Department of Robotics and Mechatronics Engineering

University of Dhaka

Project Report

Course: RME 3211 (Intelligent Systems and Robotics Lab)

Name of the project: Financial Data Analysis with Approximate Q Learning

Group no.: 06

Group members:

1) Abdul Monaf Chowdhury (Roll: FH-092-001)

Prepared by: Abdul Monaf Chowdhury (Roll: FH-092-001)

Project performed from: 8th September 2022

Project performed to: 14th November 2022

Date of submission: 15th November 2022

Submitted to:

Dr. Sejuti Rahman,

Assistant Professor & Chairperson,

RME, University of Dhaka

Abstract

Stock trading strategies play a critical role in investment. However, it is challenging to design a profitable strategy in a complex and dynamic stock market. This project presents an algorithmic trading system based on reinforcement learning. We formulate the trading problem as a Markov Decision Process (MDP). The formulated MDP is solved using Approximate Q-Learning. To improve the performance of Approximate Q-Learning, we augment MDP states with an estimate of current market/asset trend information as technical indicators. Our proposed algorithm is able to achieve a high return on investment when we invest ৳10,000.

1. Introduction:

An automatic program that generates constant profit from the financial market is lucrative for every market practitioner. It is applied to optimise capital allocation and maximise investment performance, such as expected return. Return maximisation can be based on the estimates of potential return and risk. However, it is challenging for analysts to consider all relevant factors in a complex and dynamic stock market. An important factor affecting trading decisions is the ability to predict stock market movement. The prediction of stock market movement is considered to be a challenging task of financial time series prediction due to the complexity of the stock market with its noisy and volatile environment, considering the strong connection to numerous stochastic factors such as political events, newspapers as well as quarterly and annual reports.

There are existing works on this topic. An approach for stock trading can be to model as a Markov Decision Process (MDP) and use dynamic programming to derive the optimal strategy. However, the scalability of this model is limited due to the large state spaces when dealing with the stock market. Machine learning algorithms like reinforcement learning can be used to address this issue.

In this project, we want to explore the possibility of building a machine learning agent that tries to learn an optimal trading policy/strategy using a machine learning technique like reinforcement learning. The problem we are trying to solve in this project can be summarized as “Train an ML Agent to learn an optimal trading strategy based on historical data and stock market indicators in order to maximize the generated profits.”

In this report, we lay out the project about stock market trading which was done using approximate Q learning of Reinforcement Learning. It finds the optimal trading strategy in a complex and dynamic stock market. Approximate Q learning can adjust to different market situations and maximize return subject to risk constraints.

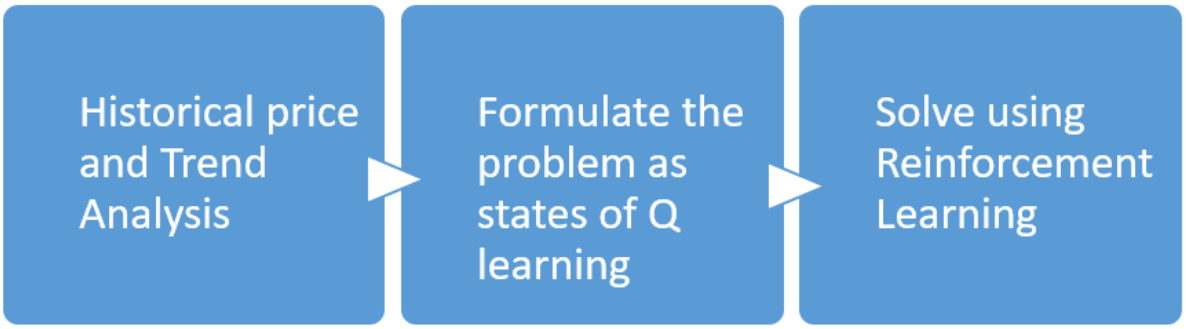

Figure 1: Proposed Approach

To explain the figure displayed earlier, the project would involve the following sub-steps:

1. Generate market indicators using the fundamental data from the stocks.
2. Formulate the trading problem as a Q-learning problem with functional approximations. The states of the Q learning not only involves historical price, number of stocks, and cash in hand but also technical indicators.
3. Finally, implement the best option generated by the Q value function.

This Q learning value approach uses stochastic gradient descent to train the data rather than batch gradient descent making it faster. The approach followed in this report resulted in profits in multiple test runs and has shown promise that it can be applied in an active trading field with some adjusted modifications necessary.

The remainder of this paper is organized as follows. Section 2 introduces related works. This section describes the works done using a similar process or some advanced optimization algorithms. Section 3 provides a systematic methodological description of the stock trading problem. In Section 4, we set up the stock trading environment using the earlier-mentioned methods. This section describes the stock data preprocessing and the experimental setup. Section 5 deals with trading the agent based on Approximate Q Learning. In Section 6, provides a summary and comparison of our results. Section 7, further identifies the current limitations of our model. Finally, Section 8, summarizes and concludes this paper and lays the foundation for future improvement and modifications.

2. Literature Review:

Recent applications of reinforcement learning in financial markets consider discrete or continuous state and action spaces. Various different techniques have been implemented in literature to train ML agents to do automated trading using machine learning techniques. For instance, [Pierpaolo G. Necchi, 2016][1], [David W. Lu, 2017][2], [Xin Du, Jinjian Zhai, Koupin Lv, 2009][3], [Jae Won Lee, Jangmin O][4] all describe different machine learning techniques like approximate q learning, deep q-learning, recurrent reinforcement learning, etc to perform algorithmic trading. [James Cumming, 2015][5] also wrote a book on the use of different reinforcement learning techniques within the Algorithmic Trading Domain. The major strength of these researches is that they are trying to investigate the best possible learning algorithm so that automated trading can be performed with minimum human intervention.

Almost all trading firms use stock market technical indicators in their trading strategy. In some cases, all algorithmic Quant trading firms develop their trading algorithms based on the indicators and rely heavily on them. In this project, we follow an approach of combining both implementations where an ML agent not only just learns an optimal trading strategy based on historical prices but also on additional information based on technical indicators.

3. Problem Description:

We model stock trading as a Markov Decision Process (MDP) which is then further incorporated into the approximate Q value approach and formulate our trading objective as a maximization of expected return.

To model the stochastic nature of the dynamic stock market, the following idioms are used:

- State s: An ndArray vector that incorporates the features of the stock data, the number of stocks that are dealt with and the remaining cash in hand. The features included the technical indicators, price as well as volume.
- Action a: A vector for actions over the number of stocks. The allowed actions on each stock include selling, buying, or holding, which result in decreasing, increasing, and no change of the stock shares. The value available for action is [1,0,2]. Here, 0 denotes sell, 1 denotes hold and 2 denotes buy.
- Reward r(s, a, s$^{'}$): The direct reward of taking action A at state S and arriving at the new state s$^{'}$. The rewards are discounted by a factor of gamma.

- Policy $\boldsymbol{\Pi}(s)$: The trading strategy at state S, which is the probability distribution of actions at state s.
- Q-value $Q_{\pi}(s, a)$: The expected reward of taking action A at state s following policy $\boldsymbol{\Pi}$.

The model generally works as follows. $Q_{\pi}(s, a)$ is updated through interacting with the stock market environment. The optimal strategy is given by the Bellman Equation, such that the expected reward of taking action a_t at state s_t is the expectation of the summation of the direct reward $r(s_t, a_t, s_{t+1})$ and the future reward in the next state s_{t+1}. The future rewards are discounted by a factor of $0 < \boldsymbol{\lambda} < 1$ for convergence purposes. The goal is to design a trading strategy that maximizes the positive cumulative change of the portfolio value $r(s_t, a_t, s_{t+1})$ in the dynamic environment, and we employ the approximate Q learning of the reinforcement learning method to solve this problem.

4. Stock Market Environment:

Before training a reinforcement learning trading agent, we carefully build the environment to simulate real-world trading which allows the agent to perform interaction and learning. In practical trading, various information needs to be taken into account, for example, the historical stock prices, current holding shares, technical indicators, etc. Our trading agent needs to obtain such information through the environment, and take actions defined in the previous section.

The environment deals with 3 stocks. They are Grameenphone Telecommunications company **(GP),** Beximco Pharmaceuticals Limited (**BX)** and finally Square group of industries (**SQ).**

The start of the dataset: 3rd January 2010
End of the dataset: 8th September 2022
Total no of business days on the dataset: 2964
The Dataset is split between train and test data as 75-25.

Using the stock price history technical indicators were created. These technical indicators were then further indoctrinated as a feature into the final train and test data. The technical indicators are:

A) **Moving Average Convergence Divergence (MACD)[6]:** MACD is a trend-following momentum indicator that shows the relationship between two exponential moving averages (EMAs) of a stock's price. The MACD line is calculated by subtracting the 26-period EMA from the 12-period EMA. The result of that calculation is the MACD line.

B) **Relative Strength Index (RSI)**[7]**:** RSI is a momentum indicator used in technical analysis. RSI measures the speed and magnitude of a security's recent price changes to evaluate overvalued or undervalued conditions in the price of that security. If the price moves around the support line, it indicates the stock is oversold, and we can perform the buy action. If the price moves around the resistance, it indicates the stock is overbought, and we can perform the selling action[8].

C) **Commodity Chanel Index (CCI):** CCI is calculated using high, low and close prices. CCI compares the current price to the average price over a time window to indicate a buying or selling action[9]. It helps to determine when an investment product is reaching the condition of being overbought or oversold.

D) **Average Directional Index (ADX):** ADX is calculated using high, low and close prices. ADX identifies trend strength by quantifying the amount of price movement[10]. Trading in the direction of a strong trend reduces risk and increases profit potential. In many cases, it is the ultimate trend indicator.

All these indicators are inculated in the final stock data. They are used as features for the data to train on. The final stock data consists of a total of **18** features each stock containing six of its own. Six of the features are the price of the stock, the volume traded on that day, MACD, RSI, CCI, and ADX of that particular stock on that day.

Finally, in the state space, a state contains 22 elements in it. The first 18 elements are the features extracted from the final stock data. The following 3 are the number of stocks of BX, GP, and SQ owned at each step. Lastly, the final element is the remaining cash in hand.
The action space contains a vector where the dimension is $(\text{number of actions})^{(\text{number of stocks})}$. As the problem contains only 3 actions, i,e, buy, sell and hold, and deals with three stocks the action space has 27 dimensions. It deals with all sorts of combinations of actions available at every trade.

5. Trading Agent Based on Approximate Q Learning:

We use an approximate Q-learning algorithm to implement our trading agent. All three of the stocks contain different ranges of values. Thereby, to fit the values more perfectly a scaling operation is done. The data is trained on a stochastic gradient descent to make it faster to converge. In SGD, it uses only a single sample to perform each iteration. The path taken by the algorithm to reach the minima is usually noisy but reaches the minima with a significantly short training time.

In the approximate Q learning process, the agent follows the epsilon-greedy method for exploration and exploitation. Initially, the epsilon is set at one, indicating it explores as much as it can. However, the epsilon is multiplied by epsilon decay of 0.995 reducing its original value at each episode. This continues till it reaches a minimum epsilon value of 0.01.
The Q-value rewards are discounted by a gamma factor of 0.95.

Finally using the stock data with all the features training and testing are done. The training requires more than a thousand episodes to deal with the optimal exploration and exploitation and thus learning the optimal policy. Eventually, we pick the trading agent that can maximize the returns adjusted to the increased risk.

6. Results:

The training data consisted of 2222 samples out of 2963 samples. Thereby, making the number of samples for test data 741. Initially, training was done.
Portfolio value after the training on the **train data**:

```
REWARDS:

Maximum:              214322.84000003972
Minimum:              2389.120000000052
Mean:                 44350.739053334
Standard Deviation:   39104.39151739811
Return of Investment: 343.5074
Average Return on All Episodes:15.466339060483566
```

Figure 2: Portfolio Values

The mean of the portfolio value is 44,350.74 bdt with a return on investment of 343.5074%. This is much higher than traditional returns. This happened due to the volatility of the exploration vs exploitation of the epsilon-greedy approach. This can be seen on the next page at the reward per episode graph. Initially, there was much volatility which returned high rewards, as the exploitation got up the rewards stabilized. But due to the high returns on the first few hundred of episodes, the return on investment became this large.

Now the reward graph on train data per episode

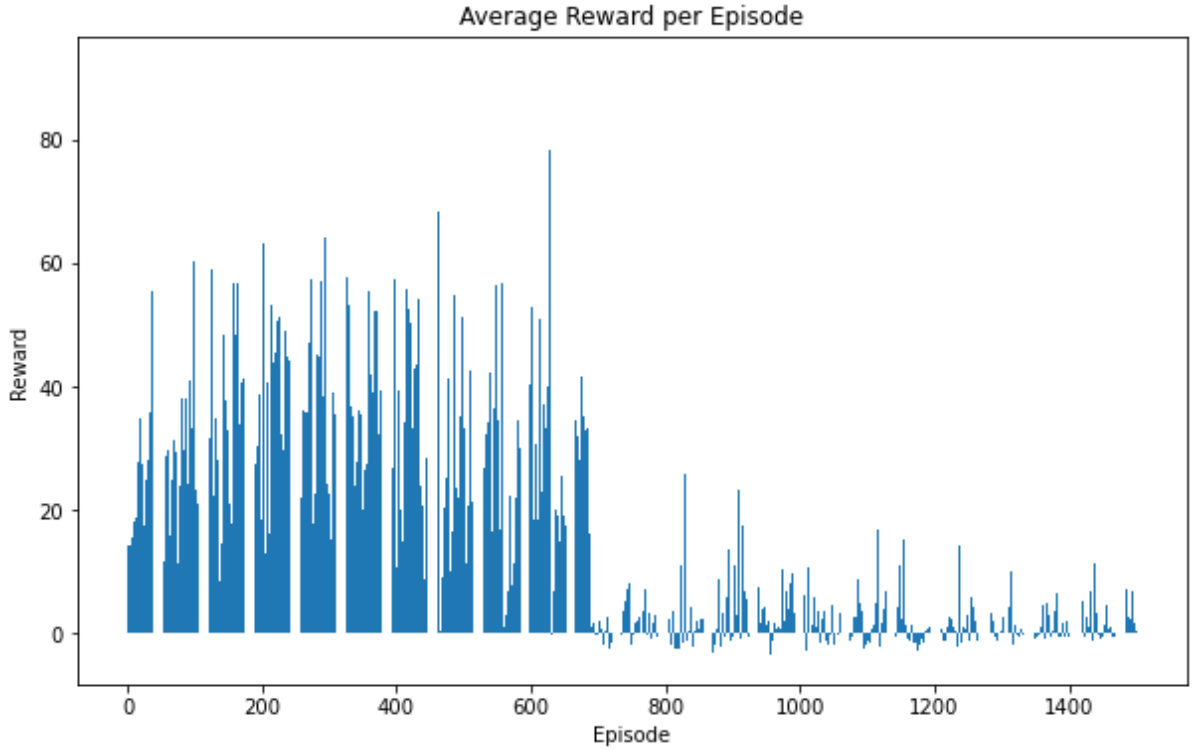

Figure 3: Reward vs Episode graph

After the training, backtesting was done using the test data.
Portfolio value using the **test data**:

```
REWARDS:

Maximum:              15087.460000000492
Minimum:              8585.700000000234
Mean:                 11175.538899999496
Standard Deviation:  1069.1225367514319
Return of Investment: 11.7554
Average Return on All Episodes:1.5885660810803979
```

Figure 4: Portfolio Values

The mean portfolio value is 11,175.54 bdt with a return on investment of 11.7554 %. This return on investment is slightly higher than the traditional market return index like the DSE30 index in this time period.

Now the reward graph on **test data** per episode:

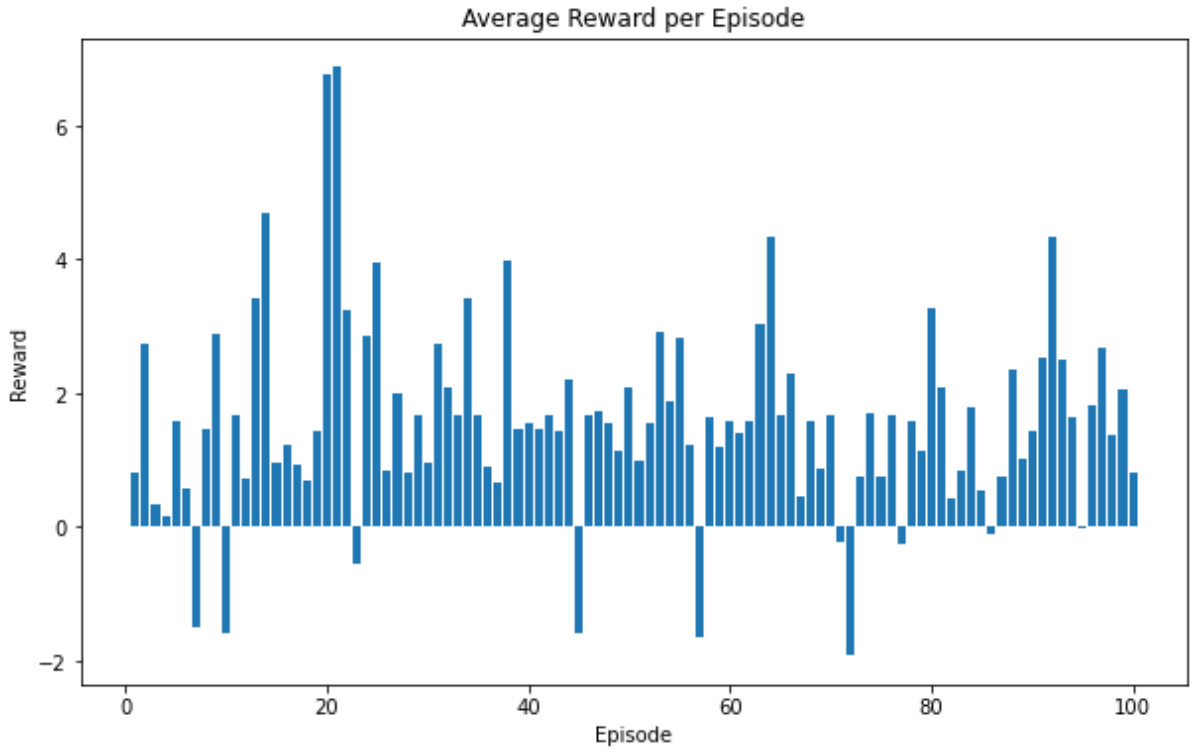

Figure 5: Reward vs Episode graph

Now according to the testing results, the best episode is then used to generate the buy/sell graph for the three companies. The red dot signals buy action and the blue dot signals sell action.
They are shown below:

Grameenphone Telecommunications Ltd (GP):

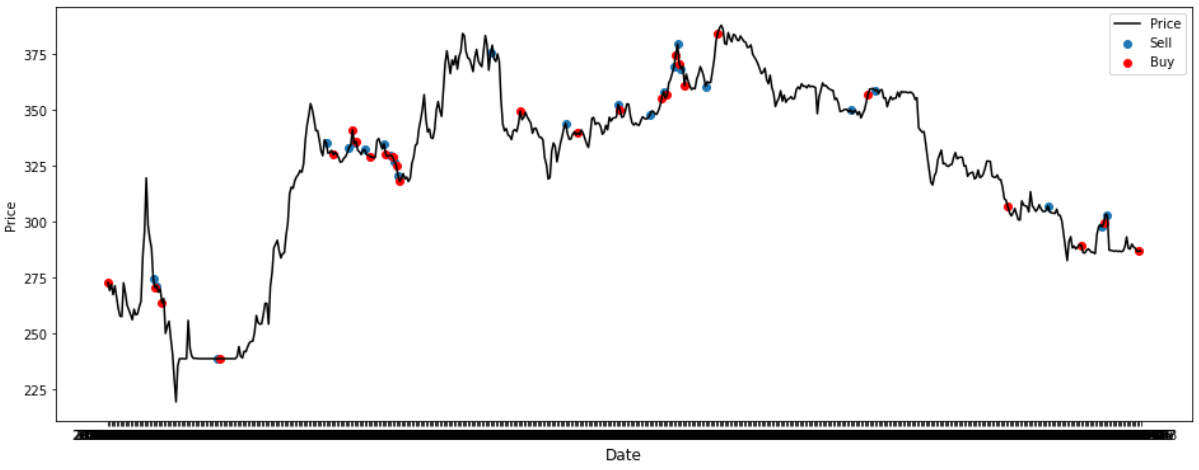

Figure 6: Grameen Phone

Beximco Pharmaceuticals Limited (BX):

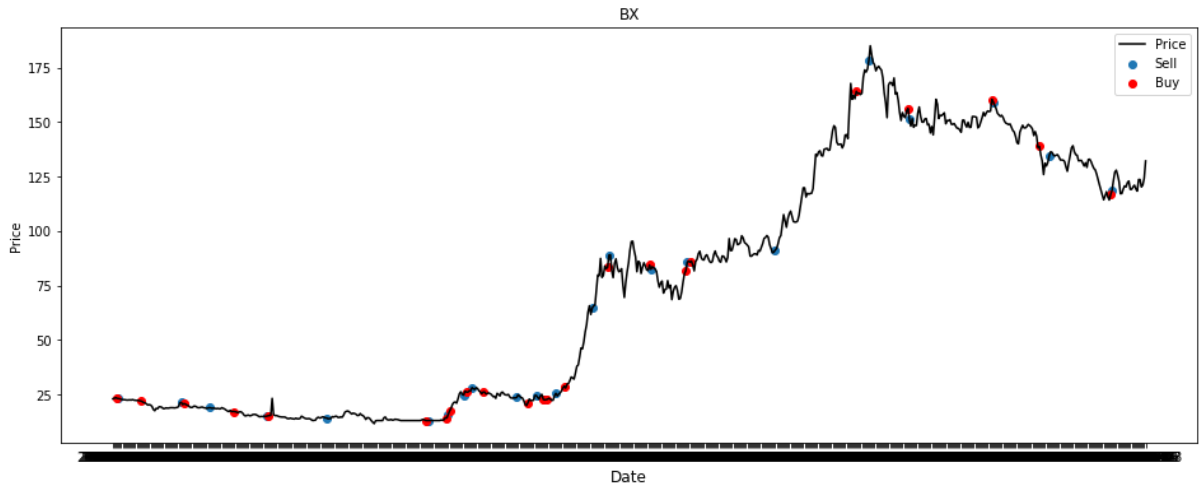

Figure 7: Beximco Pharmaceuticals Limited

Square group of industries (SQ):

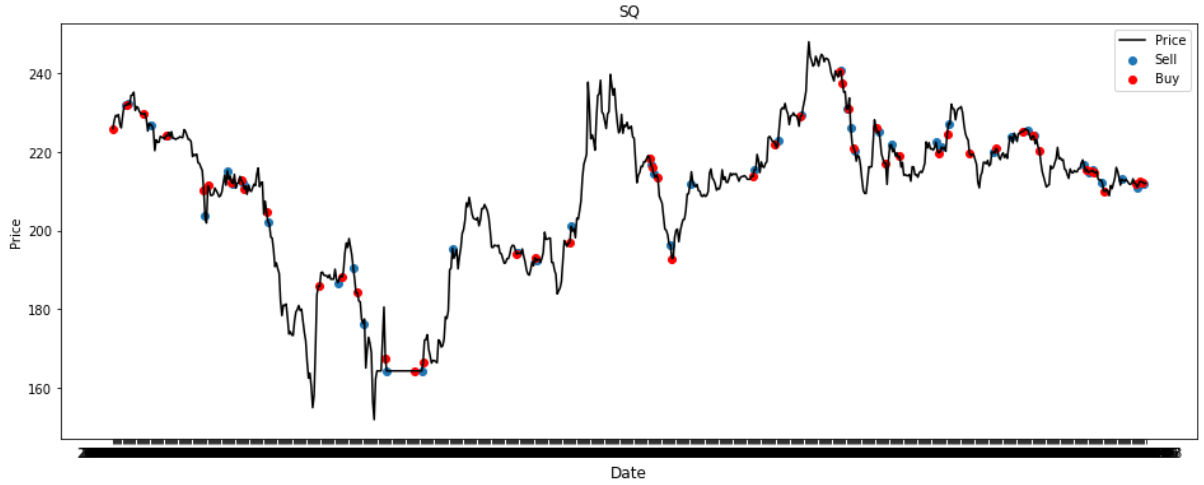

Figure 8: Square Group of Industries

Following these buy/sell graphs will result in the optimal strategy and generate a high return on investment.

7. Limitations and Further Work:

Just like every other project, this one has some of its own limitations. There was a lack of available closing price data for the stocks. DSE doesn't provide historical data with much accuracy and explanations. Thereby, the third-party website investing.com had to be used to get stock data which didn't have closing prices. Some of the technical indicators required the usage of closing price, however, this had to be resolved only with the 'price' of the stock.

Market conditions and sentiments are not analyzed and used in this project. They could've further developed it and made it more accurate and less susceptible to risks. The trading platform is still volatile and subject to high risk which will make conventional investors less prone to trading.

There are a few advancements already available in this type of stock trading project. One of which is very common among researchers is the DQN algorithm, which combines Q-learning with a deep neural network. This algorithm is much more sophisticated and less prone to volatility than approximate Q learning. Further ensemble strategies can be used to sharpen up the trading algorithms and make the best use of the available systems.

8. Conclusion:

In this report, we implemented a reinforcement learning algorithm particularly Q-Learning with functional approximations, to train an ML agent to develop an optimal strategy to perform automated trading. The ML agent was successful in learning the optimal strategy. In order to adjust to different market situations, we used technical indicators. The results showed lucrative revenues along with high volatility.

For future work, it will be interesting to explore more sophisticated models, solve empirical challenges, and deal with large-scale data. We can also explore more features for the state space such as adding advanced transaction cost and liquidity models, and natural language processing analysis of financial market news to our observations. We are interested in directly using the Sharpe ratio as the reward function, but the agents need to observe a lot more historical data, and the state space will increase exponentially.

Deep Q Learning and Deep Reinforcement learning methods have recently been applied to many different problems and have achieved very good performance. Based on the initial results from this report, we feel it could be a good idea to try to implement Deep-Reinforcement learning methods for this problem.

9. Reference:

1. Pierpaolo G. Necchi, Reinforcement Learning For Automated Trading, 2016
2. David W. Lu, Agent Inspired Trading Using Recurrent Reinforcement Learning and LSTM Neural Networks, July 2017.
3. Xin Du, Jinjian Zhai, Koupin Lv, Algorithm Trading using Q - Learning and Recurrent Reinforcement Learning, Dec 2009
4. Jae Won Lee, Jangmin O, A Multiagent Approach to Q-Learning for Daily Stock Trading, Nov 2007
5. James Cumming, An Investigation into the Use of Reinforcement Learning Techniques within the Algorithmic Trading Domain, June 2018
6. https://www.investopedia.com/terms/m/macd.asp [**MACD]**
7. https://www.investopedia.com/terms/r/rsi.asp [**RSI]**
8. Terence Chong, Wing-Kam Ng, and Venus Liew, “Revisiting the performance of macd and rsi oscillators,” Journal of Risk and Financial Management, vol. 7, pp. 1–12, 03 2014.
9. Mansoor Maitah, Petr Procha´zka, Michal C˘ erma´k, and Karel S˘ re´dl, “Commodity channel index: evaluation of trading rule of agricultural commodities,” International Journal of Economics and Financial Issues, vol. 6, pp. 176–178, 03 2016.
10. Ikhlaas Gurrib, “Performance of the average directional index as a market timing tool for the most actively traded USD based currency pairs,” Banks and Bank Systems, vol. 13, pp. 58–70, 08 2018.