CS6700 : Reinforcement Learning
Written Assignment #1

Topics: Intro, Bandits, MDP, Q-learning, SARSA, PG **Deadline**: 20 March 2023, 23:55
Name: Purvam Jain **Roll number:** EE20B101

- 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: Expected payoff $= (1-\epsilon)Q^*(a_i) + \frac{\epsilon}{N}\sum_{j=1}^{N} Q^*(a_j)$
   First Term, refers reward to when a_i is selected and second term is reward for when a random arm is selected, which will we average reward for all arms. Here, we assume that in random case the original arm a_i can be selected as well.

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: Assuming $0 < \tau < 1$,
   $G_t = R_{t+\tau+1} + \lambda R_{t+\tau+2} + \lambda^2 R_{t+\tau+3} .... = \sum_{i=1}^{\infty} \lambda^{i-1} R_{t+\tau+i}$

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

Solution: From G_t it follows that value of next sample will be $R_{t+\tau+1} + \lambda V(S_{t+\tau+1})$. Therfore $TD(0)$ equation:
$V(S_t) \leftarrow V(S_t) + \alpha[R_{t+\tau+1} + \lambda V(S_{t+\tau+1}) - V(S_t)]$

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 $f : S \rightarrow R$:

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

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: $\pi^* = argmax_a[Q(s,a)]$
$\pi_1^* = argmax_a[Q_1(s,a)] = argmax_a[Q_1(s,a) - f(s)]$
Since function $f(s)$ doesn't depend on action variable a, $max_a(f(s))$ will be a constant and doesn't affect the optimal action taken, which in turn is dependent only on $Q_1^*(s,a)$ as is the case for $Q_2^*(s,a)$. Therefore, we can conclude that M_1 and M_2 share the same optimal policies.

(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: Using Bellman optimality equation,
$Q_1^*(s,a) = R_1(s,a) + \sum_{s'} p(s'|s,a)[\gamma * max_{a'} Q_1^*(s',a')]$
$Q_2^*(s,a) = R_2(s,a) + \sum_{s'} p(s'|s,a)[\gamma * max_{a'} Q_2^*(s',a')]$
$Q_2^*(s,a) = R_2(s,a) + \sum_{s'} p(s'|s,a)[\gamma * max_{a'} (Q_1^*(s',a') - f(s'))]$
Since $f(s')$ isn't dependent on action variable, we can take it out, and then using the equation,
$Q_2^*(s,a) = Q_1^*(s,a) - f(s)$
Since we are given that transition probabilities are same, we are left with:
$R_2(s,a) - \sum_{s'} p(s'|s,a)[f(s')] = R_1(s,a) - f(s)$
Since we are asked for reward for a give transition above equation reduces to,
$R_2(s,a,s') - \gamma * f(s') = R_1(s,a,s') - f(s)$

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 space **S**: Number of cars at each location at end of the day
   Actions space **A**: Net number of cars moved between the two locations overnight
   Rewards **R**: expected rentals at both location minus cost of moving
   $r = E(rent_first + rent_second) - costofmoving$
   Transition Probabilities **P**: Total probability of rentals and returns at both locations, each probability term in p follows the poisson distribution for random variable as is defined in question:
   $p = loc1_rental * loc1_return * loc2_rental * loc2_return$
   Return **G**: Since this task is defined as a continuous finite MDP with time steps as days our return will have a discount factor as:
   $G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$

   (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: These can be accounted for by making some changes in the reward function. We can add a cost of 4 when there are 10 cars in the parking lot for the additional parking lot. Also, we can have 0 cost for the first car moved by the employee.

(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: Initially we have states as the number of cars in each garage each evening. The agent then takes some action (moves some cars) and we transition stochastically to some state by estimating the expected revenue and car counts at each location .Instead we can introduce the number of cars in the morning (after the agent has moved cars) as an afterstate. This is because the agent is able to deterministically change the environment from evening to next morning (before rentals or returns). In this case we would speed convergence by reducing the number of action-values to be calculated. After-state formulation allows for removing of one degree of freedom from this mapping - the car transfer count.

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: State Set **S** = {L: Laughter, S = Silence}
Action Set **A** = {OI',O'I,OI,O'I'} here O: playing organ, I: Burning Incense, and ' implies negation. State Transistions: **S X A X S** = {L(OI)S, L(OI')S, L(O'I)L, L(O'I')L, S(OI)L, S(OI')L, S(O'I')L, S(O'I)S}
Reward **R**: R(L(OI)S) = 1, R(L(OI')S) = 1, R(S(O'I)S) = 1, R(S(O'I')L) = -1, R(L(O'I)L) = -1, R(L(O'I')L) = -1, RS(OI)L) = -1, R(S(OI')L) = -1

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

Solution: $\pi_0(L) = pi_0(S) = (O'I), v_0(L) = v_0(S) = 0, \gamma = 0.9$

- **Policy Evaluation**:
  $v_0(s) \leftarrow \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma * V(s')]$
  $v_0(L) = 1 * [-1 + \gamma * 0]$
  $v_0(L) = 1 * [-1 + \gamma * -1]$
  $v_0(L) = 1 * [-1 + \gamma * (-1 - \gamma)] \dots$
  $v_{\pi 0}(L) = -1 - \gamma - \gamma^2 - \gamma^3 \dots = \frac{-1}{1-\gamma} = -10$ $v_0(L) = 1 * [1 + \gamma * 0]$
  $v_0(L) = 1 * [1 + \gamma * 1]$
  $v_0(L) = 1 * [1 + \gamma * (1 + \gamma)] \dots$
  $v_{\pi 0}(L) = 1 + \gamma + \gamma^2 + \gamma^3 \dots = \frac{1}{1-\gamma} = 10$

- **Policy Improvement**:
  $\pi_1(s) \leftarrow argmax_a \sum_{s',r} p(s', r|s, a)[r + \gamma * V(s')]$
  $\pi_1(L) \leftarrow (O, I) or (O, I')$
  $\pi_1(S) \leftarrow (O', I)$

- **Policy Evaluation**:
  $v_1(L) = 0.5 * [1 + \gamma * 10] + 0.5 * [1 + \gamma * 10] = 10$
  $v_1(S) = 1 * [1 + \gamma * 10] = 10$

- **Policy Improvement**:
  $\pi_2(L) \leftarrow (O, I) or (O, I')$
  $\pi_2(S) \leftarrow (O', I)$
  $\pi_1 = \pi_2 = \pi_*$ {Optimal Policy}

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

Solution:
Dear Friend,
I have come up with the following optimal policy to help you. If there is laughter

then you must play the organ and then continue burning the incense forever. If the room is silent, then you must burn the incense and must not play the organ. Hope you find this advice helpful.

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:
   $S : s_1, s_2, s_3...., s_n$
   $A : a_1, a_2, a_3...., a_m$
   Stochastic Gridworld: Policy π is deterministic. That is , given the state our action is fixed. But the environment is stochastic. $P(s'|s, a) = P(s'|s)$.
   Deterministic Gridworld: Policy π is stochastic. That is, agent takes greedy action with probability $1-\epsilon$ and random otherwise. But once the action is fixed, transition probability is also fixed. Therefore, $P(s'|s, a) = \pi(a|s)$

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

   Solution: Yes, SARSA will converge to the same policy for both worlds.
   $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha * [R_{t+1} + \gamma * Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)]$
   Since the transition probability for both the worlds is same,
   $P(s'|s) = \pi(a|s) * P(s'|s, a)$
   Therefore same trajectory with same probability will be followed in both worlds and eventually value functions will converge to equal values according to Bellman equation.

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: Yes, we can formulate multi-class classification as a contextual bandit problem. We will have as many arms as the number of classes. We will have side information space(contextual information). These will be features that describe the data point. Then we will have action space which will correspond to producing class label given side information. And the reward function can be set to give +1 for correct classification and -1 otherwise. Policy will be to maximize over the cumulative reward.

Contextual Bandits can be computationally less expensive compared to larger neural networks. Also, we may not need as many labelled datapoints as would be the case for a supervised framework since bandits will learn from interaction with the environment(although we do need rewards for interaction) and make decisions based on features alone, while we need to provide true labels for supervised learning. But contextual bandits may require considerable amount of time to converge.

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:

- TD Learning: Performs better when the dependence on previous states is limited to some window or for a short interval of memory. Assumes value funtion is a linear combination dependent on some previous and future states. Suitable when the correlation with previous states in short and weak. Note: TD(1) is basically Monte Carlo. So analysis can change based on length($\lambda \in (0, 1)$) to some extent.
- Monte Carlo: Performs well when with strong dependence on previous states and actions (upto infinity). Assumes each episode is independent which might not be true. Requires a lot of episodes to converge.
- Policy Gradient: Performs well with continuous state action space and complex policy function. Require many samples for gradients to converge.

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: No, This method is trained to output a label based on the input state, which is different from reward maximization. Also, policy gradient follows stochastic ascent to maximize return on policy, while classification engines depend on minimizing the loss function to predict true labels(REINFORCE). Also, it doesn't take advantage of any future actions or rewards as is done in actor-critic methods. We need to define an action-reward pair which connects predicting a label with a reward based on if it's correct or incorrect. This can be done by making rewards inversely proportional to loss. That is if the loss is minimized we give a higher reward and if loss increases we give negative reward.