WiDS QRL - Bandits

Nilabha Saha

January 2023

1 k-armed Bandit Problem

Consider the following learning problem. You are faced repeatedly with a choice among k different options, or actions. After each choice you receive a numerical reward chosen from a stationary probability distribution that depends on the action you selected. Your objective is to maximize the expected total reward over some time period, for example, over 1000 action selections, or time steps. This is the k-armed bandit problem.
In our k-armed bandit problem, each of the k actions has an expected or mean reward given that that action is selected; let us call this the value of that action. We denote the action selected on time step t as A_t, and the corresponding reward as R_t. The value then of an arbitrary action a, denoted $q_*(a)$, is the expected reward given that a is selected:

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

If you knew the value of each action, then it would be trivial to solve the k-armed bandit problem: you would always select the action with highest value. We assume that you do not know the action values with certainty, although you may have estimates. We denote the estimated value of action a at time step t as $Q_t(a)$. We would like $Q_t(a)$ to be close to $q_*(a)$.

2 Action-value Methods

We look closely at methods for estimating the values of actions and for using the estimates to make action selection decisions, which we collectively call action-value methods. Recall that the true value of an action is the mean reward when that action is selected. One natural way to estimate this is by averaging the rewards actually received:

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

If the denominator is zero, then we instead define $Q_t(a)$ as some default value, such as 0. As the denominator goes to infinity, by the law of large numbers,

$Q_t(a)$ converges to $q_*(a)$. We call this the sample-average method for estimating action values because each estimate is an average of the sample of relevant rewards.
The simplest action selection rule is to select one of the actions with the highest estimated value, that is, one of the greedy actions as defined in the previous section. If there is more than one greedy action, then a selection is made among them in some arbitrary way, perhaps randomly. We write this greedy action selection method as

$$A_t = \underset{a}{\text{argmax}} Q_t(a)$$

Greedy action selection always exploits current knowledge to maximize immediate reward; it spends no time at all sampling apparently inferior actions to see if they might really be better. A simple alternative is to behave greedily most of the time, but every once in a while, say with small probability ϵ, instead select randomly from among all the actions with equal probability, independently of the action-value estimates. We call methods using this near-greedy action selection rule ϵ-greedy methods. An advantage of these methods is that, in the limit as the number of steps increases, every action will be sampled an infinite number of times, thus ensuring that all the $Q_t(a)$ converge to $q_*(a)$. This of course implies that the probability of selecting the optimal action converges to greater than $1 - \epsilon$, that is, to near certainty. These are just asymptotic guarantees, however, and say little about the practical effectiveness of the methods.

3 Incremental Implementation

To simplify notation we concentrate on a single action. Let R_i now denote the reward received after the ith selection of this action, and let Q_n denote the estimate of its action value after it has been selected $n - 1$ times, which we can now write simply as

$$Q_n = \frac{R_1 + R_2 + \cdots + R_{n-1}}{n-1}$$

An incremental way to compute the above would be:

$$Q_{n+1} = Q_n + \frac{1}{n}\left[R_n - Q_n\right]$$

The above update rule is one that occurs frequently anf its general form is

$$\text{NewEstimate} \leftarrow \text{OldEstimate} + \text{StepSize}\left[\text{Target} - \text{OldEstimate}\right]$$

4 Tracking a Nonstationary Problem

The averaging methods discussed so far are appropriate for stationary bandit problems, that is, for bandit problems in which the reward probabilities do not change over time. As noted earlier, we often encounter reinforcement learning problems that are effectively nonstationary. In such cases it makes sense to

give more weight to recent rewards than to long-past rewards. One of the most popular ways of doing this is to use a constant step-size parameter.
Sometimes it is convenient to vary the step-size parameter from step to step. Let $\alpha_n(a)$ denote the step-size parameter used to process the reward received after the nth selection of action a. As we have noted, the choice $\alpha_n(a) = \frac{1}{n}$ results in the sample-average method, which is guaranteed to converge to the true action values by the law of large numbers. But of course convergence is not guaranteed for all choices of the sequence $\{\alpha_n(a)\}$.A well-known result in stochastic approximation theory gives us the conditions required to assure convergence with probability 1:

$$\sum_{n=1}^{\infty} \alpha_n(a) = \infty \quad \sum_{n=1}^{\infty} \alpha_n^2(a) < \infty$$

The first condition is required to guarantee that the steps are large enough to eventually overcome any initial conditions or random fluctuations. The second condition guarantees that eventually the steps become small enough to assure convergence.

5 Upper-Confidence-Bound Action Selection

Exploration is needed because there is always uncertainty about the accuracy of the action-value estimates. The greedy actions are those that look best at present, but some of the other actions may actually be better. ϵ-greedy action selection forces the non-greedy actions to be tried, but indiscriminately, with no preference for those that are nearly greedy or particularly uncertain. It would be better to select among the non-greedy actions according to their potential for actually being optimal, taking into account both how close their estimates are to being maximal and the uncertainties in those estimates. One effective way of doing this is to select actions according to

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

where $N_t(a)$ denotes the number of times that action a has been selected prior to time t, and the number $c > 0$ controls the degree of exploration. If $N_t(a) = 0$, then a is considered to be a maximising action.

6 Gradient Bandit Algorithms

we have considered methods that estimate action values and use those estimates to select actions. This is often a good approach, but it is not the only one possible. In this section we consider learning a numerical preference for each action a, which we denote $H_t(a)$. The larger the preference, the more often that action is taken, but the preference has no interpretation in terms of reward. Only the relative preference of one action over another is important; if we add 1000 to all the action preferences there is no effect on the action probabilities, which are determined according to a soft-max distribution (i.e., Gibbs or Boltzmann distribution) as follows:

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

Initially all action preferences are the same so that all actions have an equal probability of being selected.
There is a natural learning algorithm for this setting based on the idea of stochastic gradient ascent. On each step, after selecting action A_t and receiving the reward R_t,the action preferences are updated by:

$$H_{t+1}(A_t) = H_t(A_t) + \alpha(R_t - \bar{R}_t)(1 - \pi_t(A_t))$$

and

$$H_{t+1}(a) = H_t(a) - \alpha(R_t - \bar{R}_t \pi_t(a)$$

for all $a \neq A_t$, where $\alpha > 0$ is a step-size parameter, and $\bar{R}_t \in \mathbb{R}$ is the average of all rewards up through and including time t, which can be computed incrementally. The $\bar{R}_t$ term serves as a baseline with which the reward is compared. If the reward is higher than the baseline, then the probability of taking A_t in the future is increased, and if the reward is below baseline, then probability is decreased. The non-selected actions move in the opposite direction.