**DSCI-560: Lab 4 Report**

Course: Data Science Practicum  
Team Members: Jaival Upadhyay, Pratham Solanki, Mayank Patil  
Team Name : Guardians of the algorithm(Group 6)

Github : https://github.com/jaivalupadhyay/dsci-560/tree/main/Lab-4

**1) Algorithm Development**

Methodology

The trading algorithm is designed to make smart buy and sell decisions by combining technical analysis and predictive modeling. It uses three main strategies: Moving Average Crossover, Relative Strength Index (RSI), and Time Series Forecasting (ARIMA and LSTM). The Moving Average Crossover looks at two trends—the short-term 50-day moving average and the long-term 200-day moving average—and generates buy signals when the short-term trend moves above the long-term trend, and sell signals when it falls below. The RSI indicator helps determine if a stock is overbought or oversold, signaling potential price reversals. To make longer-term predictions, the algorithm uses ARIMA (statistical modeling) and LSTM (deep learning), which analyze historical stock price patterns to forecast future trends. By combining these techniques, the algorithm aims to capture both short-term trading opportunities and long-term market trends.

trading algorithms implemented –

* Moving Average Crossover, RSI, ARIMA, and LSTM-based predictions.

How were buy and sell signals generated –

* Buy signals occur when the **short-term SMA crosses above** the long-term SMA or when **RSI < 30** (oversold).
* Sell signals occur when the **short-term SMA drops below** the long-term SMA or when **RSI > 70** (overbought).

Model Evaluation –

* **Total Return, Annualized Return, and Sharpe Ratio** were used for evaluating strategy profitability.**MAE and RMSE** were calculated to measure forecasting accuracy for ARIMA and LSTM.

Code Explanation

The code starts by **loading stock price data** from processed\_stock\_data.csv and ensuring it has the necessary information like **date, ticker, and closing price**. It then calculates **50-day and 200-day moving averages** and assigns buy/sell signals based on when these lines cross. The **RSI is computed for 14-day periods**, marking stocks as overbought or oversold. For price forecasting, the **ARIMA model** is trained on 80% of historical stock prices and tested on the remaining 20% to evaluate its accuracy. The **LSTM model** is built using sequences of past **50-day stock prices** to predict future values. The output includes **a log of trade decisions, predicted stock prices, and performance metrics**, which are visualized through graphs and tables.

Output

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a graph

Description automatically generated**

**A screenshot of a graph

Description automatically generated**

**2) Mock Trading Environment**

The mock trading environment simulates stock trading using historical price data from processed\_stock\_data.csv. Users start with a fixed initial capital, evenly distributed across selected stocks.

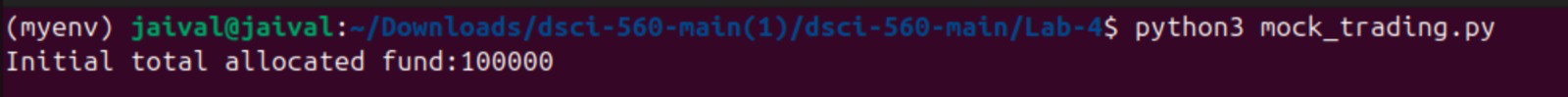
Trading Strategy

The trading strategy is based on a simple moving average (SMA) crossover method to decide when to buy and sell stocks. It tracks two averages: a 5-day SMA (short-term trend) and a 20-day SMA (long-term trend). When the 5-day SMA moves above the 20-day SMA, it signals that prices are rising, so the system buys the stock. When the 5-day SMA drops below the 20-day SMA, it suggests prices may fall, so the system sells. The goal is to buy when an uptrend starts and sell before a downtrend, aiming to maximize gains while minimizing losses.

Code explanation

The code simulates a mock trading system using a moving average crossover strategy to analyze stock trends and execute trades. It loads historical stock price data from processed\_stock\_data.csv, initializes an investment amount, and distributes capital across selected stocks. The strategy calculates 5-day and 20-day SMAs, generating buy signals when the short-term SMA crosses above the long-term SMA and sell signals when it drops below. The system executes trades, updates cash balance, portfolio value, and holdings, and logs transactions. To evaluate performance, it computes total return, annualized return, and Sharpe ratio, along with a portfolio value visualization. The results help assess the effectiveness of the trading strategy.

Output:



A screenshot of a computer

Description automatically generated

A graph on a white background

Description automatically generated

A graph on a white background

Description automatically generated

A computer screen with white text

Description automatically generated

* Transaction Logs – “transaction\_log.csv”:

A table with numbers and letters

Description automatically generated

**3) Team Discussions**

To ensure smooth collaboration, the team met daily to discuss the assignment progress and next steps. The Minutes of Meeting (MoM) document is attached separately, detailing our discussions from Wednesday to Saturday.

**4) Details**

The following files have been submitted as part of Lab 4:

* All code files for the working solution.
* Report\_readme documen explaining how to run the code.
* Minutes of Meeting documenting team discussions.
* GitHub history details for version tracking.

**5) Conclusion**

This lab provided hands-on experience in real-time stock price analysis and algorithmic trading. We successfully implemented and tested a trading algorithm in a simulated environment, evaluated its performance, and participated in a team competition.

**6) Github History**

**A black and white text

Description automatically generated with medium confidence**

A screenshot of a computer

Description automatically generated