Comparison Between Different Bandit Algorithms for Stock Investing

Group members:
MA Rongyue 20826084
SHI Hengtao 20823252

March 11th, 2022

Abstract

Decision-making of investment in stocks has been a long-standing and challenging problem. As we know, there is a quantity of uncertainty in the capital market and the multi-armed bandit algorithm can perform in uncertain environments. The main idea of Multi-Armed Bandit algorithms is to find the balance point in the explore-exploit scenarios, which is also the main challenge.

In this project, we would like to simulate the stock market using the normal distribution and try to use different bandit algorithms, such as ε-Greedy Method, UCB Method, and Gradient Method to test the effectiveness

The project contains three main parts:
1. Stock data simulation
2. Multi-armed bandit algorithm
3. Comparison and evaluation

Group Work Contribution:
We all actively participated in the initial discussion for the overall structure and ideas.

After the outline is settled, SHI Hengtao is mainly responsible for coding, and MA Rongyue is mainly responsible for report writing. Moreover, we also support and conduct cross-checking for others' parts and give suggestions.

1 Stock Data Simulation

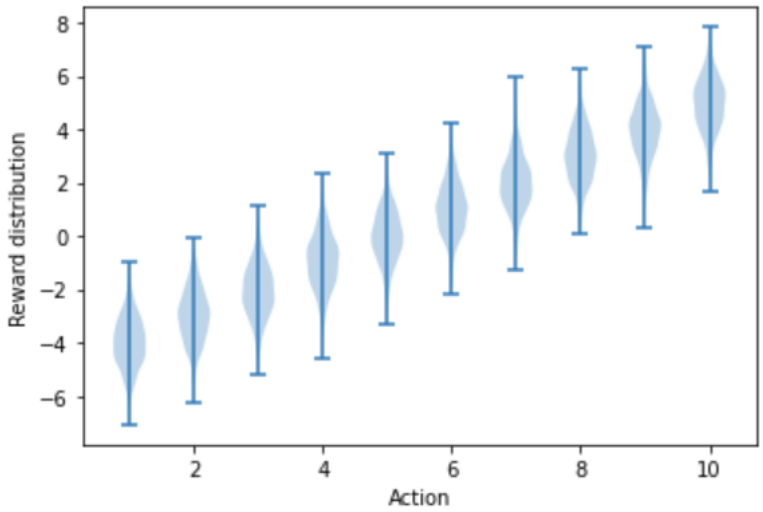
As multi-armed bandits are a simplification of the real-world problem, in order to gain an overall and straightforward view, we would like to simulate the stock data using the normal distribution.

We use the long-only strategy. Moreover, in the ideal trading environment, we assume there is no transaction fee or brokerage fee.

In our data simulation, the true values, which are the returns of stocks, are generated from the normal distribution with the mean varying from -4 to 5. The reason why we do not construct the balanced means is to simulate the real-world market, which is long-term bullish.

The reward distribution is as follows.

```
def constructData(self):
    times = self.times
    stocks = np.random.randn(10, times)
    for m in range(-4, 6):
        stocks[m+4] += m
    return stocks
```



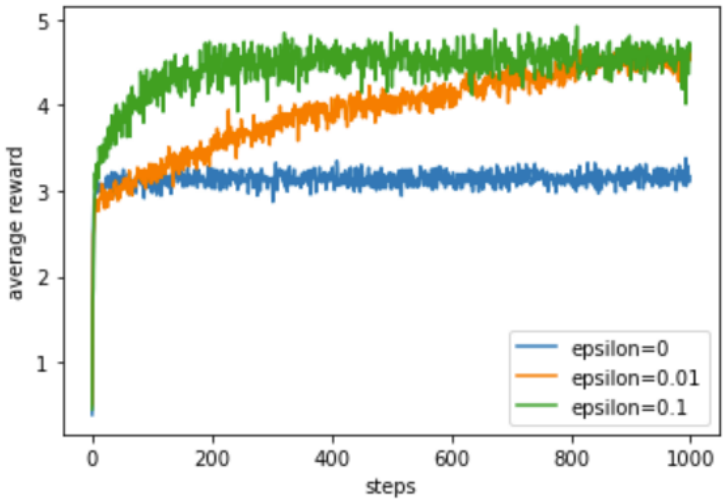
2 Multi-Armed Bandit Algorithm

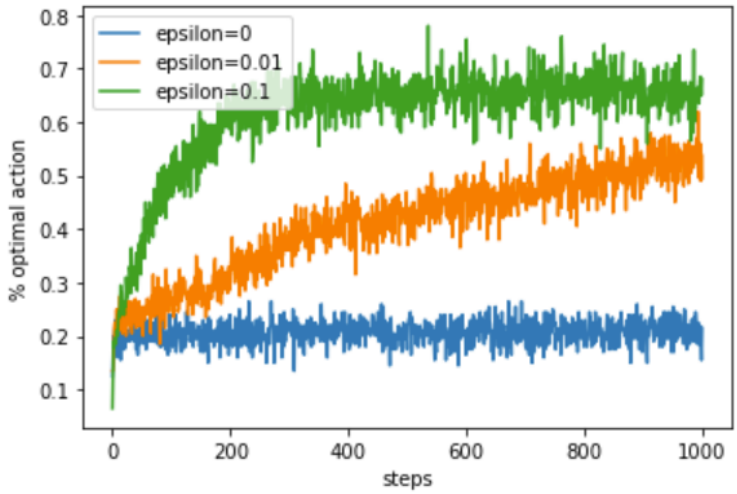
Our investment goal is to find the stocks with the higher average returns, in other words, to maximize our return over time.

In this part, we train our algorithm to maximize the long-term rewards. Meanwhile, we also try different parameters to test the training result.

Epsilon-Greedy Methods With Different ε:
We try three ε parameters, $\varepsilon = 0$, 0.01, and 0.1 to conduct the parameter study.

Comparing the ε-greedy methods, we can find $\varepsilon = 0.1$ performs the best among all methods. To be more specific, $\varepsilon = 0.01$ method improves slower compared with the method with $\varepsilon = 0.1$, and $\varepsilon = 0$ method performed significantly worse in the long run. To explore more on the greedy case (ε=0), we think the bad performance is because it is stuck in the sub-optimal point.


average reward
steps
epsilon=0
epsilon=0.01
epsilon=0.1

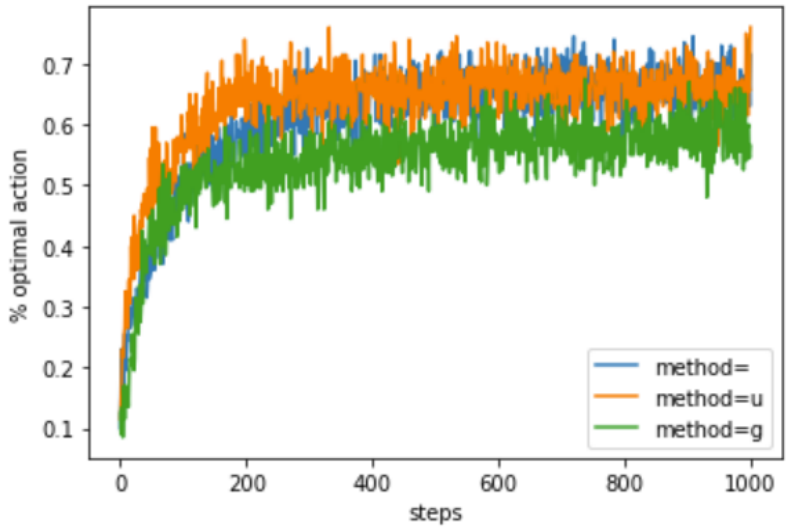

% optimal action
steps
epsilon=0
epsilon=0.01
epsilon=0.1

3 Comparison and Evaluation

In the last part, we use sample averaged rewards for each episode to assess the performance of the agent.

Please find below the output graphs.

The blue line denotes the ε-greedy method with $\varepsilon = 0.1$, the orange line denotes the UCB method, and the green line denotes the gradient method. As a result, we can observe that the three multi-armed bandit algorithms perform similarly under the ideal stock simulation environment. To be more detailed, the UCB method improves slightly faster than the other two methods, and also performs well in the long run. Nevertheless, regarding the complex real-world environment, it may not be the case as the effectiveness depends largely on the specific data and tasks.



4 Code File

The code can be found in Github via https://github.com/NicoleMa1220/RL-Project1