Stock Movement Prediction Using Neural Networks

CSCI 4050U – Machine Learning Final Project

Presented by: Smit Kalathia, Samir Ahmadi, Kyle Liao

Project Overview

01

Goal: Predict Stock Movement

Our primary objective is to accurately predict the next-day direction (Up or Down) of individual stock prices.

02

Approach: Neural Network Training

We trained sophisticated neural networks using extensive historical stock return data to identify predictive patterns.

03

Models Utilized

Two distinct neural network architectures were explored: a Multilayer Perceptron (MLP) for its feedforward capabilities and a Long Short-Term Memory (LSTM) network for its strength in sequential data analysis.

04

Deployment: Streamlit Application

The culmination of our efforts is a functional web application built with Streamlit, providing a practical interface for our predictive models.

Why Predict Stock Movement?

Complex Data Landscape

Stock prices exhibit intriguing patterns alongside significant inherent noise, making them a challenging yet rewarding domain for machine learning exploration.

Ideal ML Testbed

The difficulty of short-term stock prediction serves as an excellent benchmark for evaluating the robustness and effectiveness of various machine learning techniques.

Comprehensive ML Pipeline Demonstration

This project showcases a complete end-to-end machine learning pipeline, encompassing critical stages from initial data collection and feature engineering to model training, rigorous evaluation, and real-world deployment.

Dataset Description

Our predictive models are built upon a robust dataset derived from the financial markets, meticulously curated for relevance and breadth.

Data Source: Yahoo Finance API

All raw financial data was retrieved programmatically using the `yfinance` Python library, ensuring access to reliable and up-to-date market information.

Scope: S&P 500 Tickers

Our analysis focuses on the constituent stocks of the S&P 500 index, representing a broad cross-section of the U.S. equity market.

Key Data Points Collected

- Open price
- High price
- Low price
- Close price
- Trading volume

Processed Data: Daily Returns

To normalize volatility and focus on relative changes, all raw price data was transformed into daily percentage returns.

Feature Engineering

Transforming raw financial data into meaningful features is crucial for effective model training.

Daily Returns Calculation

The fundamental feature for our models is the daily return, calculated as the percentage change in adjusted closing prices.

20-Day Sliding Windows

To capture sequential dependencies and trends, we constructed input sequences using a 20-day sliding window approach. Each input to the model comprises the returns from the past 20 trading days.

Label Generation

The target variable (label) for each 20-day window is determined by the stock's movement on the 21st day:

- **1** → if next-day return > 0 (stock goes up)
- **0** → otherwise (stock goes down or stays flat)

Data Processing Pipeline

A streamlined and robust data pipeline ensures that our models are trained on clean, properly formatted, and relevant data.

1. Download Price Data

Historical Open, High, Low, Close, and Volume data for all S&P 500 tickers are retrieved.

2. Compute Returns

Raw prices are converted into daily percentage returns to standardize the data.

3. Build Windowed Sequences

20-day sliding windows of returns are created, along with their corresponding next-day movement labels.

4. Split Data

The dataset is partitioned into training, validation, and test sets to ensure unbiased model evaluation.

5. Convert to PyTorch Tensors

All data is converted into PyTorch tensor format, optimized for neural network operations.

MLP Model Architecture

The Multilayer Perceptron (MLP) serves as our foundational feedforward neural network, designed to capture complex non-linear relationships within the stock return data.

1

Input Layer

Accepts 20 features, corresponding to the 20-day historical returns.

2

Hidden Layer 1

`Linear(20, 128)` followed by `ReLU` activation and `Dropout(0.2)` for regularization.

3

Hidden Layer 2

`Linear(128, 64)` followed by `ReLU` activation.

4

Output Layer

`Linear(64, 1)` followed by a `Sigmoid` activation function to yield a probability.

5

Output

A single value representing the probability that the stock will move **UP** on the next trading day.

LSTM Architecture

The Long Short-Term Memory (LSTM) network is employed to specifically model temporal dependencies, which are critical for time-series data like stock returns.

Input Shape

Each input sequence is structured as `(20 time steps, 1 feature per step)`, reflecting 20 consecutive daily returns.

Temporal Relationship Modeling

LSTMs are inherently designed to recognize and learn patterns across time, making them suitable for sequential stock data.

Core Architecture

- A single `LSTM layer` with a hidden size of 64 units processes the sequential input.
- This is followed by a `Fully Connected layer` mapping to a single output.
- A `Sigmoid` activation function provides the final probability output.

Training Configuration

Optimizing our neural networks requires a carefully selected set of training parameters and methodologies.

Optimizer: Adam

We selected the Adam optimizer for its adaptive learning rate properties, enabling efficient convergence during training.

Loss Function: Binary Cross-Entropy

Given our binary classification task (Up/Down), Binary Cross-Entropy was chosen as the optimal loss function to measure prediction error.

Epochs: 15

The models were trained over 15 epochs, allowing sufficient iterations for parameter adjustment while mitigating overfitting.

Batch Size: 64

Training was conducted using a batch size of 64, balancing computational efficiency with model generalization.

Metrics: Accuracy

Model performance was primarily evaluated using accuracy, representing the proportion of correctly predicted stock movements.

Training Challenges

Noisy & Non-Stationary Data

Stock data is notoriously noisy, with numerous unpredictable factors influencing price movements. Furthermore, its non-stationary nature means statistical properties change over time, complicating long-term pattern recognition.

Random Daily Shifts

Many daily price fluctuations can be attributed to random events or news, making consistent, high-accuracy prediction extremely difficult.

Accuracy Convergence Around 50%

Due to the challenges above, models frequently converge to an accuracy hovering around 50%, akin to a coin flip, highlighting the market's efficiency and unpredictability.

Project Goal: Deployment Focus

Our primary objective was not to "beat the market" or achieve unattainable accuracy, but rather to successfully implement and deploy a complete, functional machine learning pipeline for stock prediction.

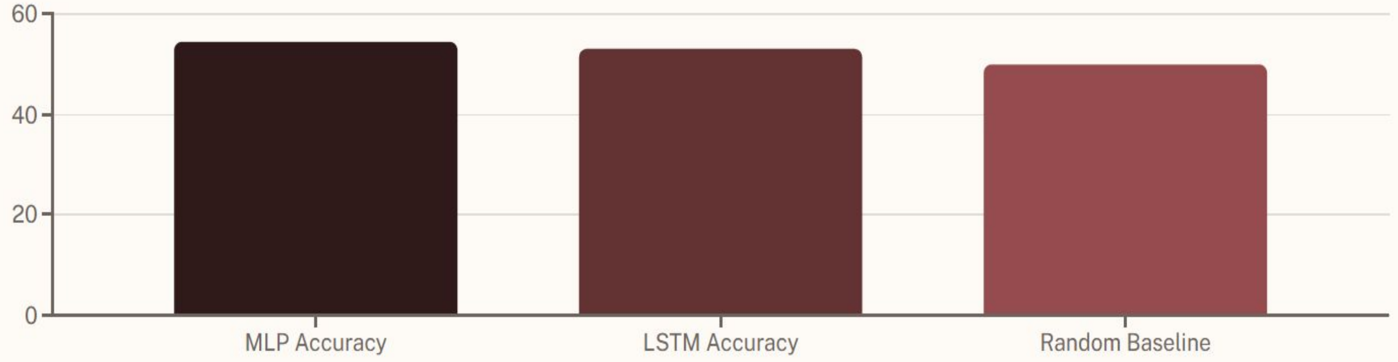
Results: Model Comparison

MLP Model

The Multi-Layer Perceptron (MLP) model achieved a test accuracy of **~54.5%**. This indicates only a marginal improvement over random chance, with performance remaining consistent across various validation runs.

LSTM Model

The Long Short-Term Memory (LSTM) model showed a test accuracy of **~53%**. This performance is similar to the MLP model, suggesting limited predictive power for complex financial time series.



Both the MLP and LSTM models demonstrated only a marginal improvement over a random baseline, highlighting the significant challenges in accurately predicting daily stock movements with these approaches.

Application Features

Our application offers a streamlined process from stock selection to prediction display, integrating several key functionalities to provide actionable insights.

Accepts any stock ticker

The application is designed to accept any valid S&P 500 stock ticker symbol, providing flexibility for user-specified analysis.

Downloads recent OHLCV data

Utilizing the yfinance API, the system automatically retrieves the most recent Open, High, Low, Close, and Volume (OHLCV) data for the selected stock.

Computes return window

Historical data is processed to compute the 20-day return window, forming the essential input for the prediction model.

Display

The output includes the calculated probability of the stock moving up, a clear BUY / DON'T BUY recommendation, and a historical price graph for contextual analysis.

Live Demo

Witness our stock prediction application in action, demonstrating real-time data retrieval, processing, and prediction for any chosen stock ticker.

Input Ticker (e.g., AAPL)

Start by entering a stock symbol like **AAPL** into the application interface.

Display Price Trend

Observe the historical price trend for the selected stock, providing visual context.

Show Probability Output

View the model's calculated probability of the stock moving **UP** along with a **BUY/DON'T BUY** recommendation.

Instantaneous Results

Experience the rapid processing, with predictions generated almost instantly after input.

The entire process, from data fetching to prediction, is designed for speed and efficiency, delivering actionable insights within moments.

Limitations

Our predictive model, while functional, operates under several key limitations that are crucial for understanding its scope and applicability.

No technical indicators included

The current model relies solely on historical price and volume data, omitting the vast array of technical indicators (e.g., RSI, MACD) that often provide crucial contextual signals in financial analysis. This simplifies the model but potentially limits its predictive power.

Only predicts 1-day horizon

Our model is specifically designed for short-term prediction, forecasting price movement for only the next trading day. It is not equipped for multi-day or long-term trend analysis, which requires different modeling approaches.

Accuracy limited by dataset nature

The inherent noise and non-stationary characteristics of financial market data significantly constrain the achievable accuracy. As observed, predictions often hover around a 50% success rate due to the unpredictable nature of daily market movements.

No risk-adjusted return evaluation

The model provides a probability of price movement but does not incorporate risk management or portfolio optimization strategies. It does not evaluate potential returns adjusted for risk, which is a critical component of real-world investment decisions.

Future Work

Our roadmap for the stock prediction model involves several key enhancements to improve its capabilities and accuracy:

Ensemble models (MLP + LSTM + CNN)

Developing an ensemble approach by combining the strengths of multiple models, including Multi-Layer Perceptron (MLP), Long Short-Term Memory (LSTM), and Convolutional Neural Networks (CNN), to achieve more robust predictions.

Expand prediction window (1–5 days)

Extending the model's prediction horizon beyond a single day to forecast price movements over a 1-to-5-day window, offering more strategic insights.

Train on intraday data

Training the model on higher-frequency intraday data to capture finer-grained market dynamics and potentially improve short-term predictive performance.