Date created: 2025-07-19-Sat

2.8 Gradient Bandit Algorithms

Preference-based action selection

So far, we have looked at methods estimate action values and select actions based on those estimates. Although this is a good apporach, there are other possible apporaches. Here, we consider learning a numerical *preference*, $H_t(a)$, for each action a. The higher the preference, the more likely the action is to be selected. But only relative preference is considered, i.e. if we add 1000 to all preferences, the resultant probabilities for selecting each action does not change. These probabilities are determined according to a *soft-max distribution*:

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

Here, we have introduced a useful notation, $\pi_t(a)$, for the probability of taking action a at time t. If initially all preferences are the same (e.g. $H_1(a) = 0$), then the probabilities are equal.

Learning preferences with stochastic gradient assent

A natural learning algorithm for preferences based on stochastic gradient assent: on each step, after selecting action A_t and receiving R_t, we update the preferences as follows:

$$\begin{aligned} H_{t+1}(A_t) &:= H_t(A_t) + \alpha(R_t - \bar{R}_t)(1 - \pi_t(A_t)), && \text{and} \\ H_{t+1}(a) &:= H_t(a) + \alpha(R_t - \bar{R}_t)\pi_t(a), && \forall a \neq A_t, \end{aligned}$$

where $\alpha > 0$ is the step-size parameter, and $\bar{R}_t \in \mathbb{R}$ is the average of all rewards up through and including time t. The $\bar{R}_t$ term serves as a baseline to which rewards are compared, if $R_t > \bar{R}_t$ then we increase the preference for the selected action, otherwise we decrease it. The preferences for non-selected actions move in the opposite direction.

Effect of baseline R_t

Figure 2.5 shows results for gradient bandit algorithms on a 10-armed testbed with $q_*(a) \sim N(4, 1)$ with and without the baseline R_t. As shown, the baseline ensures that shifting up the rewards have no effect on the algorithm, whereas setting $R_t = 0$ significantly degrades performance.

Exercises

2.9: Sigmoid

Show that in the case of two actions, the soft-max distribution is the same as that given by the logistic, or sigmoid, function often used in statistics and artificial neural networks.

Solution Denote the two preference values as H_1 and H_2, corresponding to actions 1 and 2. The softmax probability of selecting each action is:

$$\pi(1) = \frac{e^{H_1}}{e^{H_1} + e^{H_2}} = \frac{1}{1 + e^{H_2 - H_1}}, \quad \pi(2) = \frac{e^{H_2}}{e^{H_1} + e^{H_2}} = \frac{1}{1 + e^{H_1 - H_2}}$$

The sigmoid or logistic function is defined as:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

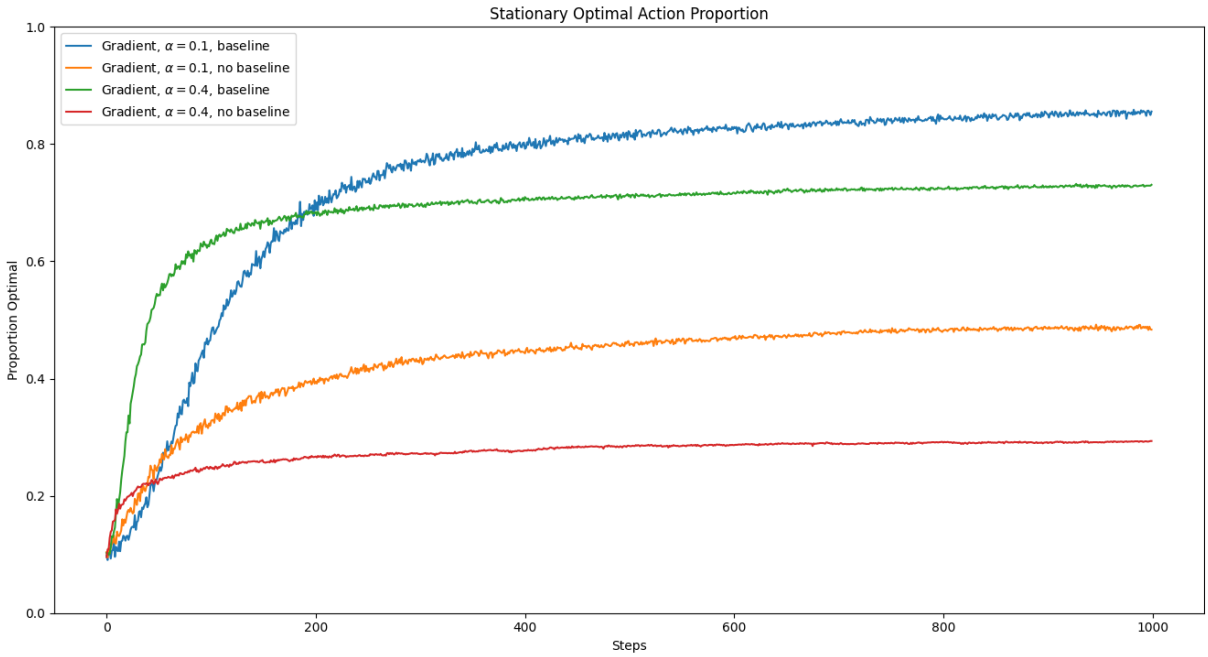

Figure 1: Gradient Bandit Algorithm on modified 10-armed testbed

If we define:

$$x = H_1 - H_2$$

Then:

$$\pi(1) = \frac{1}{1 + e^{-(H_1 - H_2)}} = \sigma(H_1 - H_2), \qquad \pi(2) = \frac{1}{1 + e^{-(H_2 - H_1)}} = \sigma(H_2 - H_1)$$

In the two-action case, the softmax distribution is equivalent to applying a sigmoid to the difference in preferences. This is exactly how binary logistic regression and binary classification in neural networks work.