Final Capstone Project:

Algorithmic Stock Trading System

Group 6: Nathan Metheny, Javon Kitson, Adam Graves

University of San Diego

AAI-590: Capstone Project

Professor: Anna Marbut

April 15, 2024

Introduction

Our objective is to demonstrate the application of Deep Reinforcement Learning (DRL) in the domain of algorithmic stock trading. By leveraging advanced machine learning techniques, we aim to develop a system capable of taking a range of input matrix and then making informed trading decisions autonomously. This system is designed to adapt dynamically to the fluctuating stock market while aligning with the personalized risk tolerance of each user. This involves the creation of a model that can analyze investors portfolio data, calculate a risk factor that is attached to each individual’s portfolio, historical stock data, understand market trends, and execute trades with the goal of maximizing returns. The demonstration will cover the setup, training, and evaluation of the DRL model, showcasing its potential to outperform traditional trading strategies.

In the core of our system, each trader’s portfolio is analyzed to determine a unique risk factor, which is pivotal in customizing the trading strategy. This risk factor is derived from a comprehensive analysis of the individual's past trading history, financial goals, and risk tolerance. It serves as a critical input to the DRL algorithm, ensuring that the trading decisions not only pursue profitability but also conform to the individual's risk appetite. This approach allows our system to manage the portfolio proactively, executing trades that optimize returns while mitigating risk exposure in line with the trader's preferences, and regulatory bodies.

Integrating the individual risk factor into the DRL framework involves training the model to recognize and react to market signals in the context of the user's risk profile and thus the restrictions. The model's reference behavior is designed to balance risk and reward efficiently, guiding the trading algorithm to make decisions that align with the expected risk-adjusted returns. This integration ensures that the system remains robust and responsive, capable of navigating market volatilities while adhering to the risk constraints.

In addition to the technical implementation, we will explore the theoretical foundations of reinforcement learning and its suitability for financial markets. This includes discussing the challenges of applying DRL in a highly volatile environment, such as stock trading, and the strategies used to mitigate these risks. Our demonstration aims to provide a comprehensive overview of how deep reinforcement learning can be utilized to innovate in the field of stock trading, offering insights into both its capabilities and limitations.

Data Summary

The dataset used in this Deep Reinforcement Learning Stock Trading is a paid a licensed dataset from Firstrate Data.

Data Information:

FREQUENCY	DATE RANGE	NUMBER OF TICKERS
1-minute, 5-minute, 30-minute, 1-hour 1-day	Jan 2005 - Mar 2024 (see below dates for each ticker)	10120 Tickers

The bundle comprises 1-minute/5-min/30-min/1-hour intraday data as well as daily end-of-day data for:

7531 most liquid US stocks (includes all active Russell 3000, S&P500, Nasdaq 100, and DJI stocks)

- 2527 most liquid ETFs
- 460 delisted stocks

Both stocks and ETFs are adjusted for splits/dividends, out-of-hours trades are included.

Demonstration comprises historical stock data, including open, high, low, close prices, and volume for a selection of stocks over a specified period. This data is essential for training our DRL model, allowing it to learn and make predictions about future stock movements based on past trends. The dataset includes a diverse range of stocks from various sectors, ensuring a comprehensive learning experience for the model. The period covered by the dataset is from [start_date] to [end_date], encompassing 10,120 stocks. We have structured the system to train and trade on a subset of the dataset that corresponds with the predefined groupings within the stock market (SP 100, DOW 30, NAS 100, etc.). This rich dataset serves as the foundation for our demonstration, enabling the DRL model to simulate trading strategies and evaluate their performance in a controlled, paper trading environment. To enhance the predictive capabilities of our Deep Reinforcement Learning (DRL) model and provide a more nuanced understanding of the stock market dynamics, we augment select variables within our dataset. This augmentation process involves creating new variables or modifying existing ones to include derived metrics, such as moving averages, volatility indicators, and technical analysis signals. These augmented variables offer additional insights into market trends, momentum, and potential reversal points, which are crucial for making informed trading decisions.

In addition we have the portfolio data which is derived from our system input by the paid clients. In this the system builds an account that has some feature engineering to build a risk factor that will be part of the trading model.

Background Information

Stock trading has been a domain of significant interest for both academic researchers, business entrepreneurs, and financial institutions. The goal of maximizing returns while minimizing risk has driven the development of various methods and technologies to predict market movements and make informed trading decisions. Our project focuses on the application of Deep Reinforcement Learning (DRL) in stock trading, aiming to create an autonomous system that can learn from historical data and adapt to changing market conditions.

Traditionally, stock trading strategies have relied on fundamental analysis, technical analysis, and human expertise. Fundamental analysis involves evaluating a company's financial health, market position, and growth prospects to determine its intrinsic value. Technical analysis, on the other hand, focuses on studying historical price and volume data to identify patterns and trends that may indicate future price movements. Human traders use a combination of these approaches, along with their experience and intuition, to make trading decisions.

Current popular algorithmic stock trading systems, such as TradeStation (tradestation.com), have been based on the ability to predict the trading price of the stock on a day-to-day basis. As they advanced, they had the ability to go deeper into the prediction at a certain point of time. The foundation of this was the ability to trade on the condition of the limit price entered.

However, these methods have limitations in capturing the complex dynamics of the stock market and adapting to changing market conditions. Specifically, these techniques fail to consider the entirety of the state space, where all other stocks and their corresponding patterns should be considered. We find that Deep Reinforcement Learning provides significant alpha. DRL combines deep learning with reinforcement learning, enabling an agent to learn optimal actions through trial-and-error interactions with a pre-defined, structured environment. In the context of stock trading, the agent (our DRL model) observes the state of the market (e.g., stock prices, technical indicators) and takes actions (e.g., buy, sell, hold) to maximize a reward signal (e.g., portfolio value -> profit). The agent learns from its experiences and adjusts its strategy over time to improve its performance.

These DRL models are also used in the world of Robotics. These models have the ability to learn robust policies bringing robustness to hyperparameters, and effective performance in a variety of simulated environments. In a research paper “Soft Actor Critic—Deep Reinforcement Learning with Real-World Robots” by T. Haarnoja et al (2018), they have the in deep discussion about the valuable properties of the Soft Actor-Critic (SAC) algorithm. In this they used the models to train a robot to move, a 3-finger dexterous robotic hand to manipulate an object, and 7-DoF Sawyer robot to stack Lego blocks.

The chosen models and architecture for our project are a combination of Deep Reinforcement Learning (DRL) models, a Feedforward Neural Network (FNN), and a Genetic Algorithm (GA), for stock trading, price prediction, and portfolio optimization, respectively. The DRL model is responsible for making trading decisions based on market conditions, while the FNN model is used to predict future stock prices, which serves as additional input to the DRL model. Finally, the GA is responsible for structuring the portfolio for optimal performance based on the trading objective.

We experimented with two popular Deep Reinforcement Learning (DRL) algorithms for stock trading: Soft Actor-Critic (SAC) an off-policy maximum entropy algorithm that provides sample-efficient learning while retaining the benefits of entropy maximization and stability (Haarnoja et all, 2018), and Proximal Policy Optimization (PPO) “seems to strike the right balance between performance and comprehension.” (Van Heeswijk, 2022). Both algorithms are designed to learn optimal trading strategies by interacting with the market environment and adapting to changing market conditions.

We have concluded that the FNNs, with their nonlinear activation functions, can model these complex nonlinear relationships between inputs (market conditions, financial indicators, etc.) and outputs (buy, sell, hold decisions), and makes them suitable for acting as the function approximators within our DRL framework.

Numerous academic articles discussing the use of DRL models to automate stock trading activity are available. One example is an academic research of an Electronic Trading System is the research paper: "Deep Reinforcement Learning for

Automated Stock Trading: An Ensemble Strategy" by H.Yang et al (2020), in which they propose an ensemble strategy combining Proximal Policy Optimization (PPO), Advantage Actor Critic (A2C), and Deep Deterministic Policy Gradient (DDPG). This approach integrates the strengths of these three actor-critic-based algorithms, aiming to create a robust system that adapts to various market conditions. In a way like ours but using a combination of different models. This is a great example of how there are multiple model for the same solutions. Both our systems are advanced, unlike single-model systems. This ensemble method leverages the collective intelligence of multiple models to improve decision-making accuracy and adaptability in the dynamic stock market.

In the research paper: “Sentiment and Knowledge Based Algorithmic Trading with Deep Reinforcement Learning” by A. Nan et al from the University of Alberta, Canada, they incorporated additional exterior factors that are prone to very frequent changes and often these changes cannot be inferred from the historical trend alone. For this they incorporated the Partially Observable Markov Decision Processes (POMDP). This will take into account activities outside the realm of trading stocks, such as, a trading data center getting destroyed, a scenario that was actual on September 11th.

In our architecture utilizing the Genetic Agent (GA) to select a subset of stocks from a larger pool of stocks based on a predefined objective, is in-line with the strategy based off the client portfolio input. This also ensures that the trading is within regulation requirements.

The DRL models (SAC and PPO) in our project are responsible for making trading decisions based on market conditions. These are well suited models for this environment. The FNN model is used as the underlying architecture for both the actor and critic networks in the SAC and PPO algorithms to predict future stock prices. This combination is well suited to build a successful stock trading system.

Experimental Methods

High Level Diagram Flow:

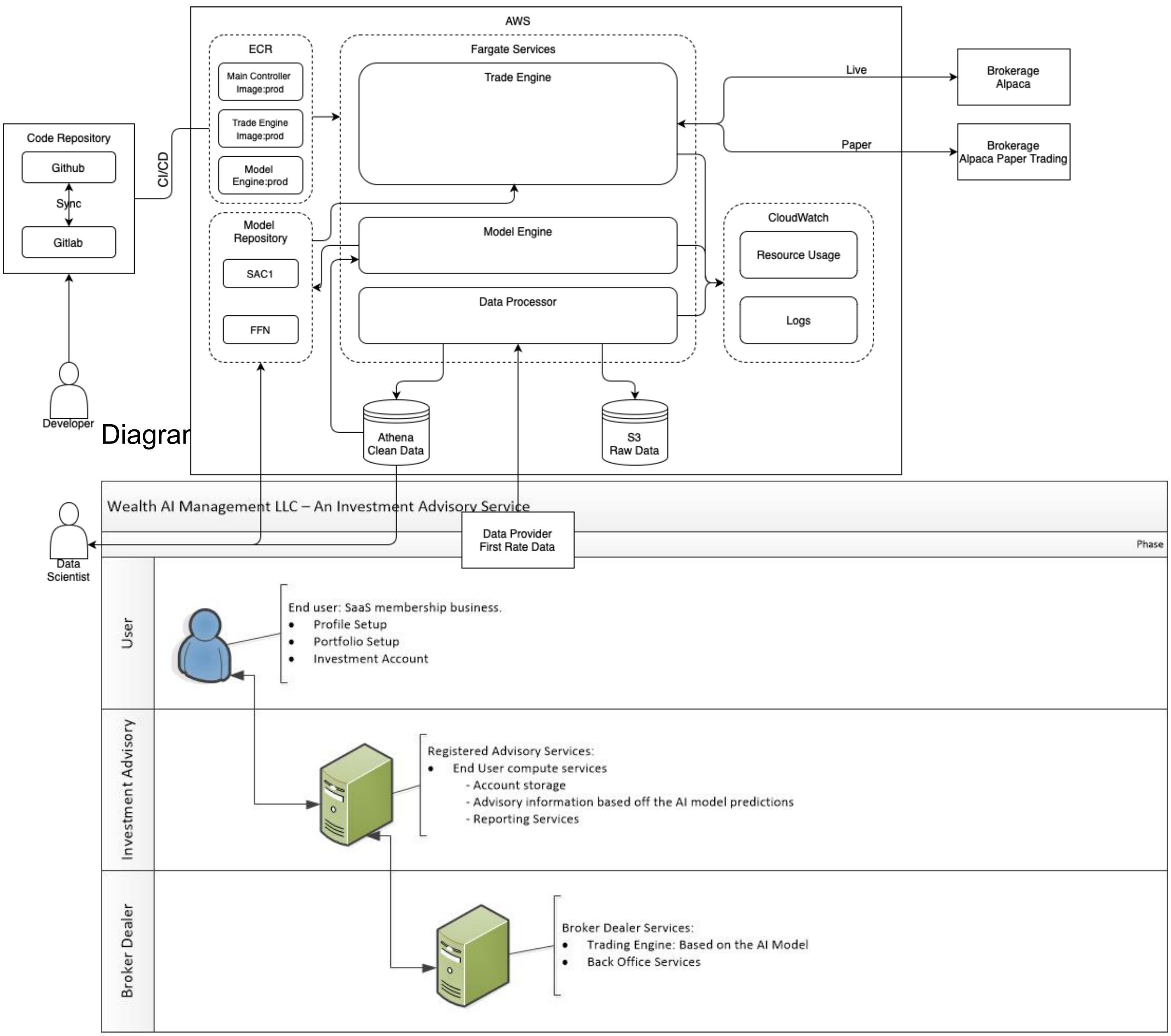

Diagram of Portfolio and Account Management

Portfolio Fields

- Name
- Social Security number or taxpayer identification number
- Address
- Telephone number
- E-mail address
- Date of birth
- Driver's license, passport information, or other government-issued identification
- Employment status and occupation
- Whether you are employed by a brokerage firm
- Annual income
- Other investments
- Financial situation and needs
- Tax status
- Investment experience and objectives
- Investment time horizon
- Liquidity needs and tolerance for risk
- Financial and trading record
- Net worth
- Trading experience
- Financial knowledge

Feature Engineering

From the Portfolio dataset we calculate logics to create a “Risk Factor”. This Risk Factor is taken into consideration in the constraints set upon the trading activity per the client. With a higher Risk Factor, the financial limits and stock volatility is limited accordingly.

Logical layer:

- Date of Birth needs to be calculated and then assigned a value 0-5. Calculate age. 21-29=0, 30-39=1, 40-49=2, 50-59=3, 60-68=4, 69+=5 (0-5)

- Networth: set to 0-10 based on the logic of:

100,000<= 0, 200,000<=1 …..1,000,000>=10

- The add the values of Networth+Trading

experience+Financial knowledge

- Add these two rows above and the other yellow

fields to get the "Liquidity needs and tolerance for risk" value

This is a value to be calculated into the model. The higher the value the lower the risk factor.

Results/Conclusion

This section of your final report should describe how your model performed and provide

evidence for how your work did/did not support your experimental hypothesis or solve your

chosen problem. The reader should be able to answer the following questions:

- How effective was your machine learning model at learning the task? e.g. Did the model

overfit/underfit the training data? What does the training accuracy/loss curve look like for your model?

- If your project is focused on answering a research question, what evidence do you have

to support or disprove your hypothesis?

- If your project is focused on solving a problem, how will your model performance affect the application of your model to that problem?

- Did your model training or performance bring about any unexpected results?

- If you were to continue this project, what would your next steps be? e.g., Would you continue to optimize your model or try other model architectures? Would you change or expand the data used in your model? What steps need to be taken to “productionize” your model

References

Hongyang Yang, Xiao-Yang Liu, Shan Zhong, Anwar Walid, S&P Global, *“Deep Reinforcement Learning for Automated Stock Trading: An Ensemble Strategy”,* November 2020.

OpenAI-SAC:

@misc{pytorch_sac,

author = {Yarats, Denis and Kostrikov, Ilya},

title = {Soft Actor-Critic (SAC) implementation in PyTorch},

year = {2020},

publisher = {GitHub},

journal = {GitHub repository},

howpublished = {\url{https://github.com/denisyarats/pytorch_sac}},

}

Taken from: https://github.com/denisyarats/pytorch_sac

Abhishek Nan, Anandh Perumal, Osmar R Zaiane, *“Sentiment and Knowledge Based Algorithmic Trading with Deep Reinforcement Learning”,* January 26, 2020.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, Sergey Levine, *“Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor”,* Taken from: chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://proceedings.mlr.press/v80/haarnoja18b/haarnoja18b.pdf

Tuomas Haarnoja, Vitchyr Pong, Kristian Hartikainen, Aurick Zhou, Murtaza Dalal, and Sergey LevineSoft Dec. 14 2018, *“Actor Critic—Deep Reinforcement Learning with Real-World Robots”,* Taken from: https://bair.berkeley.edu/blog/2018/12/14/sac/

Wouter van Heeswijk, PhD, *“Proximal Policy Optimization (PPO) Explained”*, November 29, 2022.