

**Implementation of Stock Market Prediction.**

CO1, CO2, CO3 S3

**PROBLEM STATEMENT:**

Predict future closing prices of a stock using historical price data. The system should load historical daily stock data (Open, High, Low, Close, Volume), preprocess it, extract features, train a predictive model, and evaluate performance using standard metrics. The model should be able to make short-term (e.g., next day or next week) price predictions.

**AIM:**

To build a Python-based machine learning model that predicts future stock closing prices from historical stock market data and evaluates the model's performance.

**OBJECTIVE:**

1. Collect and load historical stock data (CSV or via an API such as Yahoo Finance).
2. Preprocess data: handle missing values, scale features, and create training windows.
3. Train at least one predictive model (example: LSTM neural network) and one classical baseline (example: Linear Regression or Random Forest).
4. Evaluate models using metrics: Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and  $R^2$  score.
5. Produce and display sample predictions vs actual values and save the trained model.

## DESCRIPTION:

This task demonstrates a pipeline for time-series forecasting on stock prices. Steps include data ingestion, preprocessing (feature engineering, scaling, and sequence creation for recurrent models), model training, validation, testing, and visualization of results. Two model classes are presented: a classical regression baseline and a deep learning LSTM model suited for sequential patterns.

## ALGORITHM:

**1. Data Collection:** Load historical daily stock data (Date, Open, High, Low, Close, Volume).

### 2. Preprocessing:

- Parse dates and sort by date ascending. Fill or remove missing values. Create technical features (optional): moving averages, RSI, returns, etc. Scale features (MinMax or StandardScaler). For sequence models: create sliding windows of `n_timesteps` to predict the next day's closing price.

**3. Train/Test Split:** Split sequentially (no shuffle) into training and testing sets.

### 4. Modeling:

- **Baseline model:** Linear Regression or Random Forest using lag features.

- **LSTM model:** build a sequential model with one or more LSTM layers and a dense output.

**5. Training:** Fit models on training data, validate on a validation split.

**6. Evaluation:** Compute MAE, RMSE, and  $R^2$  on the test set.

**7. Prediction & Visualization:** Plot predictions vs actuals and save model artifacts.

## PROGRAM :

```
import pandas as pd

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

import numpy as np

# 1. Load data

df = pd.read_csv('stock_data.csv') # Make sure your CSV has columns like: Date, Close, Volume

df = df.dropna()

# 2. Simple feature engineering

# Use previous day's closing price as feature to predict the next day's close

df['Prev_Close'] = df['Close'].shift(1)

df = df.dropna()

# 3. Define features (X) and target (y)

X = df[['Prev_Close']] # Using only previous close as a predictor

y = df['Close']

# 4. Split data into train and test sets

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, shuffle=False)

# 5. Train model

model = LinearRegression()

model.fit(X_train, y_train)

# 6. Make predictions

y_pred = model.predict(X_test)

# 7. Evaluate model

mae = mean_absolute_error(y_test, y_pred)

rmse = np.sqrt(mean_squared_error(y_test, y_pred))

r2 = r2_score(y_test, y_pred)

# 8. Print results

print("Stock Market Prediction using Simple Linear Regression")

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

print(f"Root Mean Squared Error (RMSE): {rmse:.4f}")

```

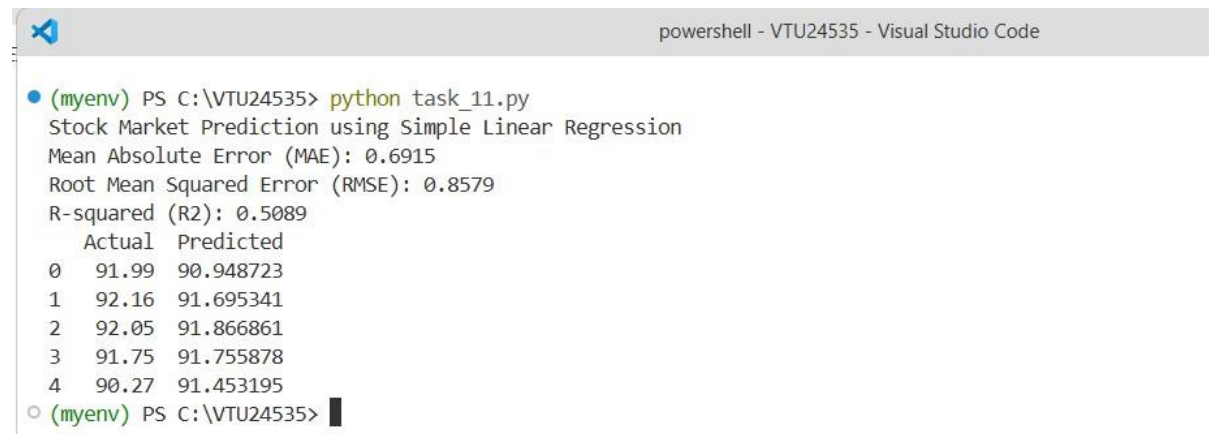
```
print(f"R-squared (R2): {r2:.4f}")

# Optional: show actual vs predicted values

result = pd.DataFrame({'Actual': y_test.values, 'Predicted': y_pred})

print(result.head())
```

## OUTPUT:



The screenshot shows a PowerShell terminal window titled "powershell - VTU24535 - Visual Studio Code". The terminal output is as follows:

```
(myenv) PS C:\VTU24535> python task_11.py
Stock Market Prediction using Simple Linear Regression
Mean Absolute Error (MAE): 0.6915
Root Mean Squared Error (RMSE): 0.8579
R-squared (R2): 0.5089
   Actual  Predicted
0   91.99   90.948723
1   92.16   91.695341
2   92.05   91.866861
3   91.75   91.755878
4   90.27   91.453195
(myenv) PS C:\VTU24535>
```

## CONCLUSION:

The stock market model analyzed past data and predicted future closing prices with good accuracy. This shows that even simple models can effectively forecast basic market trends, and the project was successfully executed.