CS 188 Spring 2021 | Introduction to Artificial Intelligence | Final Review MDPs+RL

Q1. MDP: Blackjack

There's a new gambling game popping up in Vegas! It's similar to blackjack, but it's played with a single die. CS188 staff is interested in winning a small fortune, so we've hired you to take a look at the game!

We will treat the game as an MDP. The game has states 0,1,. . . ,8, corresponding to dollar amounts, and a *Done* state where the game ends. The player starts with \$2, i.e. at state 2. The player has two actions: Stop and Roll, and is forced to take the Stop action at states 0,1,and 8.

When the player takes the Stop action, they transition to the *Done* state and receive reward equal to the amount of dollars of the state they transitioned from: e.g. taking the stop action at state 3 gives the player \$3. The game ends when the player transitions to *Done*.

The Roll action is available from states 2-7. The player rolls a **biased** 6-sided die that will land on 1, 2, 3, or 4 with $\frac{1}{8}$ probability each and 5 or 6 with probability $\frac{1}{4}$ each.

If the player Rolls from state s and the die lands on outcome o, the player transitions to state $s + o - 2$, as long as $s + o - 2 \leq 8$ (s is the amount of dollars of the current state, o is the amount rolled, and the negative 2 is the price to roll). If $s + o - 2 > 8$, the player busts, i.e. transitions to Done and does NOT receive reward.

(a) In solving this problem, you consider using policy iteration. Your initial policy π^a is in the table below. Evaluate the policy at each state, with $\gamma = 1$. Note that the action at state 0, 1, 8 is fixed into the rule, so we will not consider those states in the update. (*Hint: how does the bias in the die affect this?*)

State	2	3	4	5	6	7
$\pi^a(s)$	Roll	Roll	Stop	Stop	Stop	Stop
$V^{\pi^a}(s)$						

(b) Deciding against the previous policy, you come up with a simpler policy $\pi^{(0)}$, as shown below, to start with. Perform one iteration of Policy Iteration (i.e. policy evaluation followed by policy improvement) to find the new policy $\pi^{(1)}$. In this part as well, we have $\gamma = 1$.

In the table below, R stands for *Roll* and S stands for *Stop*. Select both R and S if both actions are equally preferred.

State	2	3	4	5	6	7
$\pi^{(0)}(s)$	Stop	Stop	Stop	Stop	Stop	Stop
$\pi^{(1)}(s)$	☐ R ☐ S	☐ R ☐ S	☐ R ☐ S	☐ R ☐ S	☐ R ☐ S	☐ R ☐ S

(c) Suppose you start with a initial policy π_0 that is the **opposite** of the optimal policy (which means if $\pi^*(s) = \textit{Roll}, \pi_0(s) = \textit{Stop}$, and vice versa). Your friend Alice claims that the Policy Iteration Algorithm can still find the optimal policy in this specific scenario. Is Alice right?

◯ Alice is right, because Policy Iteration can find the optimal policy regardless of initial policy.
◯ Alice is right, but not for the reason above.
◯ Alice is wrong, because a very bad initial policy can block the algorithm from exploring the optimal actions.
◯ Alice is wrong, but not for the reason above.

(d) Suppose you want to try a different approach, and implement a **value iteration** program to find the optimal policy for this new game. Your friend Bob claims that $V_k(s)$ has to converge to $V^*(s)$ for all states before the program declares it has found the optimal policy. Is Bob right?

◯ Bob is right, because $V_k(s)$ always converge to $V^*(s)$ for all states when the optimal policy π_* is found.
◯ Bob is right, but not for the reason above.
◯ Bob is wrong, because we cannot use value iteration to find the optimal policy.
◯ Bob is wrong, but not for the reason above

Q2. RL: Blackjack, Redux

After playing the Blackjack game in the previous question a few times with the optimal policy you found in the previous problem, you find that you're doing worse than expected! In fact, you are beginning to suspect that the Casino was not honest about the probabilities of dice's outcome. Seeing no better option, you decided to do some good old fashioned reinforcement learning (RL).

(a) First, you need to decide what RL algorithm to use.

(i) Suppose you had a policy π and wanted to find the value V^π of each of the states under this policy. Which algorithms are appropriate for performing this calculation? Note that we **do not** know the transition probabilities, and we don't have sufficient samples to approximate them.

☐ Value Iteration ☐ Policy Iteration ☐ Q-learning ☐ Direct Evaluation ☐ Temporal difference learning

(ii) Being prudent with your money, you decide to begin with observing what happens when other people randomly play the blackjack game. Which of the following algorithms can recover the optimal policy given this play data?

☐ Value Iteration ☐ Policy Iteration ☐ Q-learning ☐ Direct Evaluation ☐ Temporal difference learning

(b) You decide to use Q-learning to play this game.

(i) Suppose your initial policy is π_0. Which of the following is the update performed by Q-learning with learning rate α, upon getting reward $R(s,a,s')$ and transitioning to state s' after taking action a in state s?

○ $Q_{k+1}(s,a) = (1-\alpha)Q_k(s,a) + \alpha(R(s,a,s') + \gamma \max_{a'} Q_k(s',a'))$
○ $Q_{k+1}(s,a) = (1-\alpha)Q_k(s,a) + \alpha(R(s,a,s') + \gamma Q_k(s',\pi_0(s'))$
○ $V_{k+1}(s) = (1-\alpha)V_k(s) + \alpha(R(s,a,s') + \gamma \max_{s''} V_k(s''))$
○ $V_{k+1} = (1-\alpha)V_k + \alpha(R(s,a,s') + \gamma V_k(s'))$

(ii) As with the previous problem, denote a policy at any time-step k as π_k (and $\pi_k(a|s)$ means the **probability** of taking action a at state), and the Q values at that timestep as Q_k. In the limit of infinite episodes, which of these policies will always do each action in each state an infinite amount of times?

☐ $\pi_k(Roll|s) = \pi_k(Stop|s) = \frac{1}{2}$
☐ $\pi_k(a|s) = 1 - \frac{\epsilon}{2}$ if $a == \arg\max_a Q_k(s,a)$ else $\frac{\epsilon}{2}$
☐ $\pi_k(Roll|s) = 1, \pi_k(Stop|s) = 0$
☐ $\pi_k(Roll|s) = \frac{1}{3}, \pi_k(Stop|s) = \frac{2}{3}$
○ None of the above

(iii) Suppose you decide to use an exploration function $f(s',a')$, used in-place of $Q(s',a')$ in the Q-learning update. Which of the following choices of an exploration functions encourage you to take actions you haven't taken much before? (Recall that $N(s,a)$ is the number of times the q-state (s,a) has been visited, assuming every (s,a) has been visited at least once.)

☐ $f(s,a) = Q(s,a)$
☐ $f(s,a) = Q(s,a) + N(s,a)$
☐ $f(s,a) = \max_{a'} Q(s,a')$
☐ $f(s,a) = Q(s,a) + \frac{k}{N(s,a)}$, where $k > 0$
☐ $f(s,a) = Q(s,a) + \sqrt{\frac{\log(\sum_{a'} N(s,a'))}{N(s,a)}}$

☐ $f(s,a) = \frac{1}{N(s,a)^2}$

◯ None of the above

(iv) Suppose you start with the following Q-value table:

State	2	3	4	5	6	7
Q(State, Roll)	0	0	5	3	4	2
Q(State, Stop)	2	3	4	5	6	7

After you observe the trajectory

$$(s = 2, a = \text{Roll}, s' = 4, r = 0), (s = 4, a = \text{Roll}, s' = 7, r = 0), (s = 7, a = \text{Stop}, s' = \textit{Done}, r = 7)$$

What are the resulting Q-values after running one pass of Q-learning over the given trajectory? Suppose discount rate $\gamma = 1$, and learning rate $\alpha = 0.5$.

State	2	3	4	5	6	7
Q(State, Roll)						
Q(State, Stop)						

(v) One of the other gamblers looks over your shoulder as you perform Q-learning, and tells you that you're learning too slowly. "You should use a learning rate of $\alpha = 1$", they suggest.

If you use constant $\alpha = 1$, is Q-learning guaranteed to eventually converge to the optimal policy, assuming you observe every state, action pair an infinite amount of times? ◯ Yes ◯ No

(vi) If you continue with constant $\alpha = 0.5$, is Q-learning guaranteed to eventually converge to the optimal policy, assuming you observe every state, action pair an infinite amount of times? ◯ Yes ◯ No