Lecture 3: Bandits

1 Game

Learning objectives, overview By the end of this class, you will be able to identify bandit and contextual bandit problems and implement effective solutions to these problems. You will understand *why* treating bandit problems as RL problems is a bad idea.

2 Bandit problems.

If you want to read more about bandits, check out this book.

Definition. actions, stochastic rewards.

Examples

- Choosing which restaurant to go to
- Choosing which ad to display to a user (contextual bandit)
- Choosing which flavor of ice cream to use
- Voting.
- Prospecting for cobalt mines
- A/B testing for websites. Do users like a button that says "buy now" or "click here to purchase"? Asymptotically, you'd like to display the button that has highest sales.
- design of clinical trials. Each patient is assigned to one "arm" of the trial. Asymptotically, all patients should be assigned to the best treatment.

Objective.

$$\max \mathbb{E}[r(a)]. \tag{1}$$

This is an expectation over a few different things. It's an expectation over the rewards themselves, which are stochastic.

The **input** is a dataset $\{(a_1, r_1 \sim r(a_1)), (a_2, r_2 \sim r(a_2)), \cdots\}$. The **output** is a policy $\pi(a_t)$; often this policy will be deterministic, in which case it's sufficient to say that the output of the bandit algorithm is a single action, a_t.

We'll explore three strategies for solving the bandit problem (that is, for mapping this input to this output):

1. Greedy
2. ϵ-greedy
3. Upper confidence bound (UCB)

2.1 Greedy strategy

Take the empirical mean, the average reward at arm a when it's been taken before.[1]

$$\hat{r}(a) = \frac{\text{sum of rewards when you took action } a \text{ before}}{\text{num. times taken action } a}$$

[1] I'll use the "hat" symbol on $\hat{r}$ to denote that this is an estimate of the true quantity, $\mathbb{E}[r(a)]$.

At each time step t, take the arm with greatest estimated reward:

$$\arg\max_a \hat{r}(a). \tag{2}$$

Q: What is good about this approach? What is bad about this approach? When will this approach fail?

Limitations of the greedy approach.

- No mechanism for exploration
- Can get stuck in a local optimum.

2.2 ϵ-Greedy

With probability $1-\epsilon$, take the action $\arg\max_a \hat{r}(a)$. With probability ϵ, choose the action randomly.

Comments:

- This method does explicitly do some exploration. It won't get stuck in local optima forever.
- You have to tune ϵ. Best choice is to decay ϵ over time. But over what horizon? Is there a way to do this automatically?
- In the ϵ fraction of the time when actions are chosen randomly, all actions are chosen uniformly. But intuitively, it seems like you should sample the more promising (non-optimal) actions more frequently.
- If there's a bad action that you've taken several times, intuitively it seems like you should be able to stop taking the action. How can we codify this intuition that there's no longer any chance that some action is the best action?

2.3 Boltzmann / Softmax

How can we design a strategy so that we sample the more promising actions more frequently, but still sample the less promising actions with some frequency (on the off-chance that they are the best)?

$$\pi(a) = \textsc{SoftMax}(\hat{r}(a)) = \frac{e^{\frac{1}{\beta}\hat{r}(a)}}{\sum_{a' \in \mathcal{A}} e^{\frac{1}{\beta}\hat{r}(a')}}. \tag{3}$$

Intuition behind the temperature $\beta > 0$. For large β, we sample uniformly. For small β, we recover the greedy strategy.

Compared with ϵ-greedy, this approach samples actions proprtional to how promising they seem. However, this approach uses a fixed degree of exploration the entire time. If there's an action that you've sampled dozens of times and it's always been bad, this algorithm will nonetheless continue to sample it.

2.4 Upper Confidence Bound (UCB)

Let $N_a(t)$ be the number of times you've taken action a before time t. The upper confidence bound strategy says to take the action:

$$\arg\max_a \hat{r}(a) + \sqrt{\frac{2\log(t)}{N_a(t)}}. \tag{4}$$

Deriving UCB. Hoeffding bound: Assume $0 \leq r(a) \leq 1$.

$$P(\hat{r}(a) \geq \mathbb{E}[r(a)] + \epsilon) \leq e^{-n\epsilon^2/2}. \tag{5}$$

Let's say the confidence interval is where the probability is at most δ.

$$\delta = e^{-n\epsilon^2/2} \implies \epsilon = \sqrt{\frac{2}{n}\log(1/\delta)}. \tag{6}$$

OK, so this tells us that our confidence interval has size ϵ, so we should choose the arm with $\arg\max_a \hat{r}(a) + \sqrt{\frac{2}{n}\log(1/\delta)}$.

(hand-waving) But how should we choose δ? Do we want a really big confidence interval or a smaller confidence interval? Let's make this interval get smaller over time. So we'll use $\delta = 1/t$. Plugging this in gives us the desired result:

$$\arg\max_a \hat{r}(a) + \sqrt{\frac{2}{N_a(t)}\log t} \tag{7}$$

UCB is just about as good as you can do.

2.5 Comparison with MDPs

Like the MDP, we want to maximize the rewards. This is usually written as a summation. But unlike the MDP, we just consider one trial. The MDP objective is written as a sum of rewards, as is the bandit objective. But there's a subtle difference. In the RL objective, you get a observation and reward after each "action"; in the bandit setting, you just get the reward.

In both settings, there's a notion that you need to balance exploration vs exploitation. In the RL setting, this is usually thought of at the resolution of episodes: you'll try one strategy in one episode, another strategy in another episode. In the bandit setting, each individual action is a choice for whether to explore or exploit (or somewhere in between).

One way of thinking about the bandit problem is as a special case of the RL problem where $\gamma = 0$. To see this, recall that we weight the rewards at time step t by γ^t, so in a bandit problem the rewards would be weighted as $0^0, 0^1, 0^2, 0^3, \cdots = 1, 0, 0, 0, \cdots$.[2]

2.6 Contextual bandits

Sample state/context $s_0 \sim p_0(s_0)$, sample action $a_0 \sim \pi(a_0 \sim s_0)$, observe reward $r(s_0, a_0)$. To solve, estimate $r_\theta(s, a) \approx \mathbb{E}[r(s, a)]$. E.g., via regression:

$$\min_\theta \frac{1}{N} \sum_{s_0, a_0, r} (r_\theta(s_0, a_0) - r)^2. \tag{8}$$

You can then employ the exploration strategies we've already talked about today. Note that applying UCB is a bit nontrivial because it (naïvely) requires keeping counts for how often you've seen each state/action.

[2]We've used the convention that $0^0 = 1$.

2.7 Why is treating a bandit problem like an RL problem a bad idea?

3 Reducing the RL problem to a bandit problem

(We didn't end up covering this in class.)

One way to do this is to treat the policy *parameters* as the action. That is, for a policy $\pi_\theta(a \mid s)$, construct a bandit problem where the actions are $\theta \in \subseteq$ and the (stochastic) rewards are $r(\theta) = \mathbb{E}_{\pi_\theta(a|s)}[\sum_t \gamma^t r(s_t, a_t)]$. We will explore this strategy in a future homework.

After class, one student group proposed an alternative method, reducing the RL problem to a *contextual* bandit problem. The construction mirrored one from Bagnell et al. [1]. Assume that your problem as a finite horizon T. We'll start at the final time step and try to find the optimal policy, $\pi(a_T \mid s_T)$. Note that this is just a contextual bandit problem; we don't need to worry about future states because the episode is going to terminate. After finding the optimal policy for time step T, we can estimate a value function $V(s_T) \approx \mathbb{E}_{\pi(a_T|s_T)}[r(s_T)]$. Note that this depends on the final-step policy $\pi(a_T \mid s_T)$ that we have just determined. Then, we can compute the policy for time step $T-1$ as another contextual bandit, where the "rewards" are $r(a_{T-1}, s_{T-1}) = r(s_{T-1}, a_{T-1}) + \gamma V(s_T)$. Repeat. One limitation of this approach is that it assumes that you can "reset" the agent to arbitrary states; this is typically disallowed in the MDP problem.

References

[1] Bagnell, J., Kakade, S. M., Schneider, J., and Ng, A. (2003). Policy search by dynamic programming. *Advances in neural information processing systems*, 16.