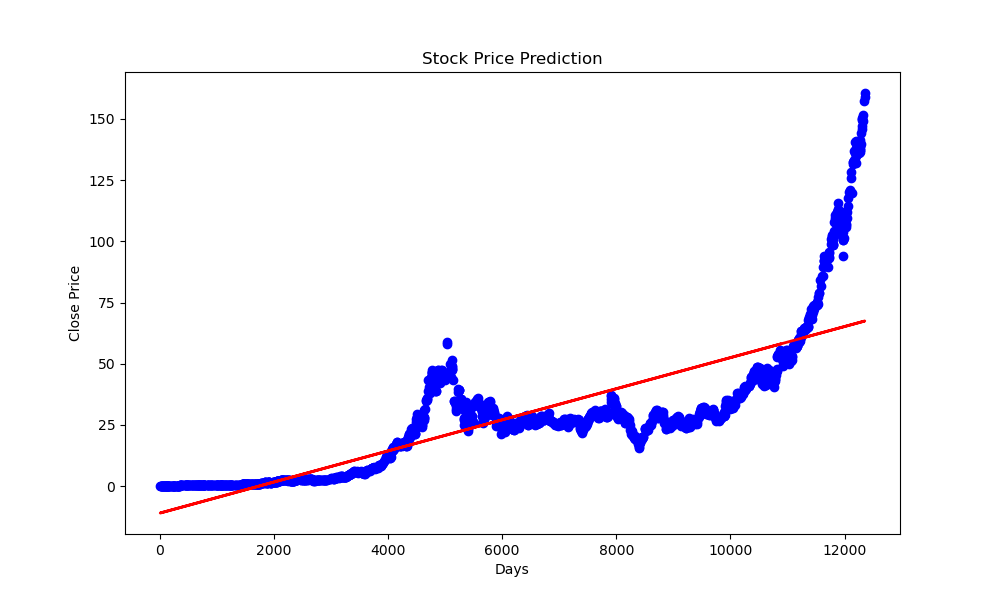
|  |
| --- |
| plt.figure(figsize=(12, 6))  plt.scatter(X\_test, y\_test, color='blue', label='Actual Prices')  plt.plot(X\_test, y\_pred, color='red', linewidth=2, label='Predicted Prices')  plt.title('Stock Price Prediction')  plt.xlabel('Date')  plt.ylabel('Closing Price')  plt.legend()  plt.show() |

* Create a plot to visualize the actual ('Actual Prices') and predicted ('Predicted Prices') stock prices.
* The x-axis represents time (date), and the y-axis represents closing prices.

***CODE SNIPPET FOR THE SAME***

******

***OUTPUT***

******