CS 188
Summer 2023

Discussion 6A Solutions

Q1. RL: Amusement Park

After the disastrous waterslide experience you decide to go to an amusement park instead. In the previous questions the MDP was based on a single ride (a water slide). Here our MDP is about choosing a ride from a set of many rides.

You start off feeling well, getting positive rewards from rides, some larger than others. However, there is some chance of each ride making you sick. If you continue going on rides while sick there is some chance of becoming well again, but you don't enjoy the rides as much, receiving lower rewards (possibly negative).

You have never been to an amusement park before, so you don't know how much reward you will get from each ride (while well or sick). You also don't know how likely you are to get sick on each ride, or how likely you are to become well again. What you do know about the rides is:

Actions / Rides	Type	Wait	Speed
Big Dipper	Rollercoaster	Long	Fast
Wild Mouse	Rollercoaster	Short	Slow
Hair Raiser	Drop tower	Short	Fast
Moon Ranger	Pendulum	Short	Slow
Leave the Park	Leave	Short	Slow

We will formulate this as an MDP with two states, well and sick. Each ride corresponds to an action. The 'Leave the Park' action ends the current run through the MDP. Taking a ride will lead back to the same state with some probability or take you to the other state. We will use a feature based approximation to the Q-values, defined by the following four features and associated weights:

Features	Initial Weights
$f_0(state, action) = 1$ (this is a bias feature that is always 1)	$w_0 = 1$
$f_1(state, action) = \begin{cases} 1 & \text{if } action \text{ type is Rollercoaster} \\ 0 & \text{otherwise} \end{cases}$	$w_1 = 2$
$f_2(state, action) = \begin{cases} 1 & \text{if } action \text{ wait is Short} \\ 0 & \text{otherwise} \end{cases}$	$w_2 = 1$
$f_3(state, action) = \begin{cases} 1 & \text{if } action \text{ speed is Fast} \\ 0 & \text{otherwise} \end{cases}$	$w_3 = 0.5$

(a) Calculate Q(*'Well'*, *'Big Dipper'*):

$1 + 2 + 0 + 0.5 = 3.5$

(b) Apply a Q-learning update based on the sample (*'Well'*, *'Big Dipper'*, *'Sick'*, -10.5), using a learning rate of $\alpha = 0.5$ and discount of $\gamma = 0.5$. What are the new weights?

Difference $= -10.5 + 0.5 * max(4, 3.5, 2.5, 2.0, 2.0) - 3.5 = -12$
$w_0 = 1 - 6 * 1 = -5$
$w_1 = 2 - 6 * 1 = -4$
$w_2 = 1 - 6 * 0 = 1$
$w_3 = 0.5 - 6 * 1 = -5.5$

(c) Using our approximation, are the Q-values that involve the sick state the same or different from the corresponding Q-values that involve the well state? In other words, is Q(*'Well'*, action) = Q(*'Sick'*, action) for each possible action? Why / Why not? (in just one sentence)

Same
They are the same because we have no features that distinguish between the two states.

Now we will consider the exploration / exploitation tradeoff in this amusement park.

(d) Assume we have the original weights from the table on the previous page. What action will an ϵ-greedy approach choose from the well state? If multiple actions could be chosen, give each action and its probability.
With probability $(1 - \epsilon\frac{4}{5})$ we will choose the Wild Mouse. Each other action will be chosen with probability $\frac{\epsilon}{5}$

(e) When running Q-learning another approach to dealing with this tradeoff is using an exploration function:

$$f(u, n) = u + \frac{k}{n}$$

(i) How is this function used in the Q-learning equations? (a single sentence)
The update replaces the max over Q values with a max over this function (with Q and N as arguments)

What are each of the following? (a single sentence each)

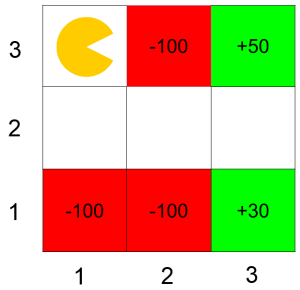
(ii) u :
The utility, given by Q

(iii) n :
The number of times this state-action pair has been visited

(iv) k :
A constant, by adjusting it we can change how optimistic we are about states we haven't visited much.

2 Deep inside Q-learning

Consider the grid-world given below and an agent who is trying to learn the optimal policy. Rewards are only awarded for taking the *Exit* action from one of the shaded states. Taking this action moves the agent to the Done state, and the MDP terminates. Assume $\gamma = 1$ and $\alpha = 0.5$ for all calculations. All equations need to explicitly mention γ and α if necessary.



(a) The agent starts from the top left corner and you are given the following episodes from runs of the agent through this grid-world. Each line in an Episode is a tuple containing (s, a, s', r).

Episode 1	Episode 2	Episode 3	Episode 4	Episode 5
(1,3), S, (1,2), 0	(1,3), S, (1,2), 0	(1,3), S, (1,2), 0	(1,3), S, (1,2), 0	(1,3), S, (1,2), 0
(1,2), E, (2,2), 0	(1,2), E, (2,2), 0	(1,2), E, (2,2), 0	(1,2), E, (2,2), 0	(1,2), E, (2,2), 0
(2,2), E, (3,2), 0	(2,2), S, (2,1), 0	(2,2), E, (3,2), 0	(2,2), E, (3,2), 0	(2,2), E, (3,2), 0
(3,2), N, (3,3), 0	(2,1), Exit, D, -100	(3,2), S, (3,1), 0	(3,2), N, (3,3), 0	(3,2), S, (3,1), 0
(3,3), Exit, D, +50		(3,1), Exit, D, +30	(3,3), Exit, D, +50	(3,1), Exit, D, +30

Fill in the following Q-values obtained from direct evaluation from the samples:

$Q((3,2), \text{N}) =$ 50 $\quad Q((3,2), \text{S}) =$ 30 $\quad Q((2,2), \text{E}) =$ 40

Direct evaluation is just averaging the discounted reward after performing action a in state s.

(b) Q-learning is an online algorithm to learn optimal Q-values in an MDP with unknown rewards and transition function. The update equation is:

$$Q(s_t, a_t) = (1 - \alpha)Q(s_t, a_t) + \alpha(r_t + \gamma \max_{a'} Q(s_{t+1}, a'))$$

where γ is the discount factor, α is the learning rate and the sequence of observations are $(\cdots, s_t, a_t, s_{t+1}, r_t, \cdots)$. Given the episodes in (a), fill in the time at which the following Q values first become non-zero. When updating the Q values, You should only go through each transition once, and the order in which you are to go through them is: transitions in ep 1, transitions in ep 2 and so on. Your answer should be of the form (**episode#,iter#**) where **iter#** is the Q-learning update iteration in that episode. If the specified Q value never becomes non-zero, write *never*.

$Q((1,2), \text{E}) =$ *never* $\quad Q((2,2), \text{E}) =$ (5,3) $\quad Q((3,2), \text{S}) =$ (5,4)

This question was intended to demonstrate the way in which Q-values propagate through the state space. Q-learning is run in the following order - observations in ep 1 then observations in ep 2 and so on.

(c) In Q-learning, we look at a window of (s_t, a_t, s_{t+1}, r_t) to update our Q-values. One can think of using an update rule that uses a larger window to update these values. Give an update rule for $Q(s_t, a_t)$ given the window $(s_t, a_t, r_t, s_{t+1}, a_{t+1}, r_{t+1}, s_{t+2})$.

$Q(s_t, a_t) = (1 - \alpha)Q(s_t, a_t) + \alpha(r_t + \gamma r_{t+1} + \gamma^2 \max_{a'} Q(s_{t+2}, a'))$
(Sample of the expected discounted reward using r_{t+1})

$Q(s_t, a_t) = (1 - \alpha)Q(s_t, a_t) + \alpha(r_t + \gamma((1 - \alpha)Q(s_{t+1}, a_{t+1}) + \alpha(r_{t+1} + \gamma \max_{a'} Q(s_{t+2}, a'))))$
(Nested Q-learning update)

$Q(s_t, a_t) = (1-\alpha)Q(s_t, a_t)+\alpha(r_t+\gamma \max((1-\alpha)Q(s_{t+1}, a_{t+1})+\alpha(r_{t+1}+\gamma \max_{a'} Q(s_{t+2}, a')), \max_{a'} Q(s_{t+1}, a')))$
(Max of normal Q-learning update and one step look-ahead update)