EE3001 Machine Learning **Fall 2021**

Lecture 19. Multi-armed Bandits

Lecturer: Jie Wang Date: Dec 15, 2021

The most important feature distinguishing reinforcement learning from other types of learning is that it uses training information that **evaluates** the actions taken rather than **instructs** by giving correct actions. This is what creates the need for **active exploration**, for an explicit search for good behavior.

Richard S. Sutton & Andrew G. Barto

The major reference of this lecture is [1].

1 The Problem Setting

Suppose that we are faced with a slot machine with K levers. After you select one of the K levers, you receive a numerical reward from a **stationary** probability distribution that depends on the lever you selected. If you are allowed to repeatedly select one of the levers, how to maximize the expected total reward in the long run?

This is the so-called K-armed bandit problem. We introduce some notations as follows.

- $\mathcal{A} = \{a_1, a_2, \ldots, a_K\}$: the set of available actions;
- $A_t \in \mathcal{A}$: the selected action at time t;
- $R_t \in \mathbb{R}$: the reward received at time t;
- $q_*(a_k) \in \mathbb{R}$: action value, i.e., the **unknown** expected reward by performing a_k;
- $\pi_t(a_k) \to [0,1]$: the probability of selecting actions at time t.

For notational convenience, let

- $q_* = (q_*(a_1), \ldots, q_*(a_K))^\top$;
- $\pi_t = (\pi_t(a_1), \pi_t(a_2), \ldots, \pi_t(a_K))^\top$;
- $\Delta_n = \{\mathbf{x} \in \mathbb{R}^n : x_i \in [0,1], i = 1, 2, \ldots, n, \sum_{i=1}^n x_i = 1\}$.

Cleary, we have $\pi_t \in \Delta_K$, $t = 1, 2, \ldots$.

We aim to maximize the expectation of the reward $\mathbb{E}[R_t]$. In particular, by noting that

$$\mathbb{E}[R_t] = \sum_{k=1}^{K} \Pr(A_t = a_k)\mathbb{E}[R_t | A_t = a_k] = \sum_{k=1}^{K} \pi_t(a_k) q_*(a_k), \tag{1}$$

we would like to solve the problem as follows

$$\pi_* = \underset{\pi_t \in \Delta_K}{\mathbf{argmax}}\ \mathbb{E}[R_t] = \underset{\pi_t \in \Delta_K}{\mathbf{argmax}} \sum_{k=1}^{K} \pi_t(a_k) q_*(a_k). \tag{2}$$

Clearly, if q_* is given, we can easily find $\pi_* \in \Delta^K$ that solves (2). However, as aforementioned, the expected reward by performing selected action $q_*(\cdot)$ is unknown.

Though we do not know the exact values of q_*, we can have reasonable estimates. Let $Q_t(a)$ be the estimated value of $q_*(a)$ at time step t. Then, at any time step, we can find at least one action with the greatest estimated value. Therefore, we can these the **greedy** actions. Selecting one of these greedy actions is equivalent to **exploiting** your **current knowledge** of the actions' values q_*. However, your current knowledge Q_t may not well approximates q_*, selecting one of the nongreedy actions may be reasonable as well, and we say you are **exploring**, as this enables you to improve your estimates of the action values. How to balance the exploitation and exploration remains a challenge in RL related topics and applications.

2 Action-value Methods

One natural definition of the estimates $Q_t(a)$, $a \in \mathcal{A}$ is by averaging the rewards received up to the current time step:

$$Q_t(a) := \frac{\text{sum of rewards when } a \text{ taken prior to } t}{\text{number of times } a \text{ taken prior to } t} = \frac{\sum_{i=1}^{t-1} R_i \cdot \mathbb{1}_{A_i=a}}{\sum_{i=1}^{t-1} \mathbb{1}_{A_i=a}},$$

where $\mathbb{1}_{\text{predicate}}$ denotes the indicator function that is 1 if predicate is true and 0 otherwise. We can initialize Q_0 as some default value, such as 0.

The greedy action selection method is simply to select

$$A_t = \operatorname*{\mathbf{argmax}}_a Q_t(a). \tag{3}$$

That is, we always exploit our current knowledge and never explore. A simple method to balance exploitation and exploration is the so-called ϵ-greedy method. Specifically, we select the greedy action according to (3) most of the time, but with small probability ϵ, we random select an action from all the available actions.

Algorithm 1 A simple bandit algorithm

```
1:  for $a = 1, \dots, K$ do
2:      $Q(a) \leftarrow 0$
3:      $N(a) \leftarrow 0$
4:  end for
5:  loop
6:      $A \leftarrow \begin{cases} \mathbf{argmax}_a\, Q(a), & \text{with probability } 1-\epsilon \\ \text{a random action}, & \text{with probability } \epsilon \end{cases}$
7:      $R \leftarrow$ bandit$(A)$
8:      $N(A) \leftarrow N(A) + 1$
9:      $Q(A) \leftarrow Q(A) + \frac{1}{N(A)}(R - Q(A))$
10: end loop
```

3 The Preference Function

Notice that, the problem in (2) is a constrained optimization problem, as $\pi_t \in \Delta_K$. Transforming a constrained optimization problem to an unconstrained one often leads to an easier problem.

A commonly used approach for problem (2) is to introduce a set of preference functions $H_t(a_k)$, $k = 1, 2, \ldots, K$, such that

$$\pi_t(a_k) = \frac{e^{H_t(a_k)}}{\sum_{j=1}^K e^{H_t(a_j)}}. \tag{4}$$

The distribution in (4) is the so-called **soft-max** distribution, also known as the Gibbs or Boltzmann distribution. Let $H_t = (H_t(a_1), H_t(a_2), \ldots, H_t(a_K))^\top$. Plugging Eq. (4) into (2) leads to

$$\max_{\pi_t \in \Delta_K} \mathbb{E}[R_t] = \max_{\pi_t \in \Delta_K} \sum_{k=1}^K \pi_t(a_k) q_*(a_k) = \max_{H_t \in \mathbb{R}^K} \sum_{k=1}^K \frac{e^{H_t(a_k)}}{\sum_{j=1}^K e^{H_t(a_j)}} q_*(a_k). \tag{5}$$

Question 1. Is problem (2) equivalent to (5)? In particular, for any $\pi_* \in \Delta_K$—that is, the optimal solution to problem (2)—can we find a H_t^* such that Eq. (4) holds for π_* and H_t^*? Moreover, is this H_t^* unique?

4 The Gradient Bandit Algorithm

To solve the problem in (5), we can use the gradient ascent algorithm to update H_t, i.e.,

$$H_{t+1}(a_k) = H_t(a_k) + \alpha \frac{\partial \mathbb{E}[R_t]}{\partial H_t(a_k)} = H_t(a_k) + \alpha \sum_{i=1}^K \frac{\partial \mathbb{E}[R_t]}{\partial \pi_t(a_i)} \frac{\partial \pi_t(a_i)}{\partial H_t(a_k)}. \tag{6}$$

Simple calculations lead to

$$\frac{\partial \mathbb{E}[(R_t)]}{\partial \pi_t(a_i)} = q_*(a_i), \tag{7}$$

$$\frac{\partial \pi_t(a_i)}{\partial H_t(a_k)} = \begin{cases} \dfrac{-e^{H_t(a_i)} e^{H_t(a_k)}}{(\sum_{j=1}^K e^{H_t(a_j)})^2}, & \text{if } i \neq k, \\ \dfrac{e^{H_t(a_k)}(\sum_{j=1}^K e^{H_t(a_j)}) - e^{H_t(a_k)} e^{H_t(a_k)}}{(\sum_{j=1}^K e^{H_t(a_j)})^2}, & \text{if } i = k. \end{cases} \tag{8}$$

Indeed, we can simplify $\partial \pi_t(a_i)/\partial H_t(a_k)$ in a unified form as follows.

$$\frac{\partial \pi_t(a_i)}{\partial H_t(a_k)} = \frac{\mathbb{1}_{i=k} e^{H_t(a_i)}(\sum_{j=1}^K e^{H_t(a_j)}) - e^{H_t(a_i)} e^{H_t(a_k)}}{(\sum_{j=1}^K e^{H_t(a_j)})^2} = \mathbb{1}_{i=k}\pi_t(a_i) - \pi_t(a_i)\pi_t(a_k) \tag{9}$$

$$= \pi_t(a_i)(\mathbb{1}_{i=k} - \pi_t(a_k)). \tag{10}$$

Question 2. The Jacobian matrix is

$$\frac{\partial \pi_t}{\partial H_t} = \begin{pmatrix} \pi_t(a_1)(1-\pi_t(a_1)) & -\pi_t(a_1)\pi_t(a_2) & \cdots & -\pi_t(a_1)\pi_t(a_K) \\ -\pi_t(a_2)\pi_t(a_1) & \pi_t(a_2)(1-\pi_t(a_2)) & \cdots & -\pi_t(a_2)\pi_t(a_K) \\ \vdots & \vdots & \ddots & \vdots \\ -\pi_t(a_K)\pi_t(a_1) & -\pi_t(a_K)\pi_t(a_2) & \cdots & \pi_t(a_K)(1-\pi_t(a_K)) \end{pmatrix}.$$

What is the rank of the matrix $\partial \pi_t/\partial H_t$? What does this imply?

Plugging Eq. (9) and Eq. (7) into Eq. (6) leads to

$$\begin{aligned}H_{t+1}(a_k) =& H_t(a_k) + \alpha \sum_{i=1}^{K} q_*(a_i)\pi_t(a_i)(\mathbb{1}_{i=k} - \pi_t(a_k))\\ =& H_t(a_k) + \alpha \mathbb{E}[q_*(A_t)(\mathbb{1}_{A_t=a_k} - \pi_t(a_k))]\\ =& H_t(a_k) + \alpha \mathbb{E}[(R_t - \bar{R}_t)(\mathbb{1}_{A_t=a_k} - \pi_t(a_k))].\end{aligned}$$

A stochastic version of the above update rule is

$$H_{t+1}(a_k) = H_t(a_k) + \alpha(R_t - \bar{R}_t)(\mathbb{1}_{A_t=a_k} - \pi_t(a_k)),\ k = 1, \ldots, K.$$

References

[1] R. S. Sutton and A. G. Barto. *Reinforcement Learning An Introduction, 2nd.* The MIT Press, 2018.