Bandits: Thompson Sampling and Gittins Index

Lucas Janson and Sham Kakade

CS/Stat 184: Introduction to Reinforcement Learning
Fall 2022

Today

- Feedback from last lecture
- Recap
- Thompson sampling
- Gittins index

Feedback from feedback forms

1. Thank you to everyone who filled out the forms!
2. Bit of confusion about Bayesian bandits
    - I'll review briefly, but section this week and HW 1 will give examples

Today

- ✓ Feedback from last lecture
- Recap
- Thompson sampling
- Gittins index

Recap

- Algorithms we've seen so far: pure greedy, pure exploration, ETC, ε-greedy, and UCB
- Bayesian bandit augments bandit environment with a prior distribution, allowing arm means to be treated as random with known distribution

Today

- ✓ Feedback from last lecture
- ✓ Recap
- Thompson sampling
- Gittins index

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: $\pi(\nu^{(1)}, \ldots, \nu^{(K)})$

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

Bayesian bandit example

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

(derived by Bayes rule and some algebra; see this week's section or HW1 for more details)

$\text{Beta}(\alpha_k, \beta_k)$ 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.

$\text{Beta}(\alpha_k, \beta_k)$ has variance (posterior variance $\approx$ 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 $1/\#\{\text{arm } k \text{ pulls}\}$

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.,

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

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

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

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

(In practice, usually 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:

Recall our instance-dependent UCB regret bound proved that with high probability,

$$N_t^{(k)} \leq \frac{2\ln(2KT/\delta)}{g_k} \text{ or, equivalently, } \frac{N_t^{(k)}}{2\ln(2KT/\delta)} \leq \frac{1}{g_k}$$

There is actually a 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

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 greatly 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
- ✓ Thompson sampling
- Gittins index

A different notion of finite horizon

So far, we have always taken the time horizon T to be fixed

Another model we might consider is for the T to be *random*, and a simple yet intuitive distribution for it is to imagine that the bandit ends at each time step with a fixed probability $1-\gamma$, given that it has reached that time step:

1. $\mathbb{P}(T=1) = 1-\gamma$
2. $\mathbb{P}(T=2) = \mathbb{P}(T=2 \mid T>1)\mathbb{P}(T>1) = (1-\gamma)\gamma$
3. $\mathbb{P}(T=3) = \mathbb{P}(T=3 \mid T>2)\mathbb{P}(T>2) = (1-\gamma)(1-(1-\gamma)-(1-\gamma)\gamma)$
$= (1-\gamma)\gamma^2$

… T is geometric, i.e., $\mathbb{P}(T=n) = (1-\gamma)\gamma^{n-1}$

Thus, assuming T independent of the data, then we'll get to r_t w/p:

Exactly optimizing the expected reward

So we see any given r_t w/p $\gamma^t \Rightarrow$ total reward, in expectation only over T, is $\sum_{t=0}^{\infty} \gamma^t r_t$

In the Bayesian bandit, there is an algorithm that *exactly* optimizes $\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$

Gittins index is both the name of a quantity that can be computed for each arm at any given time, and often also used to refer to the algorithm that chooses the argmax of the Gittins index at each time point

Proof of optimality is beyond scope of class, but the algorithm itself isn't too hard to understand intuitively

There are some computation/implementation aspects that you will deal with on HW1

One-armed bandit (OAB)

To compute the Gittins index for arm k, recall that we are in a Bayesian bandit, so we have a posterior for the distribution of arm k at any given time: $\mathbb{P}(\nu^{(k)})$, where I am suppressing the fact that we are conditioning on all the data we've seen so far—all that matters is that we have a distribution on $\nu^{(k)}$

Suppose that there were just one other arm, call it arm 0, whose reward distribution is a point mass at some *known* value λ; this is called a one-armed bandit (OAB)

Given λ and $\mathbb{P}(\nu^{(k)})$, we can ask: which arm should we pull next in order to maximize

$$\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$$ in the OAB?

One-armed bandit (cont'd)

OAB: arm 0 (with known deterministic reward λ) or arm k (with posterior $\mathbb{P}(\nu^{(k)})$)

The reward function is self-similar over time, meaning that it doesn't matter what t is when we ask this question (because at time t, I want to maximize

$$\mathbb{E}\left[\sum_{\tau=t}^{\infty}\gamma^{\tau}r_{\tau}\right] = \gamma^{t}\mathbb{E}\left[\sum_{\tau=0}^{\infty}\gamma^{\tau}r_{t+\tau}\right]$$

, which is the same as maximizing the original

expected reward $\mathbb{E}\left[\sum_{\tau=0}^{\infty}\gamma^{\tau}r_{\tau}\right]$ if time starts at t)

Also, if you choose arm 0, you gain no new information (since you already know arm 0's full reward distribution), and hence you should make the same OAB decision at the next time step, and forever after

Gittins Index

Gittins index for arm k with posterior $\mathbb{P}(\nu^{(k)})$ is the value $\lambda^\star(\mathbb{P}(\nu^{(k)}))$ of λ at which the optimal choice between arm 0 and arm k is a toss-up (i.e., for $\lambda < \lambda^\star(\mathbb{P}(\nu^{(k)}))$, arm k is the optimal choice, and for $\lambda > \lambda^\star(\mathbb{P}(\nu^{(k)}))$, arm 0 is the optimal choice)

How to think about the optimal choice in the OAB given λ and $\mathbb{P}(\nu^{(k)})$?

We can think of $\mathbb{P}(\nu^{(k)})$ as our state of belief about arm k's reward distribution, and by Bayes rule, we know how that state will evolve after the next observation, i.e., we know what $\mathbb{P}(\nu^{(k)} \mid a_0, r_0)$ will be for any a_0, r_0 (recall that $\mathbb{P}(\nu^{(k)} \mid a_0, r_0) = \mathbb{P}(\nu^{(k)})$ if $a_0 = 0$)

Now let's think about the optimal value $V(\mathbb{P}(\nu^{(k)}), \lambda, \gamma) := \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$ depending on which decision is optimal

Gittins Index (cont'd)

If arm 0 is optimal, then arm 0 will stay optimal and $V(\mathbb{P}(\nu^{(k)}), \lambda, \gamma) = \dfrac{\lambda}{1-\gamma}$

If arm k is optimal, $V(\mathbb{P}(\nu^{(k)}), \lambda, \gamma) = \mathbb{E}_{r\sim\nu^{(k)}}\left[r + \gamma V(\mathbb{P}(\nu^{(k)} \mid a_0 = k, r_0 = r), \lambda, \gamma)\right]$

So the true $V(\mathbb{P}(\nu^{(k)}), \lambda, \gamma)$ is the max of these two

Since the V function appears on both sides of the equation (though with different arguments), this formula is a recursive one

Details of recursion left to HW1 (for Bayesian Bernoulli bandit), but the point is that in some cases, $V(\mathbb{P}(\nu^{(k)}), \lambda, \gamma)$ can be extremely well-approximated quite efficiently, at which point we can just check if it is bigger than $\lambda/(1-\gamma)$ to see what the optimal choice is (if bigger, arm k optimal, otherwise arm 0 optimal)

Then just search over λ to find $\lambda^\star$, the Gittins index!

Today's summary:

Thompson sampling

- Operates in Bayesian bandit environment
- Samples optimal arm from its (posterior) distribution
- Achieves strong performance in practice

Gittins index

- Operates in Bayesian bandit with random horizon
- Exactly optimal in terms of expected (discounted) reward
- Some computational details to work out on HW1

Next time:

- Contextual bandits
- Other flavors of bandits

1-minute feedback form: https://bit.ly/3RHtlxy