Reinforcement Learning: Assignment #4

Due on Sunday, Mai 14, 2023

Well done!

like always ☺

8

Group 4 - Abu El Komboz, Tareq 3405686 | Jain, Likhit 3678905 | Wurm, Marcel 3695946

Task 1

Monte Carlo Methods vs Dynamic Programming (3P)

(a) What are advantages of Monte Carlo methods over dynamic programming? Mention at least two. (2P)

- Transition function $p(s', r|s, a)$ does not need to be known. Instead the action values for obtaining an optimal policy are estimated from experience with actual or simulated interaction with the environment.
- Less harmed by violating Markov property

(b) Give an example environment where you would use a Monte Carlo method to learn the value function rather than using dynamic programming. Explain why. (1P)

A Monte Carlo method can be used to learn the value function in environments where the dynamics of the environment are unknown, or where the Markov property does not hold. One such example is the game of Blackjack.
Dynamic programming methods such as value iteration or policy iteration require knowledge of the transition probabilities and reward function, which are not available in the case of Blackjack. On the other hand, Monte Carlo methods can estimate the value function from experience, without any knowledge of the dynamics of the game. By simulating many episodes of the game and averaging the returns obtained for each state-action pair, Monte Carlo methods can estimate the true value function.

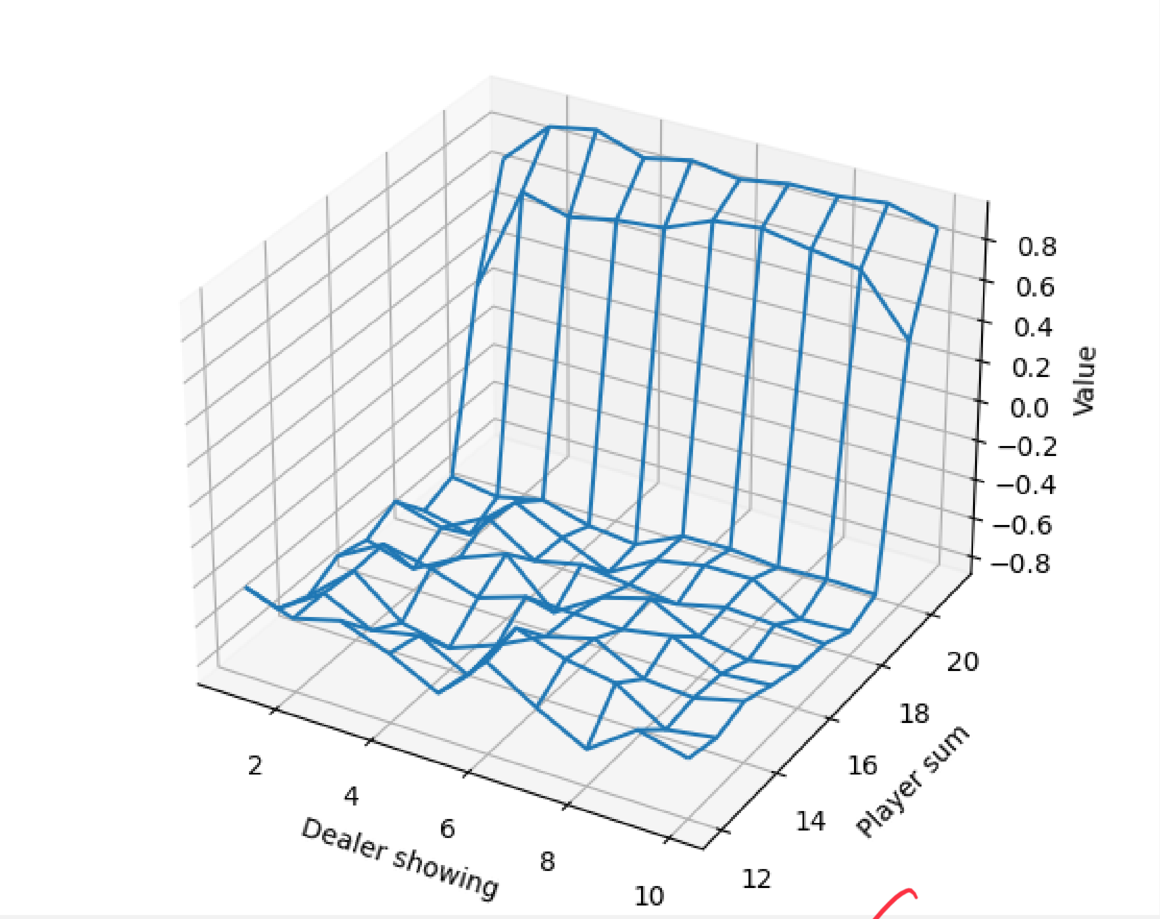
Task 2

Monte Carlo ES for blackjack (6P)

(a) Consider the version of blackjack introduced in the lecture (Example 5.1 from Sutton and Barto). Implement first-visit Monte Carlo prediction (lecture 4 slide 13) for the given policy: stick if sum ≥ 20, else hit. Try to reproduce the figures on slide 16. (3P)

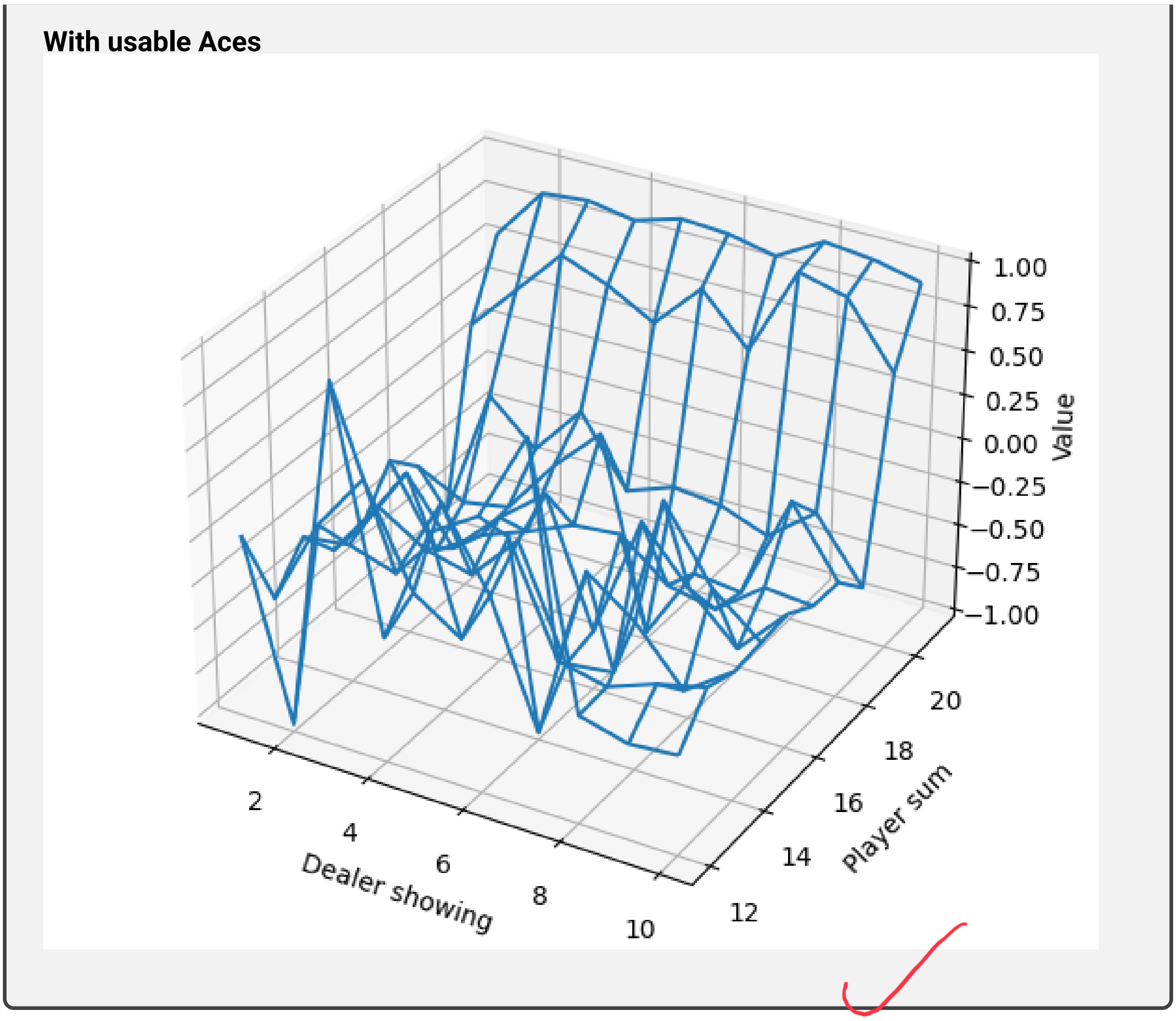
Task 2 continued on next page...

Algorithm after 10000 iterations:
Without usable Aces



✓

Task 2 continued on next page. . .



With usable Aces

(b) Implement Monte Carlo ES (slide 23) and obtain the optimal policy and state-value function for blackjack. Output the policy every 100,000 iterations (e.g. as 2 tables, one with usable ace and one without usable ace). We recommend an optimistic initialisation of Q to improve results. Let it run for at least 500,000 iterations. Letting it run until convergence might take a very long time, so intermediate results are okay! (3P)

Algorithm after 1000000 iterations:
Without usable ace:

dealer player	1	2	3	4	5	6	7	8	9	10
12	1	1	1	1	0	1	1	1	1	1
13	1	1	0	1	1	1	1	1	1	1
14	1	1	1	0	1	1	1	1	1	1
15	1	1	1	0	0	0	1	1	1	1
16	1	0	1	0	1	0	1	1	1	1
17	1	1	1	0	1	1	0	1	1	1
18	0	0	1	0	1	0	0	1	0	1
19	1	0	1	0	0	0	1	1	1	1
20	1	0	1	0	0	1	0	0	0	0
21	1	0	1	0	0	0	0	0	1	0

With usable ace:

dealer player	1	2	3	4	5	6	7	8	9	10
12	0	1	1	0	1	1	0	1	1	1
13	1	1	1	1	1	0	1	1	1	1
14	1	1	1	0	0	0	1	1	1	1
15	1	1	0	0	1	0	0	1	1	1.
16	1	1	0	0	0	1	0	1	1	1
17	0	0	0	0	0	1	1	0	1	1
18	1	1	0	0	1	1	0	1	0	1
19	1	0	0	0	0	1	1	0	1	0
20	1	0	1	0	0	1	1	0	0	0
21	0	0	0	0	0	1	0	0	0	0

✓

Seems like going towards the right direction, But something went wrong.

Nice Try!

-1