Stock prediction using Reinforcement learning

Group Number: 10
Members: Hemant Gaur 12040650, Hrishik Kanade 12040670, Arjit Bhamu 12040300, Chinmay Sahoo 12040480
Subject: Artificial Intelligence
Course code: DS 251

Abstract:
Reinforcement learning for stock prediction is an intriguing and demanding field that has garnered significant research interest. Machine learning that learns from its own actions and rewards—without labeled data or supervision—is known as reinforcement learning. When the stock market is modeled as a dynamic environment, it can be used to predict stock prices. An agent, such as an investor, can then take actions (buy, sell, or hold) based on the current state (the stock price and other indicators) and benefit (profit or loss) from the result. Nevertheless, there is a problem with Q-learning, one of the most widely used reinforcement learning algorithms: it has a propensity to overestimate the action values, which can result in less-than-ideal policies and subpar performance. In order to address this issue, we suggest using double Q-learning, a Q-learning variation that separates the evaluation and selection of actions by using two value functions as opposed to one. As a result, the algorithm's accuracy and stability are increased while the overestimation bias is decreased. To forecast the stock price of Google, one of the most significant and valuable companies in the world, we employ double Q-learning. To train our model, we take historical Google stock price data along with other pertinent features like volume, open, high, low, and close prices. We demonstrate that double Q-learning can yield higher returns and lower risks than Q-learning and deep Q-learning by comparing their respective performances. We also talked about the difficulties.

Introduction:
Predicting a stock's future price movements using historical and current data is known as stock prediction. Given that it can aid investors in improving their decision-making and portfolio optimization, it is a significant and applicable problem. But given that the stock market is impacted by a wide range of variables, including political developments, market sentiment, economic conditions, and human behavior, stock prediction is also a challenging and complex problem. In addition, the stock market is extremely stochastic and dynamic, which implies that it is erratic and ever-changing.

Numerous approaches and strategies, including machine learning, artificial intelligence, statistical analysis, technical analysis, and fundamental analysis, have been developed to address this issue. The most well-liked and promising methods among them are artificial intelligence and machine learning because of their capacity to learn from data and uncover hidden patterns and relationships. Three categories apply to machine learning and artificial intelligence: reinforcement learning, unsupervised learning, and supervised learning.

A kind of machine learning called supervised learning makes use of labeled data, like past stock prices and the labels that go along with them (up or down). Regression, classification, and neural networks are just a few of the models that can be trained with it to forecast future trends or stock prices. Supervised learning does, however, have certain drawbacks, including the requirement for sizable and trustworthy datasets, the possibility of overfitting or underfitting, and the inability to adjust to shifting market conditions.

Machine learning that uses unlabeled data, like past stock prices and other features, is known as unsupervised learning. It can be used to find the structure and distribution of the data by carrying out operations like clustering, dimensionality reduction, or anomaly detection. Unsupervised learning does, however, have certain drawbacks, including the incapacity to generate useful recommendations, the difficulty of interpreting the outcomes, and sensitivity to noise and outliers.

Machine learning that learns from its own actions and rewards—without labeled data or supervision—is known as reinforcement learning. Modeling the stock market as a dynamic environment in which an agent (the investor) can act (buy, sell, or hold) in response to the current state (the stock price and other indicators) and reap the rewards (profit or loss) of the result is one way to apply this approach. Some of the drawbacks of supervised and unsupervised learning can be addressed by reinforcement learning, including the requirement for sizable and trustworthy datasets, the possibility of overfitting or underfitting, and the inability to adjust to shifting market conditions. Additionally, actionable recommendations, like the best trading strategy or policy, can be obtained through reinforcement learning.

Methodology:
To develop a robust trading model that learns automatically on data and performs accordingly, we have used reinforcement learning. Reinforcement learning is a machine learning technique used to take actions in a particular situation to maximize the reward. It is used when we need the best possible behavior or path. It enables AI-based systems to take actions in a dynamic environment through trial and error methods in order to maximize the collective rewards based on the feedback generated for respective actions[1]. Reinforcement learning closely resembles how humans learn by error and trial where agents make a decision and learn from the consequence.
There are various reinforcement techniques including Monte Carlo, Q-learning, etc. but in this we are using Q-Learning and its variant Deep Q-learning.
Reinforcement learning implementations can be classified as:

1. **Policy-based:** This approach aims to maximize the reward by employing deterministic policies, strategies, and techniques.
2. **Value-based:** this implementation intends to optimize the arbitrary value function involved in learning.
3. **Model-based:** The model-based approach enables the creation of a virtual setting for a specific environment.

In this project we are using Q-learning which is a model-free and policy free model where the model learns by random actions(greedy approach). Here Q' refers to the quality of activities that maximize the collective rewards generated through the algorithmic process.
Q Learning uses a reward matrix, in order to store the earned rewards. Lets say, for reward 5, a reward matrix is constructed that assigns a value at position 50 to denote reward 5. These values are updated using methods such as policy iteration and value iteration.
Policy iteration means policy improvement or refinement through actions that fine tune the value function while in a value iteration, the values of the function are updated. Mathematically, Q-learning is represented by the formula:

$$Q(s,a) = (1-\alpha).Q(s,a) + \alpha.(R + \gamma.\max(Q(S2,a))$$

Where,
Q(s,a) = future value, alpha = learning rate, S2 = next state.
gamma = discount factor, R = reward,

The second reinforcement learning method used is Deep Q-Network. Unlike Q-learning, deep Q-network uses neural networks. Q-learning algorithms are inefficient in predicting and updating the state values which they are unaware of, generally unknown states. When there are billions of unknown states and hundreds of actions, it is very difficult to use a matrix with this size work with hence matrix operations tabular methods as in Q-learning
In DQN, 2D arrays are replaced by deep neural networks for efficient calculation of state values and state transitions values, thereby speeding up the learning aspect of RL[2].
Deep neural networks are very good non linear approximators hence DNNs are used to approximate the Q-functions in DQN algorithm. DQN uses Bellman equation which is:

$$Q(s,a; \Theta) = r + \gamma \max Q(s', a' ; \Theta')$$

Where s' is the next state, a' being action, and theta is the parameterised weight vector.

The following is the equation of loss function for the deep Q-network:

$$L(\theta) = \mathbb{E}[(r + \gamma \max_{a'} Q(s', a'; \theta') - Q(s, a; \theta))^2]$$

This function is to be minimized by the gradient descent.

Experimentation:
We have used the Asian paints historic data to train and test our model.
We have used starting money as 1,00,000 and our agents are trying to maximize profit. Here there are two actions possible: Buy and Sell.
The agent is trained on the historical stock data of Asian Paints and then our agent is evaluated on the profit generated on the initial sum of 1,00,000.
At last the final profit after all iterations and the final profit is also plotted on a graph.

Window size used:

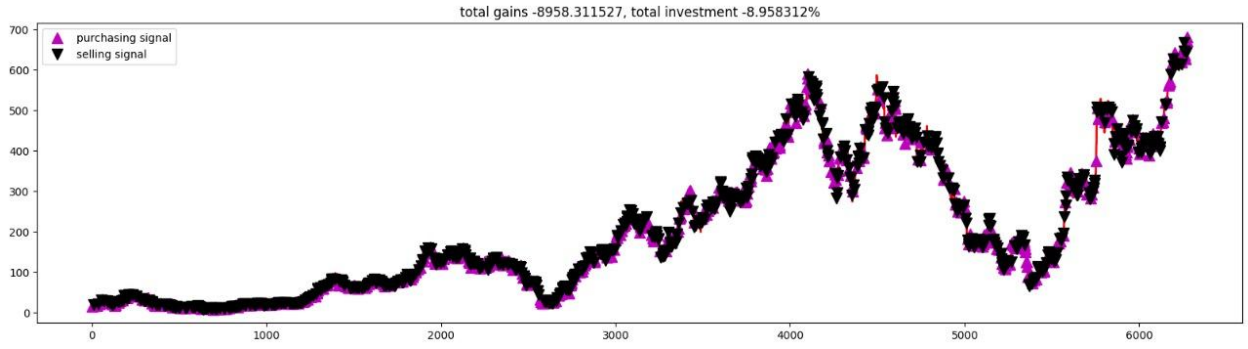
```
close = data_frame.Close.values.tolist()
initial_money = 1_00_000
window_size = 60
skip = 1
batch_size = 32
agent = Agent(state_size = window_size,
              window_size = window_size,
              trend = close,
              skip = skip)
agent.train(iterations = 250, checkpoint = 10, initial_money = initial_money)
```

Double-Q Learning

```
close = df.Close.values.tolist()
initial_money = 1_00_000
window_size = 60
skip = 1
batch_size = 32
agent = Agent(state_size = window_size,
              window_size = window_size,
              trend = close,
              skip = skip,
              batch_size = batch_size)
agent.train(iterations = 1000, checkpoint = 10, initial_money = initial_money)
```

Q Learning

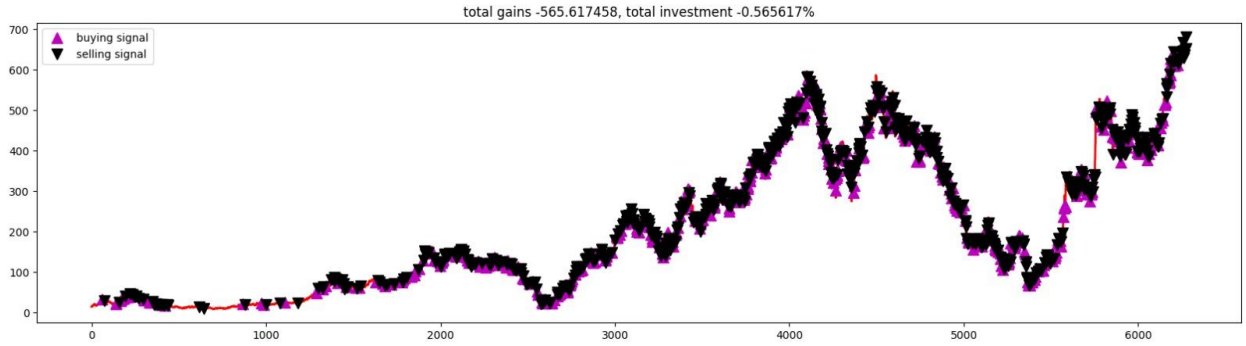
Result:



Double-Q Learning

Summary:

We use reinforcement learning, a machine learning technique that learns from actions and rewards, to develop a robust trading model for stock prediction. We use two reinforcement learning methods: Q-learning and deep Q-learning. Q-learning is a model-free and policy-free method that uses a reward matrix to store and update the expected future rewards for each state-action pair. Deep Q-learning is a variant of Q-learning that uses deep neural networks to approximate the reward function and handle large-scale and complex problems. We use the historical stock data of Asian Paints to train and test our model, and compare the performance of Q-learning and deep Q-learning. We use an initial capital of 1,00,000 and two possible actions: buy and sell. We plot the final profit after all iterations and show the results.



Q Learning

References:

1. Everything You Should Know About Reinforcement Learning (spiceworks.com)
2. Reinforcement Learning. DQN: Q-Learning with Neural Networks | Medium
3. Deep Reinforcement Learning for Stock Prediction (hindawi.com)