Reinforcement Learning for Stock Transactions

Ziyi (Queena) Zhou, Nicholas Stern, Julien Laasri

Abstract

Our goal for this project is to train an algorithm to confidently determine whether or not to buy or sell a stock based on historical performance of this stock. We plan to achieve this goal in two ways: reinforcement learning and machine learning. We defined our own Markov Decision Process (MDP) so that we could use the format of free, real-world data to train the models.

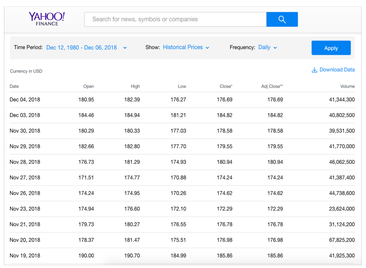
Background & Motivation

Stock markets have been well studied by researchers using different analytical tools. In this project, we are most curious to find out whether a single stock exhibits an implicit pattern, so that we could develop a specific trading strategy for that stock. Plenty of money could be made if we can successfully detect stock trends.

We chose to apply Reinforcement Learning techniques to explore the stock market, given that Reinforcement Learning has the advantages of balancing exploitation and exploration. In particular, we can encode a small possibility that our agent will deviate from the greedy algorithm, finding non-local optima from trying new things.

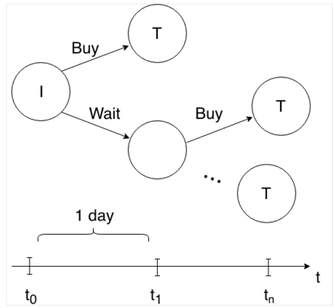
Data

We capitalized on Yahoo Finance's data, a large database of stock prices for a wide array of companies stretching back for many years. We chose to narrow our focus on only the tech companies for our study for two reasons. First, stock prices in the technology sector tend to vary on a daily basis, giving us a nice framework to set our state space based on prices going up or down in one day. Second, the broader our scope, the greater the chance that trends would get washed out across different sectors.



Date	Open	High	Low	Close*	Adj Close**	Volume
Dec 04, 2018	180.95	182.39	176.27	176.69	176.69	41,344,300
Dec 03, 2018	184.46	184.94	181.21	184.82	184.82	40,802,500
Nov 30, 2018	180.29	180.33	177.03	178.58	178.58	39,531,500
Nov 29, 2018	182.66	182.80	177.70	179.55	179.55	41,770,000
Nov 28, 2018	176.73	181.29	174.93	180.94	180.94	46,062,500
Nov 27, 2018	171.51	174.77	170.88	174.24	174.24	41,387,400
Nov 26, 2018	174.24	174.95	170.26	174.62	174.62	44,738,600
Nov 23, 2018	174.94	176.60	172.10	172.29	172.29	23,624,000
Nov 21, 2018	179.73	180.27	176.55	176.78	176.78	31,124,200
Nov 20, 2018	178.37	181.47	175.51	176.98	176.98	67,825,200
Nov 19, 2018	190.00	190.70	184.99	185.86	185.86	41,925,300

Implementation of RL Agent



Given data on daily stock prices, we chose to set up a basic agent that could learn to capitalize on daily trends. This agent had two actions: buy and wait, and was rewarded for purchasing before an increase in price the next day. We used ε-greedy Q-learning to develop a policy from initial state "I" to terminal states "T."

More formally, we discretized the state space by aggregating price fluctuations into two cardinal directions (up and down), based on the closing price on each day. A state was then defined to be the up/down trends of the last "h" days, where "h" is a parameter we denoted as the history window.

$$\mathbf{S:}\ \{\uparrow,\uparrow,\downarrow\}$$

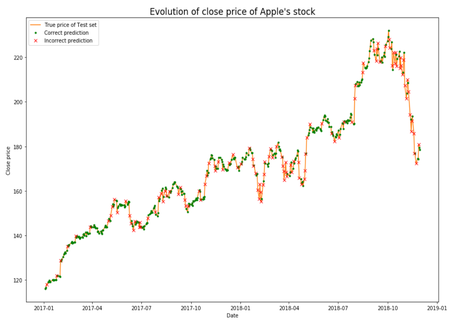
$$\boldsymbol{R:}\begin{cases} r & \text{if buy then } \uparrow \\ -r & \text{if buy then } \downarrow \\ -\lambda & \text{if wait} \end{cases}$$

$$\mathbf{A:}\ \{\ \text{'buy', 'wait'}\ \}$$

Price Prediction

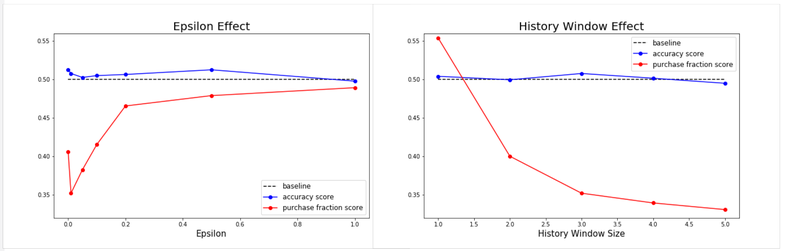
We aim to use price prediction to enhance our RL agent. Namely, our agent could make more informed decisions if the prediction is included in its state space. However, gradient boosted decision trees and logistic regression models struggled to predict next-day trends with more than 50% accuracy.

Instead of predicting trends, we defaulted to directly predicting next-day prices from the current day's price with linear regression. With a relative error tolerance of 2% or less, our model guessed the correct price 75% of the time on the test set.



Intuitively, the model did well to predict continuing trends, but failed to predict turning points. We could improve on this by including more information in the features, such as market history or seasonal info. Note: price prediction has yet to be integrated with the agent.

Preliminary Results



We evaluated the performance of our agent using two metrics:

1. **Accuracy Score** - taking the ratio of purchases that resulted in a net gain the next day.
2. **Purchase Fraction** – the number of purchases vs. the total number of decisions made by the agent.

The two figures above show the values of these metrics while varying the history window size and the randomness of the agent's actions. The trends show intuitive relationships with respect to the purchase fraction, but the results indicate that our agent does not perform much better than random actions. We ascertained this is likely due to the random nature of day-to-day stock movements.

Future Plans

Moving forward, we plan to do the following:

- Transition to Approximate Q-learning and Deep Reinforcement Learning to allow ourselves to dramatically increase the state space (e.g. include specific prices of previous days).
- Use features defined by finance specialists to better inform our agent (i.e. momentum, VWAP, RSI).
- Redefine the MDP to allow for a description of reward that incorporates price and regret. Perhaps an agent is told to buy a stock within a certain timeframe. As we know the minimum price of the stock in this timeframe, we can reward the agent according to when it decides to buy.
- Improve the price prediction model to turn it into valuable information for our agent, possibly invoking the use of ANN's.

References

1. Nevmyvaka, Yuriy and Feng, Yi and Kearns, Michael, *Reinforcement Learning for Optimized Trade Execution*, 2006.
2. Jae Won Lee. *Stock Price Prediction Using Reinforcement Learning,* 2001.
3. "Yahoo Finance - Business Finance, Stock Market, Quotes, News." *Yahoo! Finance*, Yahoo!