Stock Price Prediction using LSTM Neural Networks

Project Overview

This project fetches, processes, visualizes, and builds a machine learning model for predicting Google's (or any other) stock prices using historical data from Yahoo Finance.

Step-by-Step Execution

1. Data Collection

- Import required libraries (pandas, numpy, yfinance, etc.).
- Define a date range (last 20 years).
- Download Google's stock data using yf.download().

2. Exploratory Data Analysis (EDA)

- Display dataset structure using .head(), .shape(), .describe(), and .info().
- Visualize the closing stock prices using matplotlib.
- Use a custom function plt_graph() to plot different stock attributes dynamically.

Machine Learning Model (LSTM)

3. Model Preparation

- Import TensorFlow and Keras to build an LSTM-based neural network.
- Define a Sequential model containing LSTM layers to capture time dependencies.
- Save the trained model as "stock_price_model.keras".

4. Working Principle

- The model learns historical stock price patterns using LSTMs, which are suited for sequential data.
- It processes past stock values, identifies trends, and predicts future prices.
- The final trained model is saved for future use.

Model Breakdown & Working

5. Model Architecture (Sequential)

- **Sequential()**: Initializes a sequential model.
- **LSTM(128, return_sequences=True)**: First LSTM layer with 128 units; returns sequences for further learning.

- **LSTM(64, return_sequences=False)**: Second LSTM layer with 64 units, outputting a final sequence.
- **Dense(25)**: Fully connected layer with 25 neurons to process LSTM outputs.
- **Dense(1)**: Final Dense layer with a single neuron for predicting the next stock price.

6. Compilation & Training

- **model.compile(optimizer='adam', loss='mean_squared_error')**:
    - Uses the Adam optimizer for efficient learning.
    - Applies Mean Squared Error (MSE) loss function, best suited for regression tasks.
- **model.fit(x_train, y_train, batch_size=16, epochs=10)**:
    - Trains the model on past stock prices (x_train, y_train).
    - Uses a batch size of 16 for 10 training epochs.

7. Prediction & Application

- **predictions = model.predict(x_test)**:
    - The trained model forecasts stock prices based on test data (x_test).
    - Helps analyze past trends to predict future stock prices.

Conclusion

This project effectively combines data analysis and deep learning to predict stock prices using LSTM networks. It leverages historical stock data, visualizes key trends, and builds a robust prediction model using TensorFlow/Keras. The trained model can be used for future predictions, making it valuable for stock market analysis.