CS6700 : Reinforcement Learning
Written Assignment #1

Topics: Intro, Bandits, MDP, Q-learning, SARSA, PG **Deadline**: 20 March 2023, 23:55
Name: Shivaram **Roll number:** BE20B032

- This is an individual assignment. Collaborations and discussions are strictly prohibited.
- Be precise with your explanations. Unnecessary verbosity will be penalized.
- Check the Moodle discussion forums regularly for updates regarding the assignment.
- Type your solutions in the provided LaTeXtemplate file.
- **Please start early.**

1. (2 marks) [Bandit Question] Consider a N-armed slot machine task, where the rewards for each arm a_i are usually generated by a stationary distribution with mean $Q^*(a_i)$. The machine is under repair when a arm is pulled, a small fraction, ϵ, of the times a random arm is activated. What is the expected payoff for pulling arm a_i in this faulty machine?

Solution:

$$\begin{aligned} \textit{Expected Payoff} &= \mathbb{E}_\epsilon[R] \\ &= \epsilon\mathbb{E}_j[R] + (1-\epsilon)\mathbb{E}[R_i] \\ &= \epsilon\frac{\sum_{j\neq i} Q^*(a_i)}{N-1} + (1-\epsilon)Q^*(a_i) \end{aligned} \tag{1}$$

2. (4 marks) [Delayed reward] Consider the task of controlling a system when the control actions are delayed. The control agent takes an action on observing the state at time t. The action is applied to the system at time $t+\tau$. The agent receives a reward at each time step.

   (a) (2 marks)What is an appropriate notion of return for this task?

Solution: Return for a particular state is given by the discounted sum of rewards received after taking an action observing the state. Therefore the return is given as follows:

$$G_t = \sum_{k=0}^{T} \gamma^k R_{t+\tau+k+1}$$

(b) (2 marks) Give the TD(0) backup equation for estimating the value function of a given policy.

Solution: The value of a state is defined as the expected return we would get after performing an action corresponding to the state. Therefore state s_t is related to $s_{t+\tau+1}$ as action is performed in $t+\tau$ time step. TD(0) Update equation is as follows:

$$V(s_t) \leftarrow V(s_t) + \alpha[R_{t+\tau+1} + \gamma V(s_{t+\tau+1}) - V(s_t)]$$

3. (5 marks) [Reward Shaping] Consider two finite MDPs M_1, M_2 having the same state set, S, the same action set, A, and respective optimal action-value functions Q_1^*, Q_2^*. (For simplicity, assume all actions are possible in all states.) Suppose that the following is true for an arbitrary function $f : S \to R$:

$$Q_2^*(s,a) = Q_1^*(s,a) - f(s)$$

for all $s \in S$ and $a \in A$.

(a) (2 marks) Show mathematically that M_1 and M_2 has same optimal policies.

Solution:
The optimal policy for an MDP with optimal action-value function is given as:

$$\pi^* = argmax_a Q^*(s,a)$$

Therefore the optimal policies for M1 and M2 are as follows:

$$\pi_1^* = argmax_a Q_1^*(s,a)$$

$$\pi_2^* = argmax_a Q_2^*(s,a)$$

We have to prove that

$$\pi_1^* = \pi_2^*$$

-

$$\begin{aligned} \pi_1^* &= argmax_a Q_1^*(s,a) \\ &= argmax_a (Q_2^* + f(s)) \\ &= argmax_a (Q_2^*) + f(s) \\ &= \pi_2^* \end{aligned} \tag{2}$$

Since $f(s)$ is not dependent on a, the term $f(s)$ can be neglected as the argmax used for selection does not depend on it.

(b) (3 marks) Now assume that M_1 and M_2 has the same state transition probabilities but different reward functions. Let $R_1(s, a, s')$ and $R_2(s, a, s')$ give the expected immediate reward for the transition from s to s' under action a in M_1 and M_2, respectively. Given the optimal state-action value functions are related as given above, what is the relationship between the functions R_1 and R_2 ? That is, what is R_1 in terms of R_2 and f; OR R_2 in terms of R_1 and f.

Solution: From the Bellman optimality equation we have:

$$q^*(s,a) = \sum_{s',r} p(s',r|s,a)[r + \gamma max_{a'} q^*(s',a')]$$

Therefore MDP M1 and M2 we have:

$$Q_1^*(s,a) = \sum_{s',r} p_1(s',r|s,a)[R_1(s,a,s') + \gamma max_{a'} Q_1^*(s',a')]$$

$$Q_2^*(s,a) = \sum_{s',r} p_2(s',r|s,a)[R_2(s,a,s') + \gamma max_{a'} Q_2^*(s',a')]$$

By the relation $Q_2^*(s,a) = Q_1^*(s,a) - f(s)$ and $p_1(s,a,s') = p_2(s,a,s')$

Therefore we have,

$$\begin{aligned} f(s) &= Q_1^*(s,a) - Q_2^*(s,a) \\ f(s) &= \sum_{s',r} p(s',r|s,a)[(R_1(s,a,s' - R_2(s,a,s') + \gamma f(s')] \end{aligned} \tag{3}$$

4. (10 marks) [Jack's Car Rental] Jack manages two locations for a nationwide car rental company. Each day, some number of customers arrive at each location to rent cars. If Jack has a car available, he rents it out and is credited $ 10 by the national company. If he is out of cars at that location, then the business is lost. Cars become available for renting the day after they are returned. To help ensure that cars are available where they are needed, Jack can move them between the two locations overnight, at a cost of $ 2 per car moved. We assume that the number of cars requested and returned at each location are Poisson random variables, meaning that the probability that the number n is $\frac{\lambda^n}{n!}e^{-\lambda}$, where λ is the expected number. Suppose λ is 3 and 4 for rental requests at the first and second locations and 3 and 2 for returns. To simplify the problem slightly, we assume that there can be no more than 20 cars at each location (any additional cars are returned to the nationwide company, and thus disappear from the problem) and a maximum of five cars can be moved from one location to the other in one night.

(a) (4 marks) Formulate this as an MDP. What are the state and action sets? What is the reward function? Describe the transition probabilities (you can use a formula rather than a tabulation of them, but be as explicit as you can about the probabilities.) Give a definition of return and describe why it makes sense.

Solution:
State s: (N_1, N_2),
where:
N_1 : Number of cars in location 1.
N_2 : Number of cars in location 2.

Actions:
a: Moving a cars from location 1 to 2. Resulting next state,
$s^{'}$: $(N_1^{'} = N_1 - a, N_2^{'} = N_2 + a)$

Reward Function:

$$r = R_r \sum_{i=1,2} n_i + |a| R_a$$

where:
n_i, number of cars rented in location i.
R_r = \$10, Reward for renting a car.
R_a = -\$2, Cost of moving a car from one location to another.

$N_1,\ N_2 \in [0, 20]$
$a \in [-5, 5]$

Transition Probability:
Let
N_1, N_2 be the number of cars at the start of the day.
$N_1^{'}, N_2^{'}$ be the number of cars after taking action a.
$N_1^{''}, N_2^{''}$ be the number of cars after renting.

Then Transition probabilities is given as:

$$p((N_1^{''}, N_2^{''})|(N_1, N_2), a) = p(N_1^{''}|N_1^{'}) p(N_2^{''}|N_2^{'})$$

where:

$$p(N_i^{''}|N_i^{'}) = \sum_{X_i=0}^{N_i^{'}} p(X_i|\lambda_i, N_i^{'}) \times p(Y_i = N_i^{''} - N_i^{'} + X_i | \mu_i, N_{max} - N_i^{'} + X_i)$$

$$p(X|\lambda, N) = \begin{cases} p(X|\lambda) & X < N \\ \sum_{Z=N}^{\infty} p(Z|\lambda) & X = N \end{cases} \tag{4}$$

$X \rightarrow$ number of cars rented out.
$Y \rightarrow$number of cars returned back.

Return:
The Return can be defined as the discounted sum of money(reward) that Jack gets every time-step.

(b) (3 marks) One of Jack's employees at the first location rides a bus home each night and lives near the second location. She is happy to shuttle one car to the second location for free. Each additional car still costs \$ 2, as do all cars moved in the other direction. In addition, Jack has limited parking space at each location. If more than 10 cars are kept overnight at a location (after any moving of cars), then an additional cost of \$ 4 must be incurred to use a second parking lot (independent of how many cars are kept there). These sorts of nonlinearities and arbitrary dynamics often occur in real problems and cannot easily be handled by optimization methods other than dynamic programming. Can you think of a way to incrementally change your MDP formulation above to account for these changes?

Solution:

$$\text{Shifting cost} = \begin{cases} 2(a-1) & \text{if a} \geq 0 \\ 2(-a) & \text{if a} < 0 \end{cases} \tag{5}$$

$$\text{Parking cost} = \begin{cases} 8 & \text{if } N_1 - a > 10 \text{ and } N_2 - a > 10 \\ 4 & \text{if } N_1 - a > 10 \text{ and } N_2 - a < 10 \\ 4 & \text{if } N_1 - a < 10 \text{ and } N_2 - a > 10 \\ 0 & \text{if } N_1 - a < 10 \text{ and } N_2 - a < 10 \end{cases} \tag{6}$$

Total reward is given as follows:

$$R(s_t, a) = 10(X_1 + X_2) - \text{Shifting cost} - \text{Parking cost}$$

(c) (3 marks) Describe how the task of Jack's Car Rental could be reformulated in terms of *afterstates*. Why, in terms of this specific task, would such a reformulation be likely to speed convergence? *(Hint:- Refer page 136-137 in RL book 2nd edition. You can also refer to the video at https://www.youtube.com/watch?v=w3wGvwi336I)*

Solution: Now the state is defined as a tuple of after states as $S = \{N_1^i, N_2^i\}$, where N_1^i,N_2^i is the initial number of cars in location 1 and 2 respectively. The usual method learned two values for each state while here we learn only afterstate which uses data from both the cases. We need to save only the value function of the after state and not the action value function of the states.

5. (8 marks) [Organ Playing] You receive the following letter:
Dear Friend, Some time ago, I bought this old house, but found it to be haunted by ghostly sardonic laughter. As a result it is hardly habitable. There is hope, however, for by actual testing I have found that this haunting is subject to certain laws, obscure but infallible, and that the laughter can be affected by my playing the organ or burning incense. In each minute, the laughter occurs or not, it shows no degree. What it will do during the ensuing minute depends, in the following exact way, on what has been happening during the preceding minute: Whenever there is laughter, it will continue in the succeeding minute unless I play the organ, in which case it will stop. But continuing to play the organ does not keep the house quiet. I notice, however, that whenever I burn incense when the house is quiet and do not play the organ it remains quiet for the next minute. At this minute of writing, the laughter is going on. Please tell me what manipulations of incense and organ I should make to get that house quiet, and to keep it so.
Sincerely,
At Wits End

   (a) (4 marks) Formulate this problem as an MDP (for the sake of uniformity, formulate it as a continuing discounted problem, with $\gamma = 0.9$. Let the reward be +1 on any transition into the silent state, and -1 on any transition into the laughing state.) Explicitly give the state set, action sets, state transition, and reward function.

Solution:
States:
Here we have two states "laughter" and "quiet", which we represent as L and Q respectively.

$$\mathcal{S} = \{L, Q\}$$

.
Actions:
We have the following four actions:

- $a_B \rightarrow$ Burning incense.
- $a_O \rightarrow$ Playing Organ.

- $a_{BO} \rightarrow$ Both actions.
- $a_I \rightarrow$ Staying Idle.

$$\mathcal{A} = \{a_B, a_O, a_{BO}, a_I\}$$

Transition Probability:

$$p(Q|L, a \in \{a_O, a_{BO}\}) = 1$$
$$p(Q|Q, a \in \{a_B\}) = 1$$
$$p(L|L, a \in \{a_I, a_B) = 1$$
$$p(L|Q, a \in \{a_I, a_O\} = 1$$

Reward Function:

$$R(s = L) = -1$$
$$R(s = Q) = +1$$

(b) (2 marks) Starting with simple policy of **always** burning incense, and not playing organ, perform a couple of policy iterations.

Solution:
Initializing the Value function as $V(s) = [0, 0]$ and $\pi(s) = [a_B, a_B]$, where 1st index for state "L" and state "Q".
Iteration 1:
Policy Evaluation:
The update equation is as follows:

$$V(s) \leftarrow \sum_{s',r} p(s', r|s, a)[r + \gamma V(s')]$$

We have the following updates:

$$V(L) \leftarrow p(L, -1|L, a_B)[-1 + 0.9 \times V(L)] = -1$$
$$V(Q) \leftarrow p(Q, 1|Q, a_B)[1 + 0.9 \times V(Q)] = +1$$

Policy Improvement:
Update equation is as follows:

$$\pi(s) \leftarrow argmax_a \sum_{s',r} p(s', r|s, a)[r + \gamma V(s')]$$

We have the following updates:

$$\pi(L) \leftarrow \{a_O, a_{BO}\}$$

$$\pi(Q) \leftarrow \{a_B\}$$

Iteration 2:
Policy Evaluation:
We have the following updates:

$$V(L) \leftarrow p(Q, 1|L, a_O)[1 + 0.9 \times V(L)] = 1.9$$

$$V(Q) \leftarrow p(Q, 1|Q, a_B)[1 + 0.9 \times V(Q)] = 1.9$$

Policy Improvement:
We have the following updates:

$$\pi(L) \leftarrow \{a_O, a_{BO}\}$$

$$\pi(Q) \leftarrow \{a_B\}$$

After two iterations of *Policy Iteration* we have the following value function and policy

$$V(s) = [1.9, 1.9]$$

$$\pi(s) = [\{a_O, a_{BO}\}, a_B]$$

(c) (2 marks) Finally, what is your advice to "At Wits End"?

Solution: From the above policy iteration, we get an optimal policy of choosing to play organ along with/ without burning incense in "L" state and choosing to burn incense in state "Q".
From this inference, I would suggest "At Wits End" to play organ if the state is "laughter" and to burn incense if the state is "quiet".

6. (4 marks) [Stochastic Gridworld] An ϵ-greedy version of a policy means that with probability 1-ϵ we follow the policy action and for the rest we uniformly pick an action. Design a stochastic gridworld where a deterministic policy will produce the same trajectories as a ϵ-greedy policy in a deterministic gridworld. In other words, for every trajectory under the same policy, the probability of seeing it in each of the worlds is the same. By the same policy I mean that in the stochastic gridworld, you have a deterministic policy and in the deterministic gridworld, you use the same policy, except for ϵ fraction of the actions, which you choose uniformly randomly.

(a) (2 marks) Give the complete specification of the world.

Solution: Consider the following grid world with 4 states.

A	B
D	C

The action space is as follows $\mathcal{A} = \{UP, DOWN, LEFT, RIGHT\}$
Let's assume the following policy:

- In *State A*, we move right to B.
- In *State B*, we move down to C.
- In *State C*, we move left to D.
- In *State D*, we move up to A.

$\epsilon-$greedy in Deterministic Gridworld
With probability $(1 - \epsilon)$ action is chosen based on the policy and with probability ϵ, random action is taken from the action space.

Trajectories from:

- *State A*
  - with $(1 - \epsilon/2)$, we end in B.
  - with $\epsilon/2$, we end in D.
- *State B*
  - with $(1 - \epsilon/2)$, we end in C.
  - with $\epsilon/2$, we end in A.
- *State C*
  - with $(1 - \epsilon/2)$, we end in D.
  - with $\epsilon/2$, we end in B.
- *State D*
  - with $(1 - \epsilon/2)$, we end in A.
  - with $\epsilon/2$, we end in C.

Stochastic Gridworld
For probability $(1-\epsilon)$ we end up in the right state and with (ϵ) probability we end up in either of the adjacent states.

Therefore we have trajectories as follows:

- *State A* action selected is moving $RIGHT$.
  - with $(1-\epsilon/2)$, we end in B.
  - with $\epsilon/2$, we end in D.
- *State B* action selected is moving $DOWN$
  - with $(1-\epsilon/2)$, we end in C.
  - with $\epsilon/2$, we end in A.
- *State C* action selected is moving $LEFT$
  - with $(1-\epsilon/2)$, we end in D.
  - with $\epsilon/2$, we end in B.
- *State D* action selected is moving UP
  - with $(1-\epsilon/2)$, we end in A.
  - with $\epsilon/2$, we end in C.

Thus we obtained two Gridworlds where one with Deterministic Gridworld with $\epsilon-$greedy policy and Stochastic Gridworld with a Deterministic policy with the same trajectories.

(b) (2 marks) Will SARSA on the two worlds converge to the same policy? Justify.

Solution: We have the following update equation for SARSA:

$$q(s,a) \leftarrow q(s,a) + \alpha[r + \gamma q(s^{'}, \pi(s^{'}))]$$

The update depends on $s, a, s^{'}, a^{'}$, where the parameters $s^{'}, a^{'}$ depends on the dynamics for a given s, a. The underlying dynamics remain the same for both the grid worlds, therefore SARSA converges in both grid worlds.

7. (5 marks) [Contextual Bandits] Consider the standard multi class classification task (Here, the goal is to construct a function which, given a new data point, will correctly

predict the class to which the new point belongs). Can we formulate this as contextual bandit problem (Multi armed Bandits with side information) instead of standard supervised learning setting? What are the pros/cons over the supervised learning method. Justify your answer. Also describe the complete Contextual Bandit formulation.

Solution:
The above classification problem can be formulated as Contextual Bandits problem, where each data point is considered as a state where the features of the data point represent the context information for the bandit and labels be the arms of the bandits. If the bandit pulls the correct label, it gets a positive reward and if other arms are chosen it receives a negative reward or no reward.

Pros: In contextual bandits, we can perform explore-exploit to either choose arms with high rewards or explore which is not possible in supervised learning. Contextual Bandit is an online learning algorithm where the update happens in real, unlike the supervised learning setting.

Cons: The same data point might not appear twice, so the agent might behave differently each time.

8. (5 marks) [TD, MC, PG] Suppose that the system that you are trying to learn about (estimation or control) is not perfectly Markov. Comment on the suitability of using different solution approaches for such a task, namely, Temporal Difference learning, Monte Carlo methods, and Policy Gradient algorithms. Explicitly state any assumptions that you are making.

Solution:
Temporal Difference Learning:
Temporal difference learning is a bootstrapping method which does an n-step bootstrap to estimate the value of the current state. Where we have the TD target as $G_t = R_{t+1} + \gamma V(s_t)$. The underlying theory of TD assumes Markov property, Hence using Temporal Difference Learning in a Non-Markovian environment would lead to bias in the estimate. The bias decreases as the number of updates increases and can still work in the slightly non-markovian environment.

Monto Carlo:
Monto Carlo method unlike TD learning does sampling of trajectories to make an update. An entire sample is used to estimate the value of a state and there is no explicit usage of Markov property. Therefore, Monto Carlo methods work robustly in Non-markovian environments compared to TD learning.

Policy Gradient:
Policy Gradient methods do not consider the Markovian nature of the environment. PC learns the policy directly from the reward signal and the observed state by updating the parameters of the policy estimator. Hence PC works best in a Non-Markov environment.

9. (5 marks) [PG] Recent advances in computational learning theory, have led to the development of very powerful classification engines. One way to take advantage of these classifiers is to turn the reinforcement learning problem into a classification problem. Here the policy is treated as a labeling on the states and a suitable classifier is trained to learn the labels from a few samples. Once the policy is adequately represented, it can be then used in a policy evaluation stage. Can this method be considered a policy gradient method? Justify your answer. Describe a complete method that generates appropriate targets for the classifier.

Solution: This cannot be considered as a policy gradient method, as the policy is not directly updated from the reward and state, but it is used as a labelling problem which trains the classifier.

Run the current policy to collect samples (s,a) and label the states s using the policy in the state and we acquire (s, $a^{'}$). Now train the classifier on (s, $a^{'}$). Use the classifier to predict during the evaluation phase. The predicted labels are used to evaluate the current policy.