Q1. MDPs: Micro-Blackjack

In micro-blackjack, you repeatedly draw a card (with replacement) that is equally likely to be a 2, 3, or 4. You can either Draw or Stop if the total score of the cards you have drawn is less than 6. If your total score is 6 or higher, the game ends, and you receive a utility of 0. When you Stop, your utility is equal to your total score (up to 5), and the game ends. When you Draw, you receive no utility. There is no discount ($\gamma = 1$). Let’s formulate this problem as an MDP with the following states: 0, 2, 3, 4, 5 and a Done state, for when the game ends.

1. What is the transition function and the reward function for this MDP?

The transition function is

\[ T(s, \text{Stop, Done}) = 1 \]
\[ T(0, \text{Draw}, s') = \frac{1}{3} \text{ for } s' \in \{2, 3, 4\} \]
\[ T(2, \text{Draw}, s') = \frac{1}{3} \text{ for } s' \in \{4, 5, \text{Done}\} \]
\[ T(3, \text{Draw}, s') = \begin{cases} 
\frac{1}{3} & \text{if } s' = 5 \\
\frac{2}{3} & \text{if } s' = \text{Done} 
\end{cases} \]
\[ T(4, \text{Draw, Done}) = 1 \]
\[ T(5, \text{Draw, Done}) = 1 \]
\[ T(s, a, s') = 0 \text{ otherwise} \]

The reward function is

\[ R(s, \text{Stop, Done}) = s, s \leq 5 \]
\[ R(s, a, s') = 0 \text{ otherwise} \]

2. Perform one iteration of policy iteration for one step of this MDP, starting from the fixed policy below:
Q2. MDPs: Dice Bonanza

A casino is considering adding a new game to their collection, but need to analyze it before releasing it on their floor. They have hired you to execute the analysis. On each round of the game, the player has the option of rolling a fair 6-sided die. That is, the die lands on values 1 through 6 with equal probability. Each roll costs 1 dollar, and the player must roll the very first round. Each time the player rolls the die, the player has two possible actions:

1. \textit{Stop}: Stop playing by collecting the dollar value that the die lands on, or
2. \textit{Roll}: Roll again, paying another 1 dollar.

Having taken CS 188, you decide to model this problem using an infinite horizon Markov Decision Process (MDP). The player initially starts in state \textit{Start}, where the player only has one possible action: \textit{Roll}. State \( s_i \) denotes the state where the die lands on \( i \). Once a player decides to \textit{Stop}, the game is over, transitioning the player to the \textit{End} state.

(a) In solving this problem, you consider using policy iteration. Your initial policy \( \pi \) is in the table below. Evaluate the policy at each state, with \( \gamma = 1 \)

<table>
<thead>
<tr>
<th>States</th>
<th>0</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \pi_i )</td>
<td>Draw</td>
<td>Stop</td>
<td>Draw</td>
<td>Stop</td>
<td>Draw</td>
</tr>
<tr>
<td>( V^{\pi} )</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>( \pi_{i+1} )</td>
<td>Draw</td>
<td>Stop</td>
<td>Stop</td>
<td>Stop</td>
<td>Stop</td>
</tr>
</tbody>
</table>

\textbf{Solution:}

We have that \( s_i = i \) for \( i \in \{3, 4, 5, 6\} \), since the player will be awarded no further rewards according to the policy. From the Bellman equations, we have that \( V(s_1) = -1+1/6(V(s_1)+V(s_2)+3+4+5+6) \) and that \( V(s_2) = -1+1/6(V(s_1)+V(s_2)+3+4+5+6) \). Solving this linear system yields \( V(s_1) = V(s_2) = 3 \).

(b) Having determined the values, perform a policy update to find the new policy \( \pi' \). The table below shows the old policy \( \pi \) and has filled in parts of the updated policy \( \pi' \) for you. If both Roll and Stop are viable new actions for a state, write down both \textit{Roll}/\textit{Stop}. In this part as well, we have \( \gamma = 1 \).

<table>
<thead>
<tr>
<th>State</th>
<th>( s_1 )</th>
<th>( s_2 )</th>
<th>( s_3 )</th>
<th>( s_4 )</th>
<th>( s_5 )</th>
<th>( s_6 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \pi(s) )</td>
<td>Roll</td>
<td>Roll</td>
<td>Stop</td>
<td>Stop</td>
<td>Stop</td>
<td>Stop</td>
</tr>
<tr>
<td>( V^\pi(s) )</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>State</th>
<th>( s_1 )</th>
<th>( s_2 )</th>
<th>( s_3 )</th>
<th>( s_4 )</th>
<th>( s_5 )</th>
<th>( s_6 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \pi'(s) )</td>
<td>Roll</td>
<td>Roll</td>
<td>Roll/Stop</td>
<td>Stop</td>
<td>Stop</td>
<td>Stop</td>
</tr>
</tbody>
</table>
Solution:
For each $s_i$ in part (a), we compare the values obtained via Rolling and Stopping. The value of Rolling for each state $s_i$ is $-1 + 1/6(3 + 3 + 3 + 4 + 5 + 6) = 3$. The value of Stopping for each state $s_i$ is $i$. At each state $s_i$, we take the action that yields the largest value; so, for $s_1$ and $s_2$, we Roll, and for $s_4$ and $s_5$, we stop. For $s_3$, we Roll/Stop, since the values from Rolling and Stopping are equal.

(c) Is $\pi(s)$ from part (a) optimal? Explain why or why not.

Solution:
Yes, the old policy is optimal. Looking at part (b), there is a tie between 2 equally good policies that policy iteration considers employing. One of these policies is the same as the old policy. This means that both new policies are as equally good as the old policy, and policy iteration has converged. Since policy iteration converges to the optimal policy, we can be sure that $\pi(s)$ from part (a) is optimal.

(d) Suppose that we were now working with some $\gamma \in [0, 1)$ and wanted to run value iteration. Select the one statement that would hold true at convergence, or write the correct answer next to Other if none of the options are correct.

- $V^*(s_i) = \max \left\{ -1 + \frac{i}{6}, \sum_j \gamma V^*(s_j) \right\}$
- $V^*(s_i) = \frac{1}{6} \sum_j \max \left\{ -1 + i, \sum_k V^*(s_k) \right\}$
- $V^*(s_i) = \frac{1}{6} \sum_j \max \left\{ -1 + i, \frac{1}{6} \gamma V^*(s_j) \right\}$
- $V^*(s_i) = \frac{1}{6} \sum_j \max \left\{ i, \frac{1}{6} \gamma V^*(s_j) \right\}$
- $V^*(s_i) = \max \left\{ i, -1 + \gamma V^*(s_i) \right\}$
- $V^*(s_i) = \frac{1}{6} \sum_j \max \left\{ i, -1 + \gamma V^*(s_j) \right\}$
- $V^*(s_i) = \frac{1}{6} \sum_j \max \left\{ -1 + \gamma V^*(s_j) \right\}$
- $V^*(s_i) = \frac{1}{6} \sum_j \max \left\{ \frac{-1}{6}, -1 + \gamma V^*(s_j) \right\}$

Other

Solution:
At convergence,
$V^*(s_i) = \max Q'(s_i, a)$
$= \max \{Q'(s_i, stop), Q'(s_i, roll)\}$
$= \max \{R(s_i, stop), R(s_i, roll) + \gamma \sum_j T(s_i, roll, s_j) V^*(s_j)\}$
Q3. Policy Evaluation

In this question, you will be working in an MDP with states $S$, actions $A$, discount factor $\gamma$, transition function $T$, and reward function $R$.

We have some fixed policy $\pi : S \to A$, which returns an action $a = \pi(s)$ for each state $s \in S$. We want to learn the $Q$ function $Q^\pi(s, a)$ for this policy: the expected discounted reward from taking action $a$ in state $s$ and then continuing to act according to $\pi : Q^\pi(s, a) = \sum_{s'} T(s, a, s')[R(s, a, s') + \gamma Q^\pi(s', \pi(s'))]$. The policy $\pi$ will not change while running any of the algorithms below.

(a) Can we guarantee anything about how the values $Q^\pi$ compare to the values $Q^*$ for an optimal policy $\pi^*$?

- $Q^\pi(s, a) \leq Q^*(s, a)$ for all $s, a$
- $Q^\pi(s, a) = Q^*(s, a)$ for all $s, a$
- $Q^\pi(s, a) \geq Q^*(s, a)$ for all $s, a$
- None of the above guaranteed

Solution:

- $Q^\pi(s, a) \leq Q^*(s, a)$ for all $s, a$

(b) Suppose $T$ and $R$ are unknown. You will develop sample-based methods to estimate $Q^\pi$. You obtain a series of samples $(s_1, a_1, r_1), (s_2, a_2, r_2), \ldots, (s_T, a_T, r_T)$ from acting according to this policy (where $a_t = \pi(s_t)$, for all $t$).

(i) Recall the update equation for the Temporal Difference algorithm, performed on each sample in sequence: $V(s_t) \leftarrow (1 - \alpha)V(s_t) + \alpha(r_t + \gamma V(s_{t+1}))$

which approximates the expected discounted reward $V^\pi(s)$ for following policy $\pi$ from each state $s$, for a learning rate $\alpha$.

Fill in the blank below to create a similar update equation which will approximate $Q^\pi$ using the samples.

You can use any of the terms $Q, s_t, s_{t+1}, a_t, a_{t+1}, r_t, r_{t+1}, \gamma, \alpha, \pi$ in your equation, as well as $\Sigma$ and max with any index variables (i.e. you could write $\max_a$, or $\Sigma_a$ and then use $a$ somewhere else), but no other terms.

$Q(s_t, a_t) \leftarrow (1 - \alpha)Q(s_t, a_t) + \alpha[\ldots]$
**Solution:**

\[ Q(s_t, a_t) \leftarrow (1 - \alpha)Q(s_t, a_t) + \alpha[r_t + \gamma Q(s_{t+1}, a_{t+1})] \]

(ii) The algorithms in the previous part (part i) are:
- □ model-based
- □ model-free

**Solution:**

- model-free