RBE 595 — Reinforcement Learning
Week #3 Assignment

Arjan Gupta

Problem 1

Suppose $\gamma = 0.8$ and we get the following sequence of rewards

$$R_1 = -2,\ R_2 = 1,\ R_3 = 3,\ R_4 = 4,\ R_5 = 1.0$$

Calculate the value of G_0 by using the equation 3.8 (work forward) and 3.9 (work backward) and show they yield the same results.

Answer

Work Forward

From the the book, the *discounted return* (equation 3.8), G_t, is defined as,

$$G_t \doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \tag{3.8}$$

Plugging in the values from this problem, we get,

$$\begin{aligned} G_0 &= R_1 + \gamma R_2 + \gamma^2 R_3 + \gamma^3 R_4 + \gamma^4 R_5 \\ &= -2 + 0.8 \cdot 1 + 0.8^2 \cdot 3 + 0.8^3 \cdot 4 + 0.8^4 \cdot 1 \\ &= -2 + 0.8 + 0.64 \cdot 3 + 0.512 \cdot 4 + 0.4096 \\ &= 3.1776 \end{aligned}$$

Work Backward

From the book, the "recursive" representation of *discounted return* (equation 3.9), G_t, is defined as,

$$G_t \doteq R_{t+1} + \gamma G_{t+1} \tag{3.9}$$

Plugging in the values from this problem, we get,

$$\begin{aligned} G_0 &= R_1 + \gamma G_1 \\ &= -2 + 0.8 \cdot G_1 \end{aligned}$$

Where we apply 3.8 to G_1,

$$\begin{aligned} G_1 &= R_2 + \gamma R_3 + \gamma^2 R_4 + \gamma^3 R_5 \\ &= 1 + 0.8 \cdot 3 + 0.8^2 \cdot 4 + 0.8^3 \cdot 1 \\ &= 6.472 \end{aligned}$$

Therefore,

$$\begin{aligned} G_0 &= -2 + 0.8 \cdot G_1 \\ &= -2 + 0.8 \cdot 6.472 \\ &= 3.1776 \end{aligned}$$

Conclusion

We see that both methods yield the same result, $G_0 = 3.1776$.

Problem 2

Explain how a room temperature control system can be modeled as an MDP? What are the states, actions, rewards, and transitions.

Answer

A room temperature control system can be modeled as an MDP as follows.

States: The states are the different temperatures that the room can be in.

Actions: The actions are the different actions that the system can take to change the temperature of the room.

Rewards: The rewards are the different rewards that the system can receive for taking an action.

Transitions: The transitions are the different transitions that the system can make from one state to another.

Problem 3

In the equation,

$$NewEstimate = OldEstimate + StepSize \cdot [Target - OldEstimate]$$

what is the target?

Answer

In general, the target is the presumed desired value of the action-value function that we are trying to estimate, or the direction we are trying to move towards. That is why $[Target - Estimate]$ is known as the *error* in the estimate.

In the specific case of the multi-armed bandit problem, the target is the reward that we received after taking the action. With each time step, we attempt to get closer to the target value. For the sample-average method used in the multi-armed bandit problem, the equation in the problem takes on the following form,

$$Q_{n+1} = Q_n + \frac{1}{n}(R_n - Q_n)$$

where the target is the reward R_n that we received after taking the action.

Problem 4

What is the purpose of using Upper Confidence Bound (UCB)?

Answer

The purpose of using UCB is to provide exploratory behavior balanced with exploitation in a systematic manner.

The UCB formula is as follows,

$$A_t = argmax_a \left[Q_t(a) + c\sqrt{\frac{\ln t}{N_t(a)}} \right]$$

Here, the square root term is a measure of the uncertainty in the estimate of the current action a, and c is used to control the confidence level of that uncertainty. The way this square root term works is that it increases if the action has not been chosen often (because denominator $N_t(a)$ is the number of times the action is chosen), and decreases if the action has been chosen often. This means that the action will be chosen more often if it has not been chosen much in the past, and less often if it has been chosen a lot in the past. The nature of the logarithm term is ideal because, in the beginning it favors exploration overall because of high slope, but then as all actions are tried, it flattens out and favors exploitation.

Therefore, UCB is used to approach the true value of an action in a more 'systematic' way than epsilon-greedy.

Problem 5

Why do you think in Gradient Bandit Algorithm, we defined a soft-max distribution to choose the actions from, rather than just choosing action?

Answer

In the Gradient Bandit Algorithm, we are looking to to create a **numerical preference** for each action. The ideal way to do this is by using the *soft-max distribution*, or the Gibbs/Boltzmann distribution. This is done by exponentiating the action-value function, and then normalizing it. In the form of a formula, this is,

$$\pi_t(a) \doteq \frac{e^{H_t(a)}}{\sum_{b=1}^{k} e^{H_t(b)}}$$

which is the probability of choosing action a at time-step t.

The reason we want to create a numerical preference for each action is because we want a snapshot in the current time-step of likely we are to select each action. This gives us a high degree of predictability in our method. This is a big improvement over the epsilon-greedy method, where we have no idea how likely we are to select each action (because with epsilon probability, it is random, and just chooses an action).

Problem 6

Read the article below and summarize what you learned in a paragraph:
Introduction to Multi-Armed Bandits with Applications in Digital Advertising

Answer

This article shows how the problem of showing an optimal ad to a user can be modeled as a multi-armed bandit problem. Here, an 'optimal ad' is one that maximizes the click-through rate (CTR).

First, we are shown how we can use the greedy-epsilon method to solve this problem. The true probability of a user-click is modeled as 1 trial of a binomial distribution, with the probability of success being the CTR, a number chosen arbitrarily. An array of estimated rewards is maintained, which the article calls 'emperical CTRs'. The greedy-epsilon method is used to choose the ad based on an array of weights, where the best ad has a weight of $1 - \epsilon$, and the rest have a weight of $\frac{\epsilon}{1-K}$, where the number of ads is K. This experiment set up was run 10,000 times and the results were plotted in two graphs — how the emperical CTR changed over the course of the runs, and the % of chosen actions. The analysis of the results showed that the greedy-epsilon method found the second most effective ad, and stuck with it for a while, but eventually found the most effective ad.

The second method of modeling the problem we are shown is the Thompson sampling method. Here, Bayesian beliefs are used to model the probability of a user clicking on an ad. Therefore, instead of using an array of weights like in the greedy-epsilon method, we use a beta distribution to choose the ad. As more clicks are recorded, the alpha and beta arrays for each ad is updated. If the chosen ad was clicked, then we update the alpha array, and if it was not clicked, we update the beta array. The results of this experiment were also plotted in the same way as the greedy-epsilon approach. The analysis of the results showed that the Thompson sampling method found the most effective ad much faster than the greedy-epsilon method.

In the final part of the article, a concept of comparing the two methods is introduced: *regret*. Regret is defined as the difference between the reward of the optimal action and the reward of the chosen action. Via a definitive plot, the regret for the greedy-epsilon method was found to be much higher than the regret for the Thompson sampling method. Therefore, the Thompson sampling method is the better method for this problem.