Optimizing Stock Actions: Leveraging Markov Decision Process and Linear Regression for Predictive Decision-Making

Tommy Nguyen

Wentworth Institute of Technology

COMP-5700, Classical AI

Frank Kreimendahl

7/28/2023

Abstract

Stock prediction is a highly sought-after research area, with various algorithms aimed at forecasting optimal actions for specific stocks. In this study, I explore the potential of combining Markov Decision Process (MDP) with Linear Regression to predict the most advantageous actions for individual stocks. I utilized historical data gathered from the open-source API yfinance to train and validate my models. Through my research and implementation, I have demonstrated that the integration of MDP and Linear Regression yields promising outcomes in the context of implementing MDP to predict optimal actions. However, it's important to acknowledge that while Linear Regression offers valuable insights, it may fall short in effectively predicting future stock prices. Despite its limitations, this approach enables us to capture stock trends and make informed decisions for optimizing investments in the stock market.

Keywords: Markov Decision Process, Linear Regression, Transition Model, Reward Model, Utility function, weights

Optimizing Stock Actions: Leveraging Markov Decision Process and Linear Regression for Predictive Decision-Making

Stock market prediction has long been a contentious endeavor, stemming from the inherent volatility of stocks. Critics often argue that the unpredictability of the market renders predictions futile, pointing to a notable historical experiment wherein a blindfolded monkey managed to select stocks with superior investment potential compared to seasoned financial experts. Despite this skepticism, investors fervently seek insights into their stock's future performance and the optimal decisions to make. Unfortunately, a reliable means of forecasting the stock market has remained elusive. However, the advent of Artificial Intelligence and the proliferation of sophisticated algorithms offer a glimmer of hope in this domain. Leveraging AI algorithms, such as Linear Regression, enables the acquisition of crucial insights into specific stock trends, facilitating accurate predictions through extensive iterations. Moreover, the introduction of Markov Decision Process (MDP) empowers us to anticipate optimal actions for various market states, ultimately leading to the determination of the most advantageous decisions overall. In this pursuit, my Classical AI final project seeks to demonstrate the potential of merging Linear Regression and MDP in predicting stock values as well as the best actions to take for a stock. The ultimate aim is to equip investors with invaluable guidance, allowing them to make informed and astute decisions for individual stocks.

Related Works

During my research, I delved into various research papers that explored portfolio optimization problems using Markov Decision Processes (MDP) or their derivatives. While these papers often focused on portfolio optimization, I discovered that many of the concepts and

approaches discussed could be adapted to my specific problem of selecting optimal actions for individual stocks.

For instance, in the paper "Using Markov Decision Processes to Solve a Portfolio Allocation Problem" by Daniel Bookstaber, the author investigates the application of expected returns from stocks as rewards for states (Bookstaber, 2005). However, his reward function considers the cost of transitioning (i.e., the cost associated with buying, selling, or holding a stock). This design aligns with his goal of determining optimal decisions for an overarching optimal portfolio, which involves multiple stocks and incorporates the current stock state.

Given my project's timeline, a complete implementation of a portfolio optimization using MDP wasn't feasible. Moreover, the paper also discusses the use of Monte Carlo simulations to calculate expected future returns (Bookstaber, 2005). While these simulations offer accuracy, their resource-intensive nature and the project's one-month timeframe posed challenges. Integrating Monte Carlo simulations might be feasible with more time.

Another valuable aspect I gathered from these papers is the process of discretizing states within the transition model. Specifically, the paper "Single-Asset Portfolio Allocation Using Markov Decision Process – A Case from the Saudi Stock Market" by Khalid Al-Khodhairi et al. introduces the concept of discretizing states into "UP" and "DOWN" to represent stock changes over time (Al-Khodhairi, 2019). This approach provides insights into state movements, enabling informed decisions. Their "stock value change" reward model introduces a simpler way to implement a dynamic reward function, aligning well with my need to work with individual day-based states.

It's important to note that both papers, while presenting potential solutions using various MDP variants, didn't explore the possibility of creating a dynamic transition model, which was

crucial for my application to work with any stock. In fact, most of their transition models seemed to be built manually (i.e. gathering data by hand). While the papers introduced complexities related to MDP implementations for stocks (specifically for portfolio optimization), I needed to focus on developing a dynamic transition model tailored to individual stocks.

Method

Approach/Implementation

My final implementation of the stock market prediction application draws from various approaches discussed earlier. While these approaches offer valuable algorithms, some aspects were not suitable for realizing the real-time, data collection, and analysis application I envisioned.

Many of the algorithms and MDP implementations mentioned rely heavily on historical data and prior analysis to achieve goals (such as constructing Transition based on historical data). Moreover, these implementations are rooted in the concept of "portfolio optimization," encompassing aspects that aren't relevant to my application. Consequently, my final implementation introduces some new features that were not covered previously.

During the initial stages of implementation, my aim was to leverage current and up-to-date data to predict optimal actions for a given stock. However, it became evident that relying solely on "real-time" data—essentially the current stock value—would not suffice for constructing a comprehensive MDP model. Building such a model based solely on the present stock value is not feasible.

To overcome this challenge, I adopted a different approach. Rather than relying solely on historical data or the current stock value, I chose to employ Linear Regression to forecast the future stock value. The equation powering my Linear Regression algorithm is defined as follows:

$$W_0 + W_1 * V_1 + W_2 * V_2$$

Here, W_0, W_1, and W_2 represent weights that are initialized with random values. V_1 and V_2 denote variables representing the current cost and the specified time, respectively. The model is trained using 42 days of historical closed stock data, and it undergoes 200,000 iterations with a learning rate (alpha) of 0.00001.

After the weights are computed through this training process, they are then employed to calculate predictive stock values. The ultimate objective of employing Linear Regression is to enable the retrieval of predicted stock values. Additionally, I aim to calculate the difference between stock values for two consecutive days and store this difference as a state. Each state corresponds to a day following the current day.

This difference is computed by subtracting the current value (actual value) of the stock (denoted as P_0) from the predicted future value of the stock (P_i), where "i" signifies the ith day after the current day. This approach allows for the creation of a dynamic reward function that takes into account the changes in stock value over time.

$$P_1 - P_0, P_2 - P_0, \ldots, P_i - P_0$$

In my approach, the reward function serves as a repository for the calculated differences, while the utility function, by default, also encompasses these differences. The rationale behind storing differences instead of actual stock values lies in our focus on understanding how much the stock is fluctuating. Similar to how investors and financial experts analyze stock prices, considering the change in stock value since the last check is of importance. Consequently, my MDP assesses the extent of stock changes since the current price.

With the reward and utility functions defined, attention turns to the transition model. Since stock prices exhibit continuous behavior and rarely revisit specific values, addressing this continuous nature posed a challenge. To address this, I chose to discretize states within the transition model.

A significant similarity between my implementation and many explored in the "Relevant Research" lies in the notion of discretizing states. Dealing with data that doesn't revisit values makes constructing transition models with probabilities nearly impossible. By discretizing states within the transition model, I effectively circumvent the continuous nature of stock market values. In my case, I've opted to discretize states into three potential categories:

UP, STABLE, and DOWN

In my approach, each state within the model symbolizes the day-to-day change in stock prices. Achieving this required applying equal frequency binning to segment the data into three distinct bins, each corresponding to the UP, STABLE, or DOWN state.

To proceed, I sought insights into the decisions investors make under varying stock conditions (actions taken). However, the absence of publicly available APIs necessitated a creative solution. In order to gather information on buy, hold, and sell actions to compute probabilities, I had to provide my own insights. This process involved envisioning when I would personally buy, hold, or sell based on observed changes. Notably, the application executes these decisions in real-time, making them unprocessed and dynamic. To facilitate this, I established state-action pairs where actions encompass buying, holding, or selling a stock.

You might question whether this approach, which relies on hypothetical actions rather than real investor data, would impact overall accuracy. While unique, my approach mirrors that of many investors (buying on upswings, selling on downturns, and holding during small changes

or stagnation). Subsequent to policy calculation, I exported results to a CSV file. This file includes the stock abbreviation, the current price (at runtime), the probability of buying as the optimal action, the probability of holding, and the probability of selling.

Notably, I opted to export the probabilities associated with each action as the best overall choice. Investors predominantly focus on optimal long-term actions, dismissing redundant actions like selling and repurchasing. Utilizing MDP and Linear Regression empowers us to forecast future stock behaviors while factoring in uncertainty and avoiding obsolete data. This dynamic approach provides us with a more agile means of predicting stock actions, relying on projected future stock values.

While this approach is promising, certain implementation issues are addressed in the "Assessments and Measures" section.

Assessments and Measures

To evaluate the efficacy of my application, I conducted a test involving the prediction of optimal actions for five stocks: Tesla, Walgreens, Amazon, Apple, and MetLife. The application was tasked with forecasting the most suitable course of action 42 days after its initial execution. The table presented below outlines the projected optimal actions provided by the application, alongside the actual resulting prices observed one week after program execution:

Companies	**Tesla**	**Walgreens**	**Amazon**	**Apple**	**MetLife**
Estimate:	**Price:** $266.43 **BUY**: 100%	**Price:** $29.74 **BUY**: 28.99% **SELL**: 71%	**Price:** $132.43 **BUY:** 100%	**Price:** $196.06 **BUY:** 100%	**Price:** $62.47 **SELL:** 100%
Result:	**Price:** $260.43	**Price:** $29.46	**Price:** $142.99	**Price:** $185.31	**Price:** $62.96

In terms of short-term outcomes, the application encounters challenges when confronted with the stock market's inherent volatility. This outcome is anticipated since the Linear Regression algorithm I employed attempts to fit a straight line to historical stock data. This approach, however, falls short in accurately predicting day-to-day or even week-to-week fluctuations—a limitation attributable to the substantial and rapid changes that occur in stock values during these periods. Stock market dynamics rarely align with a simple linear trajectory.

However, the application demonstrates potential in long-term predictions. It's adept at capturing overarching data trends. It's worth noting that conclusive determination of its long-term effectiveness necessitates further testing—a comprehensive evaluation extending beyond the 42-day horizon was hindered by time constraints.

An alternative approach could involve shortening the evaluation window to slightly over a week, although this course of action would have required more time for comprehensive testing. To summarize, the findings underscore the application's short-term prediction limitations—stemming from the challenge of accommodating volatile price fluctuations with a linear model. Consequently, the proposed solution's effectiveness is compromised in the short term. Evaluating its efficacy in the long term remains inconclusive but warrants exploration through more comprehensive testing.

Conclusion

Despite the currently modest accuracy, I maintain confidence in the potential of the Markov Decision Process (MDP) approach for predicting optimal stock actions. However, it's crucial to acknowledge that an MDP implementation demands more nuanced data than mere stock values. Given extended time for development, my proposed enhancements include seeking an API to provide richer investor action data, potentially altering the Linear Regression

algorithm to fit a sinusoidal function to stocks, or even exploring alternative algorithms altogether (ensemble learning, Monte Carlo simulations, etc.). The key would be to conduct further rigorous testing to assess the application's performance over the long term.

References

Al-Khodhairi, K., Baz, A. B., & AlDurgam, M. (n.d.). *Single-Asset Portfolio Allocation Using Markov Decision Process – A Case from the Saudi Stock Market*. Proceedings of the International Conference on Industrial Engineering and Operations Management. http://www.ieomsociety.org/gcc2019/papers/128.pdf

Bookstaber, D. (n.d.). Using Markov Decision Processes to Solve a Portfolio Allocation Problem. https://cs.brown.edu/research/pubs/theses/ugrad/2005/dbooksta.pdf