Boston University
Electrical & Computer Engineering
EC463 Senior Design Project

First Semester Report

AI Trading Platform

Submitted to

Alan Pisano
ECE Department
8 Saint Mary's Street

by

Team 6
The Wolves of Commonwealth Ave

Team Members

Adrian Pawlowski | ajp12@bu.edu
Gabriel Brown | gabrown@bu.edu
Yagiz Idlman | ykaan@bu.edu
Odilon Quevillon | odilon@bu.edu

Submitted: 12/08/2024

Table of Contents

Executive Summary (Odilon)

AI Trading Platform
Team 6 - The Wolves of Commonwealth Ave

Our project addresses the challenges individual investors face in making informed investment decisions in the stock market by developing an AI-driven trading platform that provides actionable recommendations. The platform uses machine learning to provide actionable stock advice, simplifying complex data, enhancing predictive accuracy, and democratizing access to sophisticated trading tools for retail investors.

1.0 Introduction (Gabe)

Individual investors face the challenging task of making informed decisions regarding equity investments. Market volatility and the increasing reliance on algorithmic trading have created the need for accessible and sophisticated trading tools. The customer's problem lies in the difficulty of accurately predicting stock movements and determining whether to buy, sell, or hold their investments.

Our project aims to address this problem by developing an AI-driven trading platform that provides users with actionable insights for fifteen selected equities. These insights include a clear recommendation to buy, hold, or sell individual equities based on a quantitative analysis using machine learning. Our project will allow more people access to quantitative trading recommendations.

These are three main problems that our customers experience:

- Information Overload: Investors often struggle to process massive amounts of financial data efficiently.
- Unpredictable Market Behavior: Traditional trading strategies fall short when markets behave unpredictably.
- Accessibility of Advanced Tools: Sophisticated trading algorithms are often inaccessible to retail investors due to their complexity or secrecy.

Purpose of the Project

The purpose of our project is to increase access to advanced trading insights by creating a user-friendly web application that allows users to make well-informed decisions outlined by our machine learning models. Our platform aims to provide access to complex machine-learning trading algorithms for individuals who would not otherwise.

General Approach

Our approach integrates a series of advanced machine learning and deep learning models:

- Long Short-Term Memory Networks analyze historical stock data and predict future trends.
- XGBoost is used to classify in which a stock price will go up or down.
- Natural Language Processing will incorporate sentiment analysis of financial news, reports, and social media trends to output a sentiment score.

- A Reinforcement Learning algorithm will take these models as inputs and weigh each one to output a recommendation: buy, hold, or sell.

By blending these technologies, we create a robust decision-making engine that adjusts dynamically to market conditions and delivers actionable insights in real time across stocks with varying volatility. Our platform addresses the customer's pain points by:

- Simplifying Complex Data: Presenting users with digestible and actionable outputs, avoiding overwhelming charts or excessive metrics.
- Enhancing Predictive Accuracy: Leveraging machine learning models to capture market patterns and trends more effectively.
- Advanced Tools: Making sophisticated machine-learning algorithms available through a simple and intuitive interface that can be accessible with an internet connection.

Project Highlights and Special Features

- Interactive Web Application provides a seamless, easy-to-navigate interface for users to track real-time stock recommendations
- AI-powered Insights gives users a recommendation engine that integrates multiple data sources, including historical prices, market sentiment, and technical indicators to predict whether to buy, sell, or hold
- Machine learning models are all inputs to the Reinforcement Learning decision layer allowing the predictions to generalize across each stock.

2.0 Concept Development (Odilon)

What?

Our core goal is to develop an automated trading platform capable of generating actionable buy, hold, and sell signals. This platform must integrate advanced financial data analysis with reinforcement learning (RL) to outperform traditional trading strategies. Key pain points include the need for precise sentiment-driven market insights, reliable stock recommendations, and robust system performance under real-world conditions.

How?

Conceptual Approach to Solve the Problem:
We address these needs by developing an AI-driven trading platform that integrates an LSTM, an XGboost, a Natural Language Processing, and a Reinforcement Learning model.

- **Sentiment Analysis Integration**: We process financial news data and social media sentiments, ensuring it aligns with the stock universe for our RL model. By subscribing to an API for reliable data collection and applying Natural Language Processing (NLP), we convert market sentiments into actionable insights.
- **Reinforcement Learning Implementation**: Using a combination of LSTM, XGBoost, and NLP-driven inputs, we optimize an RL model that dynamically learns and improves trading signals. The reward function will be fine-tuned to maximize risk-adjusted returns.
- **Web Application Development**: We build an intuitive front-end interface supported by a scalable back-end to ensure real-time functionality and user accessibility.
- **Trading Strategy Development**: After backtesting the RL model, we evaluate custom trading strategies and compare them against benchmarks to validate their effectiveness.

Why?

Our proposed concept is a comprehensive, adaptive solution tailored to the dynamic nature of financial markets. Here's why it is the best choice for addressing our problem statement:

- **Dynamic Market Adaptability (Reinforcement Learning)**

    - Unlike traditional machine learning models, which rely on static patterns, reinforcement learning (RL) dynamically adapts to changing market conditions.
    - The ability to optimize reward functions ensures that the system focuses on maximizing long-term, risk-adjusted returns rather than short-term gains.
- **Enhanced Market Insights (Sentiment Analysis)**
    - Financial news and social media sentiments are powerful predictors of market movements. Integrating sentiment analysis allows the model to account for qualitative factors, providing a competitive edge.
    - By transforming sentiment data into actionable signals, we bridge the gap between market perception and trading strategy.
- **Robustness Through Modular Integration**
    - Combining LSTM, XGBoost, and NLP ensures the RL model leverages diverse data sources, increasing accuracy and robustness.
    - The modular approach allows flexibility in refining components independently without disrupting the entire system.
- **Comprehensive Evaluation and Optimization**
    - Backtesting and benchmarking the RL model against industry standards validates its reliability and performance.
    - Custom trading strategies developed from the RL outputs ensure practical application and profitability in real-world scenarios.
- **Scalable and User-Friendly Design**
    - A web-based application with a responsive front-end makes the system accessible to users, while the back-end ensures scalability and performance.

Why Not Alternatives?

- **Traditional ML Models**
    - These models are effective at analyzing static relationships but struggle to adapt to dynamic market trends and new data streams, limiting their efficacy in real-world trading.
- **Heuristic Strategies Based on Sentiment Alone**
    - While sentiment-driven rules can offer insights, they lack the sophistication and adaptability of RL-based strategies, resulting in inconsistent and suboptimal performance.

This concept harnesses the adaptability of RL, the predictive power of sentiment analysis, and a robust technical framework to deliver a state-of-the-art trading platform.

3.0 System Description (Yagiz)

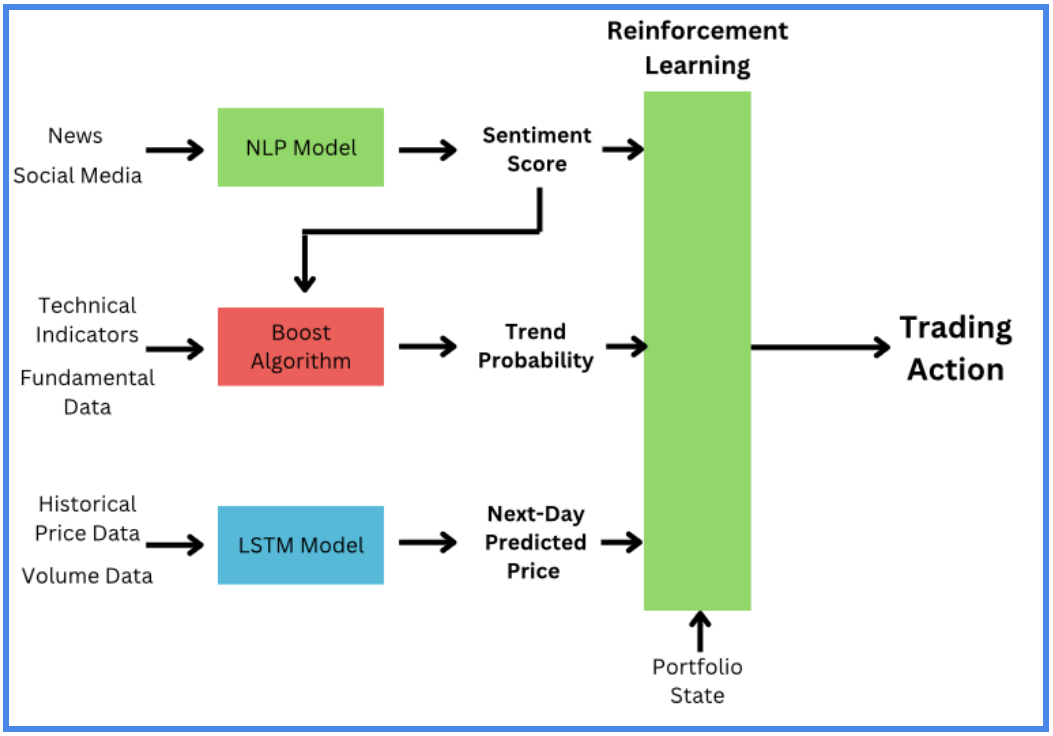

Figure 1: AI Trading Model Architecture

This section provides a detailed explanation of our **Machine Learning-Driven Trading Platform's** technical design. The system contains sophisticated ML models, a variety of data sources, and a RL model that is able to make optimal trading decisions. Each subdivision in Figure 1 is created to process certain data types and submit final information to the trading activities as a group for the RL model to determine a final **Trading Action**. The architecture at the general level, as shown in Figure 1, depicts how data transitions through the major parts.

High-Level System Architecture

Our proposed solution consists of four interconnected subsystems:

- **NLP Model**
    - **Inputs:** Real-time news feeds and social media data
    - **Functionality:** The model will be used to figure out the sentiment of market-related news.
    - **Output:** A numerical score in the range from -1 to 1. The values closest to +1 signify a positive sentiment, and those close to -1 signify negative sentiment.

    - **Role:** This figure, therefore, becomes intelligence that one can interpret in the face of changes in the market.
- **Boost Algorithm:**
    - **Inputs:** Technical indicators (for example Moving Averages, RSI, MACD) and fundamental metrics (for example P/E Ratio, Volume data).
    - **Functionality:** The boost algorithm analyzes varying data and finds non-linear connections.
    - **Output:** A probability measure that gives the chance of a stock going up or down a certain amount (e.g., bullish or bearish) in the future.
    - **Role:** The computation of probabilities for short-term predictions of the market will allow our model to react and act on both technical and fundamental signs.
- **LSTM Model (Long Short-Term Memory)**
    - **Inputs:** Historical price and volume data over a configurable time window (such as the past 30–90 days)
    - **Functionality:** The LSTM model is optimized for time-series forecasting. It captures long-term relationships in the data, enabling it to predict the Next-Day Price of a given asset.
    - **Output:** A numerical prediction of the asset's price for the following trading day
    - **Role:** This prediction helps in decision-making by providing a forward-looking perspective on market trends.
- **Reinforcement Learning Component:**
    - **Inputs:**
        - Sentiment Score (NLP model)
        - Trend Probability (Boost algorithm)
        - Next-Day Predicted Price (LSTM model)
    - **Functionality**: The RL model optimizes trading decisions. It does so by simulating many portfolio states and trading scenarios, as well as by evaluating the long-term rewards associated with each action (buy, sell, hold).
    - **Output:** A **Trading Action** that maximizes portfolio returns while minimizing risk
    - **Role:** The RL component serves as our decision-maker, combining predictions from the other models.

Data Flow and Integration

The platform's operation begins with incorporating data streams, which are processed by subsystems:

- **Data Collection and Preprocessing**
    - **News and Social Media:** Scraped from APIs (like RSS feeds, News APIs), tokenized, and cleaned for NLP processing
    - **Technical Indicators:** Calculated using real-time market data
    - **Historical Price and Volume Data:** Retrieved from trading APIs (like Alpha Vantage, Yahoo Finance)
- **Processing and Decision Flow**
    - The NLP model processes text data to produce a Sentiment Score.
    - Technical and fundamental data are analyzed by the Boost algorithm, generating a Trend Probability.
    - The LSTM model predicts the Next-Day Price based on historical data.
    - These outputs are fed into the RL component to determine the optimal trading action.

User Interface

Our platform's user interface (UI) is designed with accessibility and usability:

- **Real-Time Dashboards:** Interactive visualizations display sentiment trends, predicted prices, and portfolio performance metrics.
- **Trade Recommendations:** Users receive clear, actionable signals.
- **Customizability:** Users will have the ability to create their own personalized portfolios.

4.0 First Semester Progress (Adrian)

1. Initial Setup and Planning

Our team began the semester by establishing a strong foundation for the project. We created a GitHub repository to centralize code management and to enable efficient version control across all team members. We conducted extensive research to develop a comprehensive design plan outlining how the LSTM, XGBoost, and Reinforcement Learning (RL) models would interact within our algorithmic trading platform. This planning phase also included setting up an Anaconda environment to ensure consistent dependency management and streamline the development process for all team members.

2. LSTM Model Development

The team worked on developing the LSTM model for stock price prediction. Initially, we coded and tested a baseline LSTM model, but its performance revealed room for improvement during hyperparameter tuning and backtesting, leading to further research and the implementation of an Attention LSTM model, informed by insights from academic literature demonstrating its efficacy in processing financial time series data. To enhance the model's predictive capabilities, we incorporated fundamental indicators derived from our team's analysis, including metrics like the Debt-to-Equity Ratio, Price-to-Book Ratio, and Beta. Initial backtesting and error metric evaluations, such as MAE, RMSE, and MAPE, validated the functionality of the Attention LSTM model. While results showed potential, they also revealed opportunities for further optimization, which we aim to explore in the next phase.

3. XGBoost Model Development

In parallel with the LSTM model, our team focused on developing an XGBoost model designed for classification tasks, specifically predicting stock price movements (up or down). We performed detailed research to identify effective methodologies for integrating the XGBClassifier. A Genetic Algorithm (GA) was implemented to optimize feature selection by identifying the most relevant technical indicators, such as Moving Averages, Bollinger Bands, and RSI. We iteratively tuned the XGBoost model and GA parameters to improve prediction accuracy and relevance. Our backtesting process highlighted the model's ability to adapt to varying stock characteristics, as demonstrated by its performance across TSLA, MSFT, and KO. The XGBoost model complements the LSTM model by focusing on directional predictions, adding a layer of robustness to our overall platform.

4. Reinforcement Learning (RL) Model Initiation

We began initial development of the Reinforcement Learning (RL) component using the Proximal Policy Optimization (PPO) algorithm. While progress was made in defining states, actions, and rewards, challenges arose with tuning the reward system and defining the action space. Despite delays caused by the incomplete NLP component, we initiated early training efforts using simulated data. The RL model is designed to act as a decision-making layer, dynamically adjusting weights assigned to the outputs of the LSTM and XGBoost models to optimize portfolio returns.

5. Sentiment Analysis Component

We are currently leveraging the Alpha Vantage API to conduct sentiment analysis based on financial news. However, our team plans to develop and implement a custom NLP model in the next phase to enhance the accuracy and relevance of sentiment predictions. This custom model will analyze news articles, earnings reports, and social media to generate sentiment scores, which will serve as inputs for the RL model.

6. Prototype and Testing Framework

To evaluate the performance of our models, we developed a comprehensive prototype and testing framework. This framework includes a dedicated testing folder for organizing scripts and documenting results. Extensive backtesting was conducted for both the LSTM and XGBoost models using data from TSLA, MSFT, and KO. We designed a profit/loss (P/L) simulation to assess the models' practical applicability. For example:

- TSLA exhibited high potential for profit due to its volatility but showed larger drawdowns.
- MSFT demonstrated consistent and steady growth, making it a suitable choice for balanced strategies.
- KO delivered stable performance with minimal drawdowns, aligning with conservative investment approaches.

Metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), Sharpe Ratio, and Maximum Drawdown were used to evaluate model performance.

7. Summary of Key Results from First Deliverable Testing

Our first prototype tests yielded valuable insights:

- **LSTM Results:**
    - Achieved relatively low error metrics, including MAE and RMSE, for all three stocks
    - Highlighted a need for improved risk-adjusted performance, particularly for volatile stocks like TSLA

Tesla Results:

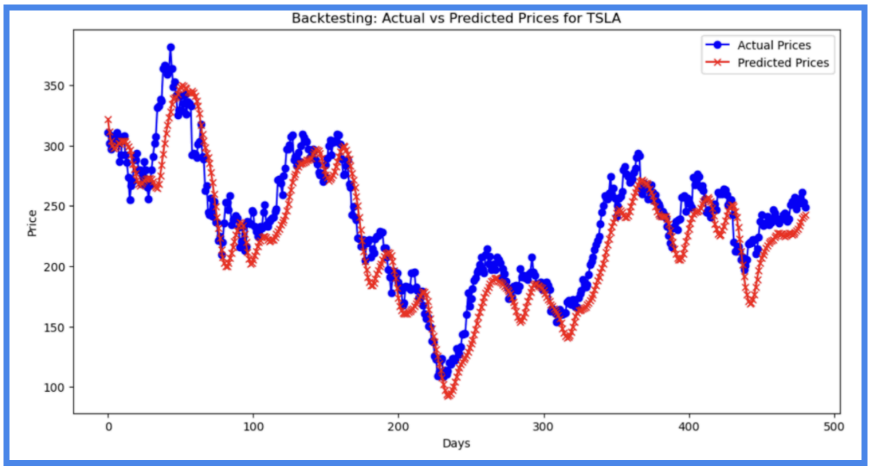

Figure 2: Backtesting: Actual vs Predicted Prices for TSLA

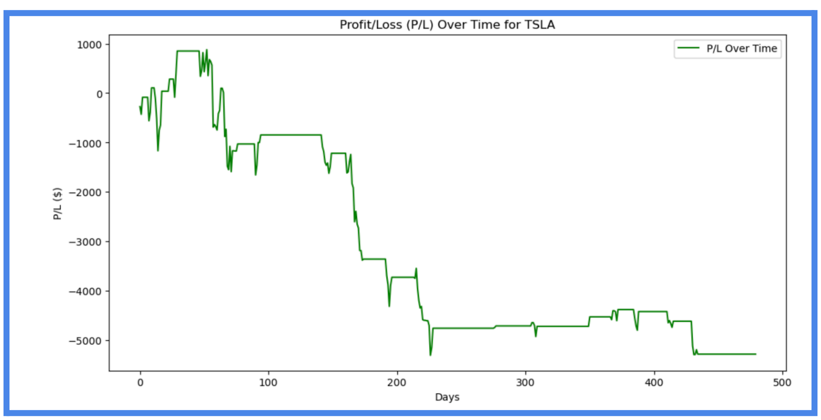

Figure 3: P/L Over Time for TSLA

Coca-Cola Results:

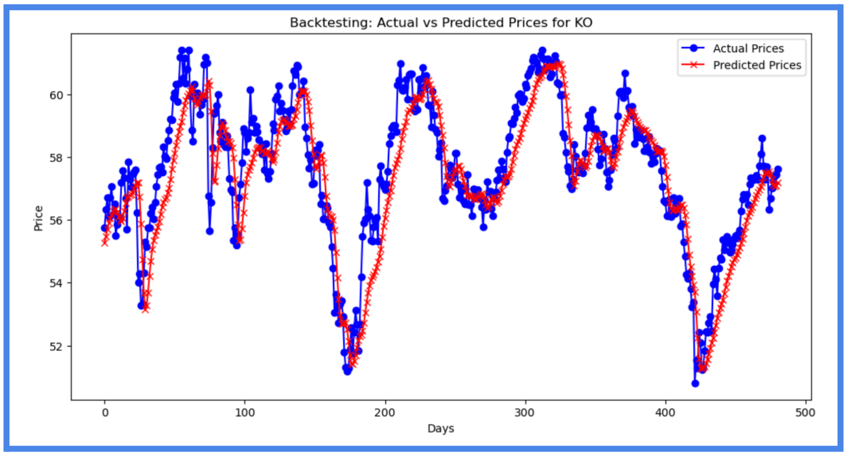

Figure 4: Backtesting: Actual vs Predicted Prices for KO

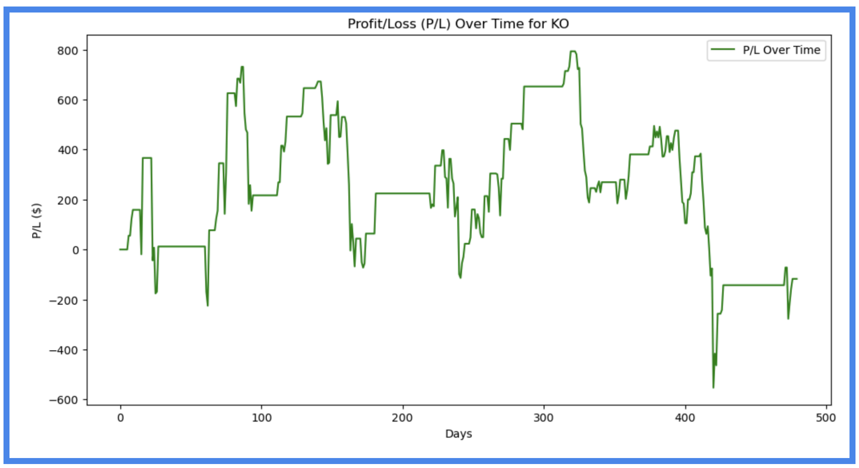

Figure 5: P/L Over Time for KO

Microsoft Results:

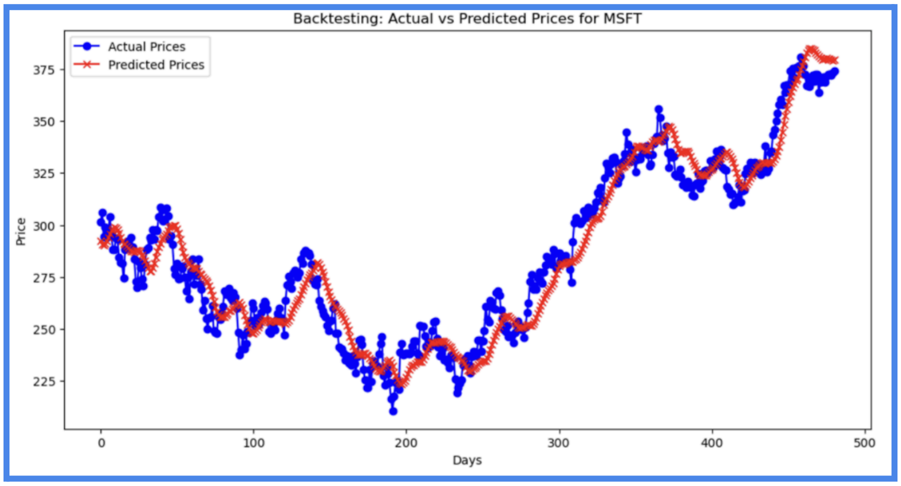

Figure 6: Backtesting: Actual vs Predicted Prices for MSFT

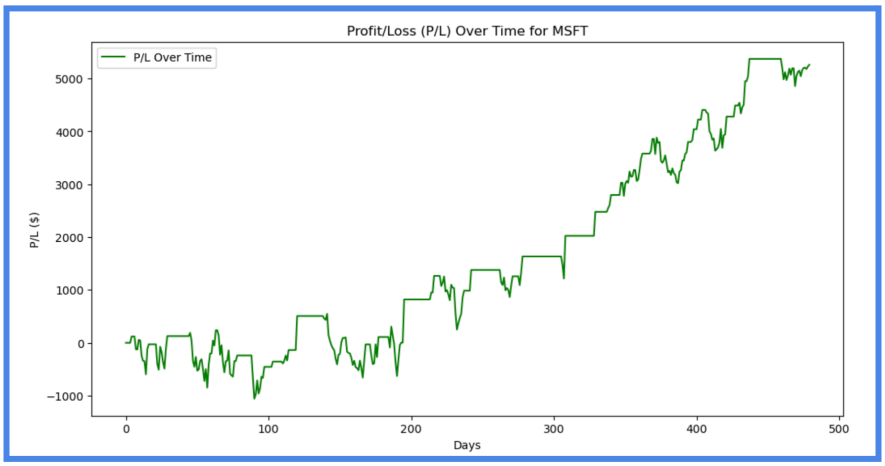

Figure 7: P/L Over Time for MSFT

- **XGBoost Results:**
    - Demonstrated adaptability across different stock profiles, with TSLA showing the highest overall P/L
    - Precision, recall, and F1 scores were consistent, indicating reliable directional predictions.
    - The model achieved a cumulative P/L growth trajectory with moderate drawdowns, particularly for MSFT and KO.

Tesla Results:

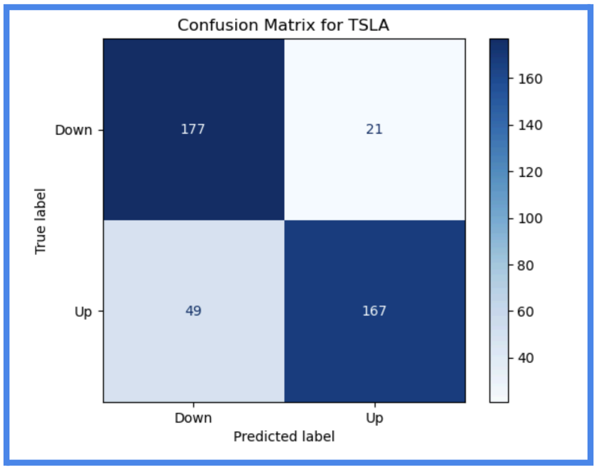

Figure 8: Confusion Matrix for TSLA

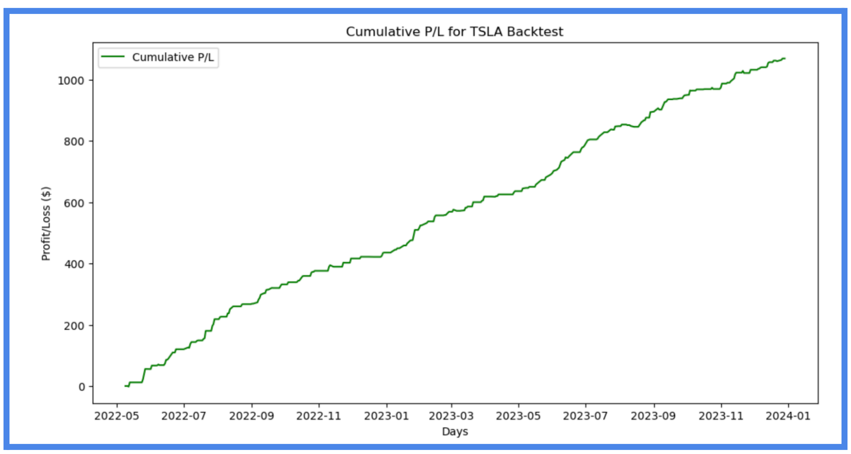

Figure 9: Cumulative P/L for TSLA Backtest

Coca-Cola Results:

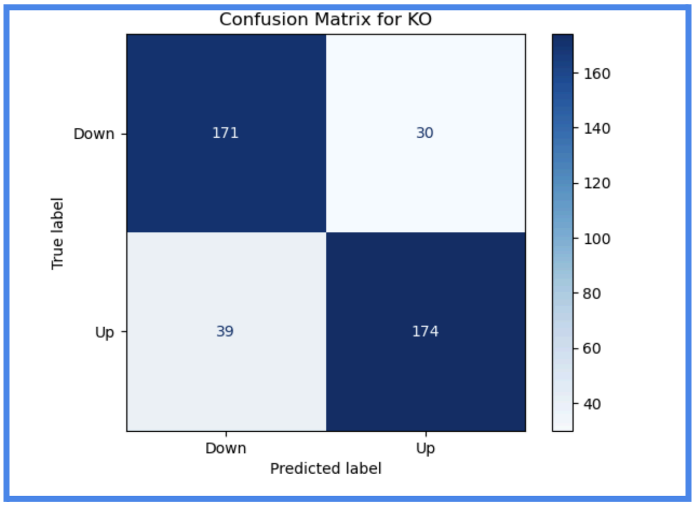

Figure 10: Confusion Matrix for KO

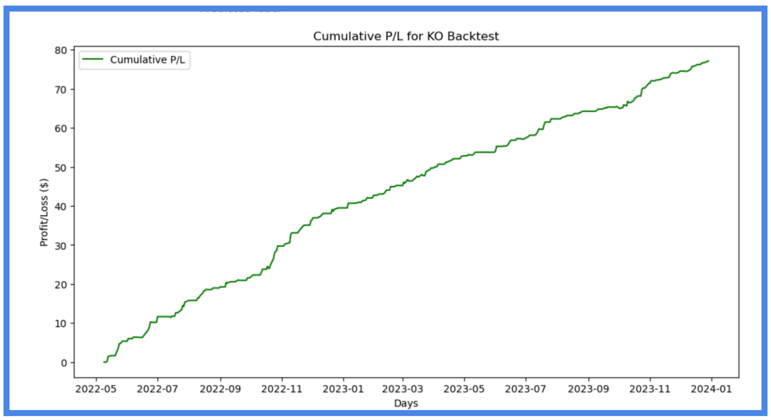

Figure 11: Cumulative P/L for KO Backtest

Microsoft Results:

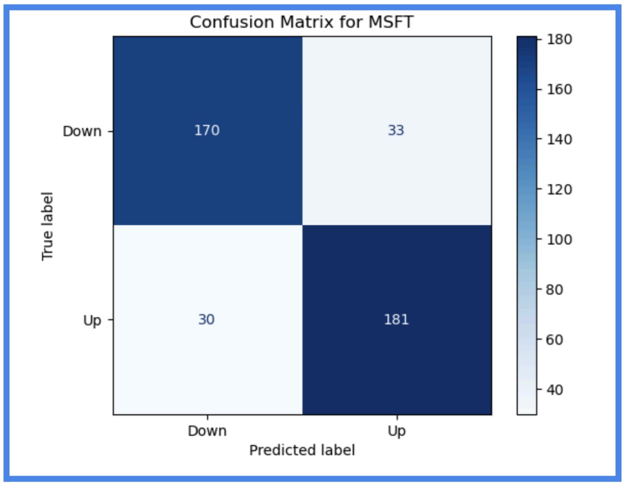

Figure 12: Confusion Matrix for MSFT

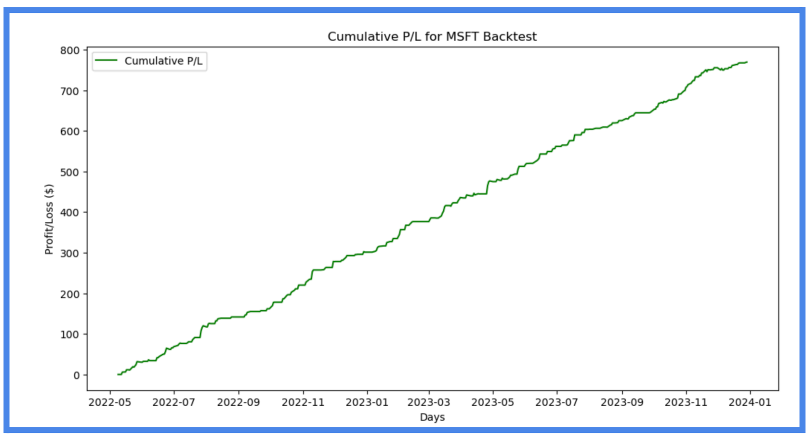

Figure 13: Cumulative P/L for MSFT Backtest

5.0 Technical Plan (Gabe)

Task 1: Sentiment Analysis Integration

Description: Financial news sentiment data will be processed and integrated into our existing Reinforcement Learning model. Tasks include training a sentiment analysis model on labeled financial datasets and validating its accuracy. We also need to ensure we have a reliable way to collect data from headlines and social media regarding the stocks used for training the model by subscribing to an API. Additional steps will involve preprocessing this data for consistency and ensuring it aligns with our stock universe.

Task 2: Reinforcement Learning Model Implementation

Description: We will refine and optimize the reinforcement learning model based on earlier research. This stage process involves improving reward functions, tuning hyperparameters, and integrating inputs from the LSTM, XGBoost, and NLP components. The goal is to develop a fully functional RL model capable of generating actionable buy, hold, and sell signals. Particular attention will be given to stability during training and deep analysis on backtesting.

Task 3: Reinforcement Learning Model Backtesting

Description: Backtesting of the RL model will be conducted to assess its performance under real-world conditions. Historical stock data will be used to evaluate the model's ability to generate accurate recommendations. Tasks include defining evaluation metrics (e.g., profit/loss, Sharpe ratio), analyzing the model’s decision-making process, and comparing its performance to benchmarks developed from our LSTM and XGBoost model alone. Insights gained will be used for further refinement of the RL model.

Task 4: Web Application Development

- Task 4A: Backend Development
  Description: We will develop the backend architecture and implement APIs to expose our Python models. This involves ensuring that the backend can handle real-time requests efficiently, securing data transfer, and enabling seamless integration with the front-end. We will also evaluate and select a hosting service for the backend.
- Task 4B: Frontend Development
  Description: We will develop an interactive front end to display stock recommendations with seamless API integration. This includes designing an intuitive user interface that gives access to the predetermined stocks we train the

models on. Tasks also include planning for deployment and responsiveness for different devices.

Task 5: Trading Strategy Evaluation

Description: We will develop and rigorously test custom trading strategies based on AI outputs. Backtesting will be performed using historical stock data to validate the effectiveness of the strategies. Metrics such as cumulative returns, drawdown, and risk-adjusted performance will be analyzed. We will also take into account transaction costs in the strategy. Based on these evaluations, we will select our trading strategy.

Task 6: Final Testing and Optimization

Description: We will perform system-wide functional testing and optimization to ensure the entire platform works cohesively. This includes testing real-time performance, user experience, and predictive accuracy. Specific tasks include debugging backend and frontend code, validating data flow, analyzing user feedback, and ensuring the system is production-ready.

Task 7: Report and Comparison

Description: After completing the project, the team will prepare a comprehensive report detailing the development process, performance metrics, error analysis, trading strategies, and the platform's profitability. The report will also let us compare our platform against industry-standard quantitative trading systems, showing how close we could get our returns in comparison to theirs.

6.0 Budget Estimate (Odilon)

Since we are not currently performing trades and do not intend to utilize the platform, our required budget is **$0**. Nevertheless, the following proposed budget reflects costs associated if we were to develop and test the platform with live trading. The budget is structured to reflect necessary components for development and testing, considering the possibility of receiving donated items when needed.

Major Budget Items:

1. **Investing Capital**
    - **Cash allocated for algorithmic trading:** For practical purposes of the project
        - *Estimated:* No required minimum but starting with $1,000 would be reasonable as it would allow for diversification
        - *Potential Donation*: ECE Department
2. **Computational Resources**
    - **Cloud Computing Services**: For training and testing the reinforcement learning model, as well as hosting the backend of the web application
        - *Estimated Cost*: $1,000 for initial development and testing (6 months)
        - *Potential Donation*: Seek credits from providers like AWS, Azure, or Google Cloud
3. **Miscellaneous Items**
    - Includes domain registration for web application, minor hardware needs, and team collaboration tools
        - *Estimated Cost*: $500

Summary of Costs:

Item	Estimated Cost	Sources
Investing Capital	$1,000	ECE Department
Cloud Computing Services	$1,000	Cloud provider credits
Miscellaneous Items	$500	Personal reserve
Total Estimated Cost	**$2,500**	

Key Notes:

- The budget is scalable, allowing for adjustments based on the resources we secure through donations or partnerships.
- If we were to scale up, the primary cost driver would be hardware, as high-performance computing would be essential for analyzing large datasets and updating our stock positions in real time.
- We plan to approach vendors and service providers for possible donations or discounts, especially given the educational and research-oriented nature of the project.

This budget is designed to meet short-term project needs while leaving room for future expansion if large scale live trading is pursued.

7.0 Attachments

7.1 Appendix 1 – Engineering Requirements

Team #6 Team Name: The Wolves of Commonwealth Ave

Project Name: AI Trading Platform

5.1 Data Collection and Preprocessing
Historical Stock Data: We must gather at least ten years of daily stock data for our equity including open, high, low, and close prices. We will need technical indicators in the data including trading volume and moving averages. We must format the data to train our LSTM from it. **Real-Time Data:** We will integrate a real-time financial data source for continuous updates on stock markets and relevant market indicators using a public API. This will be displayed to our users via our web application.
5.2 LSTM Model
Prediction Accuracy: The LSTM model must predict daily stock prices with an accuracy, measured by the mean squared loss, of less than 0.10 on the test set.
Overfitting Prevention: The model must implement regularization techniques to prevent overfitting. L2 Regularization with tunable lambda values will be applied to limit model complexity and improve generalization.
5.3 XGBoost Algorithm
Signal Classification: The model must classify trading actions (buy, sell, hold) with at least 70% precision. In addition to precision, the model's performance will be evaluated using recall and F1-score to ensure a balanced approach, particularly in minimizing false positives and false negatives. **Feature Integration:** The model must combine technical indicators, sentiment scores, and LSTM predictions to generate its trading signals.
5.4 Reinforcement Learning for Trading Strategy Optimization
State-Action Space: The reinforcement learning agent must support three trading actions (buy, sell, hold) and incorporate input features from the LSTM model, XGBoost, and Sentiment Analysis. **Backtesting:** The system must support backtesting for a given equity. **Reward Function:** The reward function must incentivize profit maximization while penalizing large drawdowns.
5.5 Web Application
Backend Architecture: The system must include a robust backend that handles requests from the user interface, processes data, and communicates with the machine learning models and a database through REST APIs. **Dashboard Visualization:** A Web-based dashboard that provides real-time visualization of a stock price and trading signals that we determine. **Cloud Integration:** The web application will be hosted on a cloud service and be accessible to all users.
5.6 Return Goals
Initial Model: Return of 0% on the first model with no loss.
Published Model: Return of > 5% on the published model.

7.2 Appendix 2 – Gantt Chart