Assignment 3 – Final

Sai Mahesh Pullagura
12/10/2021

Overview

The task of this project is to perform reinforcement learning using the dataset "NVDA". The main objective of the task is to implement Q-Learning algorithm from scratch to train an agent to learn the trends in stock price, perform a series of trades and finally end with a profit. The entire code has been implemented with the help of Python.

Dataset

The data set used in this assignment is the historical stock price for Nvidia for the last 5 years. It has 1258 entries starting 10/27/2016 to 10/26/2021 and the features include information such as the price at which the stock opened, the intraday high and low, the price at which the stock closed, the adjusted closing price and the volume of shares traded for the day.

Data processing

- The data is split into train and test data sets using iloc() function.

Reinforcement Learning

- The main idea behind the Reinforcement Learning is that given an autonomous agent, it must learn to perform a task by trial and error without any guidance from the human operator and in the process, maximize the awards.
- One of the popular algorithm used in Reinforcement Learning is the **Q-Learning** Algorithm.

Q-Learning Algorithm:

- This algortihm can acquire optimal control strategies from delayed rewards even when the agent has no prior knowledge of the effects of its actions on the environment.
- Given a set of states S and actions A, the agent looks to learn control policy $\pi: S \rightarrow A$ so that the expected sum of rewards will be maximized.
- For this purpose we make use of the Q-table which is the combination of S states ad A actions.
- For a current state s and the action considered a, the Q-table is updated as follow:

  $$\hat{Q}(s,a) \leftarrow (1-\alpha) * \hat{Q}(s,a) + \alpha * (r + \gamma * \max_{a'} \hat{Q}(s',a'))$$
- Note that we randomly move across various states by following a random action till we reach the destination. This whole process is called as an episode.
- Once we reach the destination state, we follow the same process iteratively for many episodes till we reach the optimal values in the Q-table.

- Once the training is done, we can test the performance of the learnt Q-table by verifying it on the test data set where instead of choosing the actions randomly, we use that action which yields the best result.

- **Programmatically**, we first begin with initializing the Q-Table with 4 states and 3 actions (for the current problem).

- We then set the no of episodes to the desired value (here we chose 1000). Also, we instantiate other parameters (like discount_factor, Epsilon and Learning_rate etc) utilized while updating the Q-Table.

- In the training stage of the Q-learning algorithm, we begin with resetting the parameters, run through multiple episodes (1000) and update the parameters till we reach the destination state. Once the final state is reached, we again reset the parameters to default value and iterate through the same process till the Q-table gets filled with the best possible optimal values.

- During each episode of the training state, we first determine the action to be worked on. We randomly select either any action or the action that yields best Q-value for the current state. Note that for this purpose, we make use of a parameter 'epsilon' whose value is updated at every episode by the process of decaying using a constant epsilon_decay.

- Then with the obtained action, we call the step() which contains the major implementation steps that decide the upcoming actions. The step() performs the below steps on a high level:

    - Based on the action selected, the agent bus/sells/holds the shares and reciev the immediate reward for that action.
    - Next, with respect to number of shares currently held by the agent, it calculates the total account value.
    - Based on whether the overall price increased ort reduced and if the agent current hold an shares or not, the agent decides one of the 4 possible states to go next time.
- Once the step() executes, it returns 4 values; the next state to travel, the immediate reward received for the current state, whether or not we reached the destination state and a info variable which stores any additional information.
- We then calculate the maximum Q-value possible from the newly obtained next state using below equation:

    max_next_state = np.max(self.Q[new_obs])

- Now, we determine the Q-table value for the current state and the selected action using below formula:

  self.Q[obs][action_index]= self.Q[obs][action_index]+ self.learning_rate*(new_rew + (self.discount_factor*max_next_state) - self.Q[obs][action_index])

- Note that we also store the total reward value of each episode.
- Once we run through all the episodes, we will be left with the optimal Q-table.
- We make used of this table to verify the performance of the model on the Test dataset.
- In the evaluation part, we onl go through one episode by making use of the optimal Q-table obtained in the training stage. Also we only choose the action that yields the best result.
- In this manner, we iterate through all states till we reach the destination state.

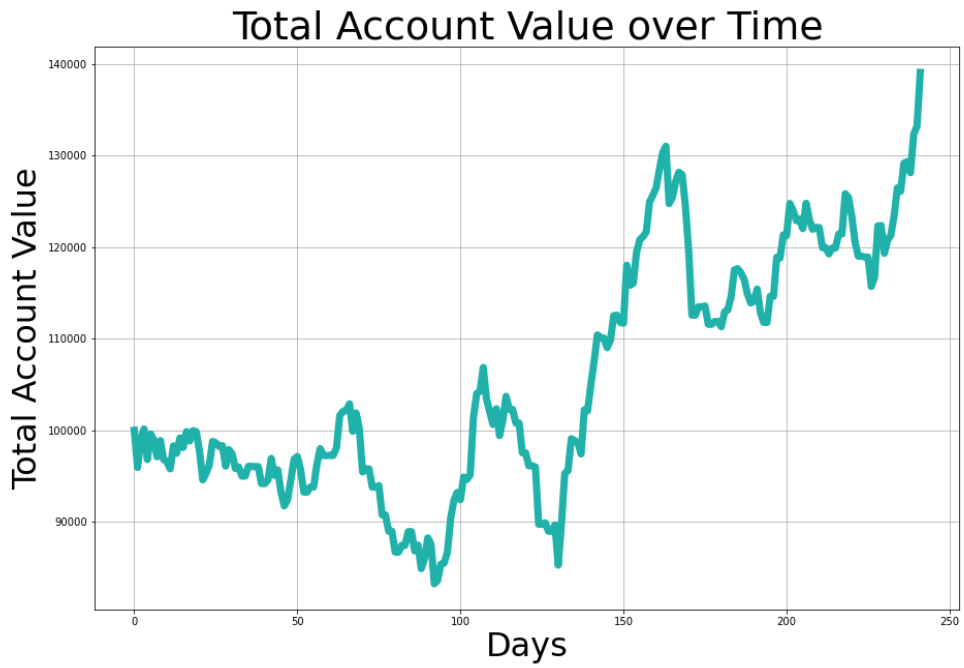
- Finally, once the evaluation process is done, we then display the results using plots and graphs as seen in the results section.

----Next Page----

Results

The accuracy of Q-Learning algorithm in Reinforcement Learning can be seen using below plots.

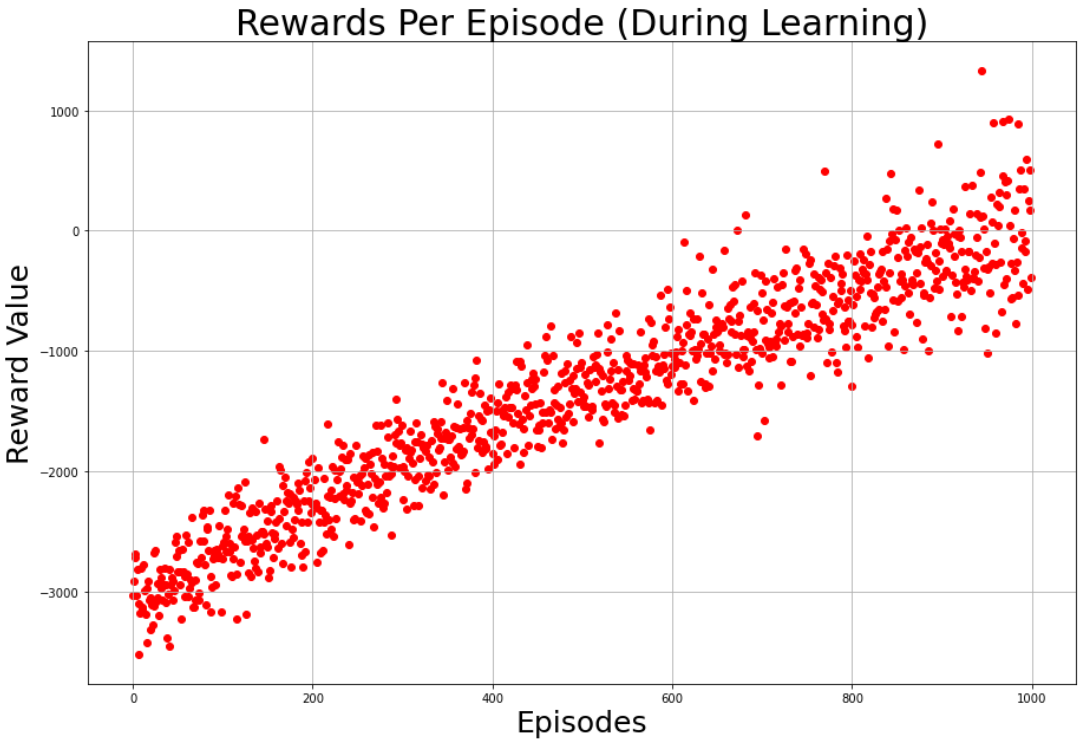
- The plot “Total Account Value over Time” shows how the overall account value of the agent over time. It could be seen that the value raises upward as the number of days passes by.




```
The total account value obtained is:  139095.30966699996
```

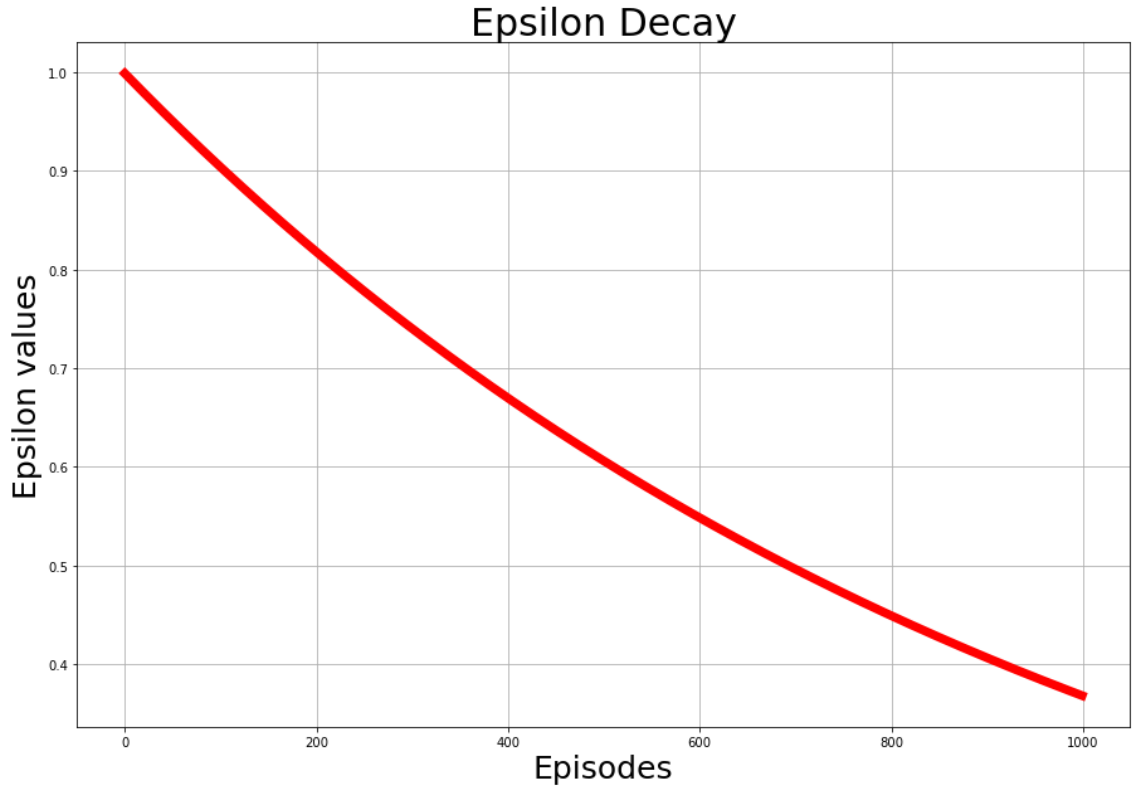
Note that the total account value obtained is: 139095.30966699996.

- The value of reward for each episode is as follows:



As the number of episodes increased, the reward value also got increased.

- The decay of the parameter “Epsilon” for each episode is as follows:



References

- Lecture slides -- 15.1-Reinf-Learning.pdf
- Lecture slides -- 15.3-Q-Learning.pdf
- Chaitra cheluvaraju -- chaitracheluvaraju_assignment_sample_report.pdf
- Url -- https://colab.research.google.com/drive/1E2RViy7xmor0mhqskZV14_NUj2jMpJz3
- Url -- https://piazza.com/class/ksuh71kldm036e?cid=317
- Url -- https://piazza.com/class/ksuh71kldm036e?cid=326