Chapter 2

David Ding

October 3, 2024

1 Problem Introduced

Suppose we have k-bandits, each of them has a hidden value behinds it. We can only choose one bandit each time and get a reward from the bandit we choose. The goal is to quickly converges to the bandit that has the highest value.

We denote the action that we choose at time t as A_t, and the reward we get from the action as R_t. The value for this arbitary step is $q_*(A_t)$, and the expected value that the bandit a is chosen is

$$q_*(a) = \mathrm{E}[R_t|A_t = a]$$

The solution to this problem is to use the greedy algorithm, which always choose the bandit that has the highest value. However, this algorithm will not give the best choice when the observation at initial conditions haven't done enough to figure out the actual value of each bandit. This raise the most important problem in this chapter, the **exploration-exploitation** diplomacy.Should we continune to update the value estimation at time t denotes as $Q_t(a)$, that is to randomly choose a bandit for the update of its value, or just stick to our past $t-1$'s experience and choose the bandit that is estimated to be the best at that time. And this chapter give out some tricks or special solution to tackle with this problem.

- **ϵ - greedy algorithm**

  The ϵ - greedy algorithm helps to add some **randomness** to the greedy algorithm. The higher the ϵ is, the more easily it will reach the optimal bandit estimate, but the randomness influence its optiamal rate at the end of the training, letting the expected correctness to be lower than others. While using a small ϵ would have completely different result. So sometimes for stationary condition, the **variable** ϵ could be introduced in order to reduce the effect of the randomness.

- **Optimistic initial values**

  - Large Positive Initial Value

    By setting the initial value of each bandit to **higher** than the possible value of bandits, every exploration will definitely reduce the estimate of the bandit, "upsetting" the selection for this bandit and turn to other less frequently selected bandits where the recent value is more closely related to the intial value.

  - An Unbiased Constant-Stepsize Trick

    As we can see, the setting of our initial value directly influence our estimate of the value for each bandit. So setting the $\beta_t \in [0, 1]$(the percentage of how the new observed value would influence the update of the estimate of the bandit at time t) to be a dereasing value(from 1 to α) would be a good choice. This help to prevent the **Initial Bias** for the estimation while keeps the property of expotential recency descendent of weights.

- **Upper confidence bound**

  The Upper confidence bound(UCB) is a method that take the account of uncertainty from the lack of enough observations and the compensation of the more explore due to the increase of time. By adding a term to the estimated action value to reach the **plausible upper bound**, exploration would be encouraged.

- **Gradient bandit algorithm**

  This approach stands somewhat further than the methods discussed in this chapter, that it doesn't relied on the estimation of action value, instead, a numerical **preference** denotes as $H_t(a) \in \mathbb{R}$ is introduced, and we focus on the relative preference over another action that really matters in our decision making. In order to transform this kind of preference into our choice from the bandits, the *Softmax* Method is used for convenience.

- **Associative search**

  Associative search is actually a more complicated aspect that takes into account of different situation, in which the best bandit might be different meaning a variable policy. Thus the agent should be able to sense the environment and make the right decision.

2 The ϵ - greedy algorithm

2.1 Action-Value Methods

The Action-Value method is the basis of the ϵ - greedy method. As we know, one of the key step in a reinforcement learning is to identify the estimated value for a certain action. The Action-Value method is to estimate the value of each action by the average of the rewards that we get from this action.

$$Q_t(a) = \frac{\sum_{i=1}^{t-1} 1_{A_i=a} \cdot R_i}{\sum_{i=1}^{t-1} 1_{A_i=a}} \tag{1}$$

Then the choice of bandit at time t is $A_t = \arg\max_a Q_t(a)$.

For the ease of more efficient computation and the logic of updating of the estimate rather than calculating the average again and again every time, we would like to use the **Incremental Method**.

$$\begin{aligned} Q_{n+1} &= \frac{1}{n}\sum_{i=1}^{n} R_i \\ &= \frac{1}{n}R_n + \frac{1}{n}\sum_{i=1}^{n-1} R_i \\ &= \frac{1}{n}R_n + \frac{n-1}{n}Q_n \\ &= Q_n + \frac{1}{n}[R_n - Q_n] \end{aligned} \tag{2}$$

we denote the stepsize variable as $\alpha(t)$, in the sample mean case, $\alpha_t = \frac{1}{n}$.

2.2 Constant Stepsize

For the need to set the stepsize larger in the long run, we could use a constant stepsize that ensure the update is equal no matter t is early or not. The update formula is:

$$Q_{n+1} = Q_n + \alpha[R_n - Q_n]$$

Writing this formula a little more, we could derive the relationship with each observation:

$$\begin{aligned} Q_{n+1} &= Q_n + \alpha[R_n - Q_n] \\ &= \alpha R_n + (1-\alpha)Q_n \\ &= \alpha R_n + (1-\alpha)[(1-\alpha)Q_{n-1} + \alpha R_{n-1}] \\ &= (1-\alpha)^n Q_1 + \sum_{i=1}^{n} \alpha(1-\alpha)^{n-i} R_i \end{aligned} \tag{3}$$

By adding the weights over Q_1 and R_1 together, it's easy to show that the sum of the weights is 1[hint: adding the first term and the second sum in pairs]

When the stepsize is not constant, then the stepsize is variable with time t, then we could rewrite the formula into:

$$\begin{aligned} Q_{n+1} &= \alpha_n R_n + [1-\alpha_n]Q_n \\ &= \prod_{i=1}^{n}(1-\alpha_i)Q_1 + \sum_{i=1}^{n}\alpha_i \prod_{j=i+1}^{n}(1-\alpha_j)R_i \end{aligned} \tag{4}$$

2.3 Algorithm

A simple bandit algorithm

Initialize, for $a = 1$ to k:
$Q(a) \leftarrow 0$
$N(a) \leftarrow 0$

Loop:
$A \leftarrow \begin{cases} \arg\max_a Q(a) & \text{with probability } 1-\epsilon \text{ (breaking ties randomly)} \\ \text{random action} & \text{with probability } \epsilon \end{cases}$
$R \leftarrow \text{bandit}(A)$
$N(A) \leftarrow N(A) + 1$
$Q(A) \leftarrow Q(A) + \frac{1}{N(A)}[R - Q(A)]$

3 Optimistic initial values

3.1 Large Positive Initial Value

- Setting a high initial action value encourages exploration by making all bandits initially attractive.
- Not suitable for nonstationary problems, as the initial value won't drive the exploration permanently.

3.2 An Unbiased Constant-Stepsize Trick

This trick combines the advantages of the sample-average methods that it will not produce a **Initial Bias** and the constant stepsize methods that it gives **more weight** on newly observed rewards for certain actions.

The stepsize is defined as:

$$\beta_n = \frac{\alpha}{\overline{o}_n} \tag{5}$$

Where the α is the stepsize we want in the long turn, and by setting the $\overline{o}_n$ to increase over time, we will get a bigα when t is small, and the β_1 equals to 1, which means the initial value is completely disgarded and the inital update of estimates will focus more on recency. The update of $\overline{o}_n$ is defined as:

$$\overline{o}_n = \overline{o}_{n-1} + \alpha[1 - \overline{o}_{n-1}] \tag{6}$$

4 Upper confidence bound

The UCB upper confidence bound is simple in practice when dealing with our multi-bandit problem, that the formula could be easily written as:

$$UCB_t(a) = R_t(a) + c\sqrt{\frac{\ln t}{N_t(a)}} \tag{7}$$

Where c is the constant that balance the exploration and exploitation trade-off, $\sqrt{\ln T}$ encourage explorations along the increasing time, and $\sqrt{\frac{1}{N_t(a)}}$ would prompt unexplored bandits to have larger potential.

5 Gradient bandit algorithm

5.1 Gradient Method Introduced

As we mentioned in the first section, we would like to adjust the numerical preference on each of the bandit in order to get the largest expected rewards. The preference for each of the bandit at time t is denoted as $H_t(a)$, and we transform the preference for a certain bandit into a absolute probability of choosing it considered the relative preference with the other bandits, the probability is denoted as $\pi_t(a)$, with the expression:

$$\pi_t(a) = \frac{\exp^{H_t(a)}}{\sum_x \exp^{H_t(x)}} \tag{8}$$

Using this prior estimated probability, we randomly choose one bandit from this distribution, and get the reward R_t, and we also calculate the average reward $\overline{R_t}$ from time $1-t, [R_1 = \overline{R_1}]$.

As we randomly sample a action A_t according to the preference at time t, this serves as a **Stochastic Gradient Descendent**, aka the expectation of every update is exactly the "Gradient descendent"[1] of the Rewards. The update formula when we choose Action A_t with rewards R_t and the average rewards $\overline{R_t}$ is then:

$$\begin{cases} H_t(A_t) = H_t(A_t) + \alpha(R_t - \overline{R_t})(1 - \pi_t(A_t)) \ \text{ for } a \ = \ A_t \\ H_t(a) = H_t(a) - \alpha(R_t - \overline{R_t})\pi_t(A_t) \ \text{ for } a \ \neq \ A_t \end{cases} \tag{9}$$

Where the α is the rate of learning, while $\overline{R_t}$ serve as a baseline.

5.2 Mathmetical Model for Gradient Bandit Method

The general idea for the gradient method is to obtain the highest expectation of rewards $E[R_t]$ under parameters $H_t(a)$. By using the partial derivative of the expectation value towards different $H_t(a)$, then it writes:

$$H_{t+1}(a) = H_t(a) + \alpha \frac{\partial E[R_t]}{\partial H_t(a)} \tag{10}$$

This formula is not pratical yet as we couldn't calculate the exact partial derivative for a expectational value, where the $E[R_t]$ is:

$$E[R_t] = \sum_x \pi_t(x) q^*(x) \tag{11}$$

Substituting into the former formula, and use the property that $\sum_x \frac{\partial \pi_x}{\partial H_t(a)} = 0$, we get:

$$\frac{\partial E[R_t]}{\partial H_t(a)} = \sum_x (q^*(x) - B_t) \frac{\partial \pi_t(x)}{\partial H_t(a)} \tag{12}$$

Where the B_t is a action-invariant value that influence the absolute change of each action value while does not change the **relative**update in $H_t(a)$. For faster convergence, we choose this B_t as $\overline{R_t}$

$$\begin{aligned} \sum_x (q^*(x) - \overline{R_t}) \frac{\partial \pi_t(x)}{\partial H_t(a)} &= \sum_x \pi_t(x)(q^*(x) - \overline{R_t}) \frac{\partial \pi_t(x)}{\partial H_t(a)} \cdot \frac{1}{\pi_t(x)} \\ &= E[(R_t - \overline{R_t}) \frac{\partial \pi_t(A_t)}{H_t(a)} \cdot \frac{1}{\pi_t(A_t)}] \end{aligned} \tag{13}$$

A_t is a random variable that is chosen according to the probability distribution of π_t, and the rewards R_t satisfy $E[R_t|A_t] = q^*(A_t)$. Using the chain rules on the derivative $\frac{\partial \pi_t(x)}{\partial H_t(a)}$, we get:

$$\begin{cases} \frac{\partial \pi_t(A_t)}{\partial H_t(a)} = \pi_t(A_t)(1 - \pi_t(a)) \ \text{ for } A_t \ = \ a \\ \frac{\partial \pi_t(A_t)}{\partial H_t(a)} = -\pi_t(A_t)\pi_t(a) \ \text{ for } A_t \ \neq \ a \end{cases} \tag{14}$$

[1] Here it actually is not descendent but to follow the direction of the gradient in order to maximize the expectation of rewards $E[R_t]$.

$$E[(R_t - \overline{R_t})\frac{\partial \pi_t(A_t)}{H_t(a)} \cdot \frac{1}{\pi_t(A_t)}] = E[(R_t - \overline{R_t})(1_{A_t=a} - \pi(a))] \tag{15}$$

Thus, the final update formula could be written as:

$$H_{t+1}(a) = H_t(a) + \alpha E[(R_t - \overline{R_t})(1_{A_t=a} - \pi_t(a))] \tag{16}$$

Again keep in mind that A_t is the random variable that is being sampled under π_t. Comparing formula (9), the **expectation** of that update is exactly the formula derived using gradient method.

5.3 Algorithm

Gradient Bandit Algorithm

Initialize, for $a = 1$ to k:

$H(a) \leftarrow 0$
$\overline{R} \leftarrow 0$
$N \leftarrow 0$
$\alpha \leftarrow \alpha_0$

Loop:

$\pi(a) = \frac{\exp^{(H(a))}}{\sum_x \exp^{H(x)}}$

$A \leftarrow \begin{cases} \text{draw from distribution } \pi(a) & \text{with probability} 1-\epsilon \\ \text{random action} & \text{with probability } \epsilon \end{cases}$

$R \leftarrow \text{bandit}(A)$
$N \leftarrow N + 1$
$\overline{R} \leftarrow \overline{R} + \frac{1}{N}(R - \overline{R})$

$H(a) \leftarrow \begin{cases} H(a) + \alpha * (R - \overline{R})(1 - \pi(a)), & a = A \\ H(a) - \alpha * (R - \overline{R})\pi(a), & a \neq A \end{cases}$