Bandits: Bayesian Bandits and Thompson Sampling

Lucas Janson and Sham Kakade

CS/Stat 184: Introduction to Reinforcement Learning

Fall 2023

Today

- Feedback from last lecture
- Recap
- Bayesian bandit
- Thompson sampling

Feedback from feedback forms

1. Thank you to everyone who filled out the forms!
2.

Today

- ✓ Feedback from last lecture
- Recap
- Bayesian bandit
- Thompson sampling

Recap

- Pure greedy, pure exploration, ETC, ε-greedy achieve suboptimal regret
- UCB uses Optimism in the Face of Uncertainty (OFU) principle and achieves *optimal* rate $\tilde{O}(\sqrt{T})$ of regret
- Theory is nice, but what about in practice?

Today

- ✔ Feedback from last lecture
- ✔ Recap
- Bayesian bandit
- Thompson sampling

Bayesian bandit

A Bayesian bandit augments the bandit environment we've been working in so far with a prior distribution on the unknown reward distributions: $\pi(\nu^{(1)}, \ldots, \nu^{(K)})$

E.g., in a Bernoulli bandit, each $\nu^{(k)}$ is entirely characterized by its mean $\mu^{(k)} = \mathbb{P}_{r \sim \nu^{(k)}}(r = 1)$, so a prior on the $\nu^{(k)}$ is equivalent to a prior on the $\mu^{(k)}$

One such prior, since all the $\mu^{(k)}$ are bounded between 0 and 1, is the prior that is *Uniform* on the unit hypercube, i.e.,
$(\mu^{(1)}, \ldots, \mu^{(K)}) =: \boldsymbol{\mu} \sim \text{Uniform}([0,1]^K)$

Note that the Bernoulli bandit reduced everything unknown about the bandit system to a K-dimensional vector $\boldsymbol{\mu}$

Without the Bernoulli assumption, we may need many more dimensions to describe the possible distributions, and hence have to define a much higher-dimensional prior

Bayesian Bernoulli bandit

The really nice thing about a Bayesian bandit is that we can use Bayes rule to exactly characterize our uncertainty about the reward distributions at every time step.

Example: Bayesian Bernoulli bandit

1. At $t = 0$, how can we characterize our uncertainty about $\boldsymbol{\mu}$?
   We have no data, and the distribution of the reward distributions is simply given by the prior on the reward parameters $\boldsymbol{\mu}$:

$$\mathbb{P}(\boldsymbol{\mu}) = \pi(\boldsymbol{\mu})$$

($\mathbb{P}$ will sometimes denote a continuous density instead of a true probability, e.g., for $\boldsymbol{\mu} \sim$ Uniform$([0,1]^K)$, we would write $\mathbb{P}(\boldsymbol{\mu}) = 1_{\{0 \leq \mu^{(k)} \leq 1 \ \forall k\}}$)

Bayesian Bernoulli bandit (cont'd)

1. At $t = 0$, $\mathbb{P}(\boldsymbol{\mu}) = \pi(\boldsymbol{\mu})$
2. At $t = 1$, we have one data point $r_0 \sim$ Bernoulli$(\mu^{(a_0)})$, and the distribution of $\boldsymbol{\mu}$ gets updated via Bayes rule:

$$\mathbb{P}(\boldsymbol{\mu} \mid r_0, a_0) = \frac{\mathbb{P}(r_0, a_0 \mid \boldsymbol{\mu})\mathbb{P}(\boldsymbol{\mu})}{\mathbb{P}(r_0, a_0)} = \frac{\mathbb{P}(r_0, a_0 \mid \boldsymbol{\mu})\mathbb{P}(\boldsymbol{\mu})}{\int_{\tilde{\boldsymbol{\mu}} \in [0,1]^K} \mathbb{P}(r_0, a_0 \mid \tilde{\boldsymbol{\mu}})\mathbb{P}(\tilde{\boldsymbol{\mu}})d\tilde{\boldsymbol{\mu}}}$$

$$= \frac{\mathbb{P}(r_0 \mid a_0, \boldsymbol{\mu})\mathbb{P}(a_0 \mid \boldsymbol{\mu})\mathbb{P}(\boldsymbol{\mu})}{\int_{\tilde{\boldsymbol{\mu}} \in [0,1]^K} \mathbb{P}(r_0 \mid a_0, \tilde{\boldsymbol{\mu}})\mathbb{P}(a_0 \mid \tilde{\boldsymbol{\mu}})\mathbb{P}(\tilde{\boldsymbol{\mu}})d\tilde{\boldsymbol{\mu}}}$$

Can you see any way to simplify?

$$= \frac{\mathbb{P}(r_0 \mid a_0, \boldsymbol{\mu})\mathbb{P}(a_0)\mathbb{P}(\boldsymbol{\mu})}{\int_{\tilde{\boldsymbol{\mu}} \in [0,1]^K} \mathbb{P}(r_0 \mid a_0, \tilde{\boldsymbol{\mu}})\mathbb{P}(a_0)\mathbb{P}(\tilde{\boldsymbol{\mu}})d\tilde{\boldsymbol{\mu}}} = \frac{\mathbb{P}(r_0 \mid a_0, \boldsymbol{\mu})\mathbb{P}(\boldsymbol{\mu})}{\int_{\tilde{\boldsymbol{\mu}} \in [0,1]^K} \mathbb{P}(r_0 \mid a_0, \tilde{\boldsymbol{\mu}})\mathbb{P}(\tilde{\boldsymbol{\mu}})d\tilde{\boldsymbol{\mu}}}$$

Bayesian Bernoulli bandit (cont'd)

1. At $t = 0$, $\mathbb{P}(\boldsymbol{\mu}) = \pi(\boldsymbol{\mu})$
2. At $t = 1$, we have one data point $r_0 \sim$ Bernoulli($\mu^{(a_0)}$), and the distribution of $\boldsymbol{\mu}$ gets updated via Bayes rule:

$$\mathbb{P}(\boldsymbol{\mu} \mid r_0, a_0) = \frac{\mathbb{P}(r_0 \mid a_0, \boldsymbol{\mu})\mathbb{P}(\boldsymbol{\mu})}{\int_{\tilde{\boldsymbol{\mu}} \in [0,1]^K} \mathbb{P}(r_0 \mid a_0, \tilde{\boldsymbol{\mu}})\mathbb{P}(\tilde{\boldsymbol{\mu}}) d\tilde{\boldsymbol{\mu}}} = \frac{(\mu^{(a_0)})^{r_0}(1-\mu^{(a_0)})^{1-r_0}\pi(\boldsymbol{\mu})}{\int_{\tilde{\boldsymbol{\mu}} \in [0,1]^K} (\tilde{\mu}^{(k)})^{r_0}(1-\tilde{\mu}^{(a)})^{1-r_0}\pi(\tilde{\boldsymbol{\mu}}) d\tilde{\boldsymbol{\mu}}}$$

If prior is Uniform($[0,1]^K$), i.e., $\pi(\boldsymbol{\mu}) = 1 \ \ \forall \boldsymbol{\mu}$:

$$= \frac{(\mu^{(a_0)})^{r_0}(1-\mu^{(a_0)})^{1-r_0}}{\int_{\tilde{\boldsymbol{\mu}} \in [0,1]^K} (\tilde{\mu}^{(a_0)})^{r_0}(1-\tilde{\mu}^{(a_0)})^{1-r_0} d\tilde{\boldsymbol{\mu}}} = \frac{(\mu^{(a_0)})^{r_0}(1-\mu^{(a_0)})^{1-r_0}}{\int_0^1 (\tilde{\mu}^{(a_0)})^{r_0}(1-\tilde{\mu}^{(a_0)})^{1-r_0} d\tilde{\mu}^{(a_0)}} = 2(\mu^{(a_0)})^{r_0}(1-\mu^{(a_0)})^{1-r_0}$$

Bayesian Bernoulli bandit (cont'd)

1. At $t = 0$, $\mathbb{P}(\boldsymbol{\mu}) = \pi(\boldsymbol{\mu})$
2. At $t = 1$, we have one data point $r_0 \sim$ Bernoulli($\mu^{(a_0)}$), and the distribution of $\boldsymbol{\mu}$ gets updated via Bayes rule:
$$\mathbb{P}(\boldsymbol{\mu} \mid r_0, a_0) = 2(\mu^{(a_0)})^{r_0}(1 - \mu^{(a_0)})^{1-r_0}$$
3. At $t = 2$, we have another data point $r_1 \sim$ Bernoulli($\mu^{(a_1)}$), and we can update the distribution of $\boldsymbol{\mu}$ again via Bayes rule, treating $\mathbb{P}(\boldsymbol{\mu} \mid r_0, a_0)$ as the prior

$\vdots$

Bayes rule at time step t gives us a distribution (called the posterior distribution)
$$\mathbb{P}(\boldsymbol{\mu} \mid r_0, a_0, r_1, a_1, \ldots, r_{t-1}, a_{t-1})$$
that exactly characterizes our uncertainty about $\boldsymbol{\mu}$. We can use this to choose a_t!

Bayesian Bernoulli bandit (cont'd)

Bayesian Bernoulli bandit with uniform prior on $\boldsymbol{\mu}$ gives a running posterior on the mean of each arm k that is Beta$(1 + \#\{\text{arm } k \text{ successes}\}, 1 + \#\{\text{arm } k \text{ failures}\})$

(derived by Bayes rule and some algebra, see HW2)

Beta(α_k, β_k) has mean (posterior mean = what we *expect* $\mu^{(k)}$ to be):

$$\frac{\alpha_k}{\alpha_k + \beta_k} = \frac{1 + \#\{\text{arm } k \text{ successes}\}}{2 + \#\{\text{arm } k \text{ pulls}\}}$$

which starts at 1/2 and approaches the sample mean of arm k with more pulls.

Beta(α_k, β_k) has variance (posterior variance ≈ how uncertain we are about $\mu^{(k)}$):

$$\frac{\alpha_k}{\alpha_k + \beta_k} \times \frac{\beta_k}{\alpha_k + \beta_k} \times \frac{1}{\alpha_k + \beta_k + 1}$$

which decreases at a rate of roughly $1/\#\{\text{arm } k \text{ pulls}\}$

Bayesian bandit summary

A Bayesian bandit augments the bandit environment we've been working in so far with a prior distribution on the unknown reward distributions; for Bernoulli bandits, the reward distributions are entirely characterized by $\boldsymbol{\mu}$, so prior is: $\pi(\boldsymbol{\mu})$

Bayes rule at time step t gives us a distribution (called the posterior distribution)

$$\mathbb{P}(\boldsymbol{\mu} \mid r_0, a_0, r_1, a_1, \ldots, r_{t-1}, a_{t-1})$$

that exactly characterizes our uncertainty about $\boldsymbol{\mu}$.

Note that although we are now treating $\boldsymbol{\mu}$ as random, we still assume its value is only drawn once (from the prior) and then stays the same throughout t

What changes with t is our information about $\boldsymbol{\mu}$, i.e., the posterior distribution, as we collect more and more data by pulling arms via a bandit algorithm

Today

- ✓ Feedback from last lecture
- ✓ Recap
- ✓ Bayesian bandit
- Thompson sampling

Thompson sampling

Bayesian bandit environment means that at every time step, we know the distribution of the arm reward distributions conditioned on everything we've seen so far

In particular, we know the exact probability, given everything we've seen so far, that each arm is the true optimal arm, i.e.,

$\forall k,$ we know $\mathbb{P}(k = k^\star \mid r_0, a_0, \ldots, r_{t-1}, a_{t-1})$

Thompson sampling: sample from this distribution to determine next arm to pull

For $t = 0,\ldots,T-1$:

$a_t \sim$ distribution of $k^\star \mid r_0, a_0, \ldots, r_{t-1}, a_{t-1}$

How can we sample from this distribution? Draw a sample $\boldsymbol{\mu}_t \sim$ distribution of $\boldsymbol{\mu} \mid r_0, a_0, \ldots, r_{t-1}, a_{t-1}$ and then compute $a_t = \arg\max_k \mu_t^{(k)}$, which is the same thing as $a_t \sim$ distribution of $k^\star \mid r_0, a_0, \ldots, r_{t-1}, a_{t-1}$

That's it! Statistically, this is a super simple and elegant algorithm (though computationally, it may not be easy to update the posterior at each time step)

Thompson sampling intuition

Thompson sampling: $a_t \sim$ distribution of $k^\star \mid r_0, a_0, \ldots, r_{t-1}, a_{t-1}$

Why is this a good idea?

A good tradeoff of exploration vs exploitation should:

a) Sample the optimal arm as much as possible (duh)
b) Ensure arms that might still be optimal aren't overlooked
c) Not waste undue time on less promising arms

Intuitively: want to sample arms proportionally to how promising they are

This is exactly what Thompson sampling does, where "promising" is encoded very naturally as: "the probability that the arm is the optimal arm, given all the data so far"

No arbitrary δ tuning parameter, but do have to choose prior π

π can often be chosen "uninformatively" to a default prior such as the uniform, or can encode nuanced prior information/belief about the arms' reward distributions

Thompson sampling vs other algorithms

Thompson sampling samples arms proportionally to how promising they are

Note this sampling is much more sophisticated than, say, ε-greedy, which really just samples according to 2 categories: “most promising” and “other”

But it’s also quite different from UCB, whose OFU approach doesn’t really involve “sampling” at all, i.e., every a_t for UCB is a *deterministic* function of the previous data

My interpretation: OFU provides a simple heuristic to accomplish what Thompson sampling does by design, namely, sample arms according to how promising they are

Thompson sampling can do this because of the Bayesian bandit: assuming a prior on the reward distributions makes the arm means random, otherwise it wouldn’t even make sense to talk about “the probability that an arm is the best arm”

Although derived from the Bayesian bandit, Thompson sampling has excellent practical performance across bandit problems, whether or not they are Bayesian!

Thompson sampling in practice

Thompson sampling has excellent performance in practice, but is still just a heuristic

However, asymptotically, i.e., as $T \to \infty$, it actually is optimal in a certain sense

There is an *instance-dependent* lower-bound result that says that for any bandit algorithm:

$$\liminf_{T\to\infty} \frac{\mathbb{E}[N_T^{(k)}]}{\ln(T)} \geq \frac{1}{d(\nu^{(k^\star)}, \nu^{(k)})},$$

where d is a distance between distributions called the Kullback—Leibler divergence

It turns out that Thompson sampling satisfies this lower-bound with equality!

So it is asymptotically optimal, not just in its rate, but its constant too!

(UCB is not, but there are more complicated versions of it that are)

Thompson sampling in practice (cont'd)

So Thompson sampling is basically exactly optimal for large T

What could go wrong for smaller T? Suppose $K = 2$ and $T = 3$, and:

- $t = 0$: $a_0 = 1$, $r_0 = 1$
- $t = 1$: $a_1 = 2$, $r_1 = 0$
- $t = 2$ (last time step, with $\hat{\mu}_2^{(1)} = 1$ and $\hat{\mu}_2^{(2)} = 0$): $a_2 = ?$

Thompson sampling has a decent probability of choosing $a_2 = 2$, since with just one sample from each arm, Thompson sampling isn't sure which arm is best.

But $a_2 = 1$ is clear right choice here: there is no future value to learning more, i.e., no reason to explore rather than exploit.

Thompson sampling doesn't know this, and neither does UCB (although UCB wouldn't happen to make the same mistake in this case).

Thompson sampling in practice (cont'd)

For small T, Thompson sampling is not greedy enough

Fix: add a tuning parameter to make it more greedy. Some possibilities:

- Update the Beta parameters by $1+\epsilon$ instead of just 1 each time
- Instead of just taking one sample of $\boldsymbol{\mu}$ and computing the greedy action with respect to it, take n samples, compute the greedy action with respect to each, and pick the *mode* of those greedy actions

All of these favor arms that the algorithm has more confidence are good (i.e., arms that have worked well so far), as opposed to arms that *may* be good

Such tuning can improve Thompson sampling's performance even for reasonably large T (the asymptotic optimality of vanilla TS is *very* asymptotic)

Today

- Feedback from last lecture
- Recap
- Bayesian bandit
- Thompson sampling

Summary:

- Bayesian bandit adds an additional assumption of prior on reward distributions
- Bayes rule gives exact running uncertainty quantification for any algorithm
- Thompson sampling samples optimal arm from its (posterior) distribution
- Thompson sampling achieves excellent performance in practice

Attendance:

bit.ly/3RcTC9T

Feedback:

bit.ly/3RHtlxy