Bandits: Gittins index and Contextual bandits

Lucas Janson and Sham Kakade

CS/Stat 184: Introduction to Reinforcement Learning

Fall 2022

Today

- Feedback from last lecture
- Recap
- Gittins index
- Contextual bandits

Feedback from feedback forms

1. Thank you to everyone who filled out the forms!
2. Microphone for both of us so we’re both audible in recording

Feedback from feedback forms

1. Thank you to everyone who filled out the forms!
2. Microphone for both of us so we're both audible in recording

One logistical reminder:

HW collaboration is allowed as long as you report on the homework who you worked with, but solutions must be written up independently

Today

- Feedback from last lecture
- Recap
- Gittins index
- Contextual bandits

Recap

Recap

- In Bayesian bandit environment, know distribution over everything at all t

Recap

- In Bayesian bandit environment, know distribution over everything at all t
- If we consider random/discounted reward $\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$, we know that too, in principle, for any given algorithm (where expectation is over random arm distributions too)

Today

- Feedback from last lecture
- Recap
- Gittins index
- Contextual bandits

Exactly optimizing the expected reward

Exactly optimizing the expected reward

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

Exactly optimizing the expected reward

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

Exactly optimizing the expected reward

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 we can cover and discuss the algorithm at a high level

Exactly optimizing the expected reward

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 we can cover and discuss the algorithm at a high level

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

One-armed bandit (OAB)

Computing the Gittins index for arm k:

One-armed bandit (OAB)

Computing 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 $\nu^{(k)}$

One-armed bandit (OAB)

Computing 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)

One-armed bandit (OAB)

Computing 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

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

One-armed bandit (cont'd)

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

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)

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

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)

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 $\mathbb{P}(\nu^{(k)})$?

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 $a_0 = 0$)

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)

Gittins Index (cont'd)

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

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

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

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

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)

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!

Gittins Index (cont’d)

Gittins Index (cont'd)

Comments about Gittins index:

Gittins Index (cont'd)

Comments about Gittins index:

- Its exact optimality is quite complicated to understand and beyond the scope of this class

Gittins Index (cont'd)

Comments about Gittins index:

- Its exact optimality is quite complicated to understand and beyond the scope of this class
- Relies on knowing γ and the prior π, and optimal in expectation over π

Gittins Index (cont'd)

Comments about Gittins index:

- Its exact optimality is quite complicated to understand and beyond the scope of this class
- Relies on knowing γ and the prior π, and optimal in expectation over π
- The Gittins index can be computed for an arm without any information about the other arms!

Gittins Index (cont'd)

Comments about Gittins index:

- Its exact optimality is quite complicated to understand and beyond the scope of this class
- Relies on knowing γ and the prior π, and optimal in expectation over π
- The Gittins index can be computed for an arm without any information about the other arms!
- It performs REALLY well in practice, even a little outside of the regime it is exactly optimal for (e.g., fixed arm means, fixed T)

Gittins Index (cont'd)

Comments about Gittins index:

- Its exact optimality is quite complicated to understand and beyond the scope of this class
- Relies on knowing γ and the prior π, and optimal in expectation over π
- The Gittins index can be computed for an arm without any information about the other arms!
- It performs REALLY well in practice, even a little outside of the regime it is exactly optimal for (e.g., fixed arm means, fixed T)
- Hard to extend exact optimality beyond this setting, though could inspire ideas for new algorithms in other settings

Today

- Feedback from last lecture
- Recap
- Gittins index
- Contextual bandits

Beyond simple bandits

Beyond simple bandits

In a bandit, we are presented with the same decision at every time

Beyond simple bandits

In a bandit, we are presented with the same decision at every time

In practice, often decisions are not the same every time

Beyond simple bandits

In a bandit, we are presented with the same decision at every time

In practice, often decisions are not the same every time

E.g., in online advertising there may not be a single best ad to show all users on all websites:

Beyond simple bandits

In a bandit, we are presented with the same decision at every time

In practice, often decisions are not the same every time

E.g., in online advertising there may not be a single best ad to show all users on all websites:

- maybe some types of users prefer one ad while others prefer another, or

Beyond simple bandits

In a bandit, we are presented with the same decision at every time

In practice, often decisions are not the same every time

E.g., in online advertising there may not be a single best ad to show all users on all websites:

- maybe some types of users prefer one ad while others prefer another, or
- maybe one type of ad works better on certain websites while another works better on other websites

Beyond simple bandits

In a bandit, we are presented with the same decision at every time

In practice, often decisions are not the same every time

E.g., in online advertising there may not be a single best ad to show all users on all websites:

- maybe some types of users prefer one ad while others prefer another, or
- maybe one type of ad works better on certain websites while another works better on other websites

Which user comes in next is random, but we have some context to tell situations apart and hence learn different optimal actions

Contextual bandit environment

Contextual bandit environment

Context at time t encoded into a variable x_t that we see before choosing our action

Contextual bandit environment

Context at time t encoded into a variable x_t that we see before choosing our action

x_t is drawn i.i.d. at each time point from a distribution ν_x on sample space $\mathscr{X}$

Contextual bandit environment

Context at time t encoded into a variable x_t that we see before choosing our action

x_t is drawn i.i.d. at each time point from a distribution ν_x on sample space $\mathcal{X}$

x_t then affects the reward distributions of each arm, i.e., if we choose arm k, we get a reward that is drawn from a distribution that depends on x_t, namely, $\nu^{(k)}(x_t)$

Contextual bandit environment

Context at time t encoded into a variable x_t that we see before choosing our action

x_t is drawn i.i.d. at each time point from a distribution ν_x on sample space $\mathcal{X}$

x_t then affects the reward distributions of each arm, i.e., if we choose arm k, we get a reward that is drawn from a distribution that depends on x_t, namely, $\nu^{(k)}(x_t)$

Accordingly, we should also choose our action a_t in a way that depends on x_t, i.e., our action should be chosen by a function of x_t (a policy), namely, $\pi_t(x_t)$

Contextual bandit environment

Context at time t encoded into a variable x_t that we see before choosing our action

x_t is drawn i.i.d. at each time point from a distribution ν_x on sample space $\mathscr{X}$

x_t then affects the reward distributions of each arm, i.e., if we choose arm k, we get a reward that is drawn from a distribution that depends on x_t, namely, $\nu^{(k)}(x_t)$

Accordingly, we should also choose our action a_t in a way that depends on x_t, i.e., our action should be chosen by a function of x_t (a policy), namely, $\pi_t(x_t)$

(Sorry for overlap with notation for prior in Bayesian bandit; both extremely standard)

Contextual bandit environment

Context at time t encoded into a variable x_t that we see before choosing our action

x_t is drawn i.i.d. at each time point from a distribution ν_x on sample space $\mathcal{X}$

x_t then affects the reward distributions of each arm, i.e., if we choose arm k, we get a reward that is drawn from a distribution that depends on x_t, namely, $\nu^{(k)}(x_t)$

Accordingly, we should also choose our action a_t in a way that depends on x_t, i.e., our action should be chosen by a function of x_t (a policy), namely, $\pi_t(x_t)$

(Sorry for overlap with notation for prior in Bayesian bandit; both extremely standard)

If we knew everything about the environment, we'd want to use the optimal policy

$$\pi^\star(x_t) := \arg \max_{k \in \{1,\ldots,K\}} \mu^{(k)}(x_t), \qquad \text{where } \mu^{(k)}(x) := \mathbb{E}_{r \sim \nu^{(k)}(x)}[r]$$

Contextual bandit environment

Context at time t encoded into a variable x_t that we see before choosing our action

x_t is drawn i.i.d. at each time point from a distribution ν_x on sample space $\mathcal{X}$

x_t then affects the reward distributions of each arm, i.e., if we choose arm k, we get a reward that is drawn from a distribution that depends on x_t, namely, $\nu^{(k)}(x_t)$

Accordingly, we should also choose our action a_t in a way that depends on x_t, i.e., our action should be chosen by a function of x_t (a policy), namely, $\pi_t(x_t)$

(Sorry for overlap with notation for prior in Bayesian bandit; both extremely standard)

If we knew everything about the environment, we'd want to use the optimal policy

$\pi^\star(x_t) := \arg \max_{k \in \{1,\ldots,K\}} \mu^{(k)}(x_t)$, where $\mu^{(k)}(x) := \mathbb{E}_{r \sim \nu^{(k)}(x)}[r]$

$\pi^\star$ is the policy we compare to in computing regret

Contextual bandit environment (cont'd)

Formally, a contextual bandit is the following interactive learning process:

Contextual bandit environment (cont'd)

Formally, a contextual bandit is the following interactive learning process:

For $t = 0 \to T - 1$

Contextual bandit environment (cont'd)

Formally, a contextual bandit is the following interactive learning process:

For $t = 0 \to T - 1$

1. Learner sees context $x_t \sim \nu_x$

Contextual bandit environment (cont'd)

Formally, a contextual bandit is the following interactive learning process:

For $t = 0 \to T - 1$

1. Learner sees context $x_t \sim \nu_x$ Independent of any previous data

Contextual bandit environment (cont'd)

Formally, a contextual bandit is the following interactive learning process:

For $t = 0 \to T - 1$

1. Learner sees context $x_t \sim \nu_x$ Independent of any previous data
2. Learner pulls arm $a_t = \pi_t(x_t) \in \{1, \ldots, K\}$

Contextual bandit environment (cont'd)

Formally, a contextual bandit is the following interactive learning process:

For $t = 0 \to T - 1$

1. Learner sees context $x_t \sim \nu_x$ — Independent of any previous data
2. Learner pulls arm $a_t = \pi_t(x_t) \in \{1,\ldots,K\}$ — π_t policy learned from all data seen so far

Contextual bandit environment (cont'd)

Formally, a contextual bandit is the following interactive learning process:

For $t = 0 \to T - 1$

1. Learner sees context $x_t \sim \nu_x$ — Independent of any previous data
2. Learner pulls arm $a_t = \pi_t(x_t) \in \{1, \ldots, K\}$ — π_t policy learned from all data seen so far
3. Learner observes reward $r_t \sim \nu^{(a_t)}(x_t)$ from arm a_t in context x_t

Contextual bandit environment (cont'd)

Formally, a contextual bandit is the following interactive learning process:

For $t = 0 \to T - 1$

1. Learner sees context $x_t \sim \nu_x$ — Independent of any previous data
2. Learner pulls arm $a_t = \pi_t(x_t) \in \{1,\ldots,K\}$ — π_t policy learned from all data seen so far
3. Learner observes reward $r_t \sim \nu^{(a_t)}(x_t)$ from arm a_t in context x_t

Note that if the context distribution ν_x always returns the same value (e.g., 0), then the contextual bandit <u>reduces</u> to the original multi-armed bandit

Contextual bandit environment (cont'd)

Formally, a contextual bandit is the following interactive learning process:

For $t = 0 \rightarrow T - 1$

1. Learner sees context $x_t \sim \nu_x$ Independent of any previous data
2. Learner pulls arm $a_t = \pi_t(x_t) \in \{1,\ldots,K\}$ π_t policy learned from all data seen so far
3. Learner observes reward $r_t \sim \nu^{(a_t)}(x_t)$ from arm a_t in context x_t

Note that if the context distribution ν_x always returns the same value (e.g., 0), then the contextual bandit reduces to the original multi-armed bandit

π_t might seem unfamiliar since we haven't talked about a policy in bandits before, but actually we've always had it, it's just that without context, we didn't need a name or notation for it because it was so simple!

Contextual bandit algorithms

What was π_t for UCB? (π_t has no argument because there was no context)

Contextual bandit algorithms

What was π_t for UCB? (π_t has no argument because there was no context)

$$\pi_t = \arg\max_k \mathsf{UCB}_t^{(k)}$$

Contextual bandit algorithms

What was π_t for UCB? (π_t has no argument because there was no context)

$$\pi_t = \arg\max_k \mathsf{UCB}_t^{(k)}$$

For Thompson sampling?

Contextual bandit algorithms

What was π_t for UCB? (π_t has no argument because there was no context)

$$\pi_t = \arg\max_k \mathrm{UCB}_t^{(k)}$$

For Thompson sampling?

π_t was a *randomized* policy that sampled from the posterior distribution of $k^\star$

Contextual bandit algorithms

What was π_t for UCB? (π_t has no argument because there was no context)

$$\pi_t = \arg\max_k \mathrm{UCB}_t^{(k)}$$

For Thompson sampling?

π_t was a *randomized* policy that sampled from the posterior distribution of $k^\star$

Now what about contextual versions?

Contextual bandit algorithms

What was π_t for UCB? (π_t has no argument because there was no context)

$$\pi_t = \arg\max_k \mathrm{UCB}_t^{(k)}$$

For Thompson sampling?

π_t was a *randomized* policy that sampled from the posterior distribution of $k^\star$

Now what about contextual versions?

Thompson sampling with contexts is conceptually identical!

Contextual bandit algorithms

What was π_t for UCB? (π_t has no argument because there was no context)

$$\pi_t = \arg\max_k \mathrm{UCB}_t^{(k)}$$

For Thompson sampling?

π_t was a *randomized* policy that sampled from the posterior distribution of $k^\star$

Now what about contextual versions?

Thompson sampling with contexts is conceptually identical!

Still start from a prior on $\{\nu^{(k)}(x)\}_{k\in\{1,\ldots,K\},x\in\mathcal{X}}$,

Contextual bandit algorithms

What was π_t for UCB? (π_t has no argument because there was no context)

$$\pi_t = \arg\max_k \text{UCB}_t^{(k)}$$

For Thompson sampling?

π_t was a *randomized* policy that sampled from the posterior distribution of $k^\star$

Now what about contextual versions?

Thompson sampling with contexts is conceptually identical!

Still start from a prior on $\{\nu^{(k)}(x)\}_{k\in\{1,\ldots,K\},x\in\mathcal{X}}$,

but now this is $K|\mathcal{X}|$ (usually $\gg K$) distributions, so need more complicated prior

Contextual bandit algorithms

What was π_t for UCB? (π_t has no argument because there was no context)

$$\pi_t = \arg\max_k \mathrm{UCB}_t^{(k)}$$

For Thompson sampling?

π_t was a *randomized* policy that sampled from the posterior distribution of $k^\star$

Now what about contextual versions?

Thompson sampling with contexts is conceptually identical!

Still start from a prior on $\{\nu^{(k)}(x)\}_{k\in\{1,\ldots,K\},x\in\mathcal{X}}$,

but now this is $K|\mathcal{X}|$ (usually $\gg K$) distributions, so need more complicated prior

Still can update distribution on $\{\nu^{(k)}(x)\}_{k\in\{1,\ldots,K\},x\in\mathcal{X}}$ after each reward $r_t \sim \nu^{(a_t)}(x_t)$

Contextual bandit algorithms

What was π_t for UCB? (π_t has no argument because there was no context)

$$\pi_t = \arg\max_k \text{UCB}_t^{(k)}$$

For Thompson sampling?

π_t was a *randomized* policy that sampled from the posterior distribution of $k^\star$

Now what about contextual versions?

Thompson sampling with contexts is conceptually identical!

Still start from a prior on $\{\nu^{(k)}(x)\}_{k\in\{1,\ldots,K\},x\in\mathcal{X}}$,

but now this is $K|\mathcal{X}|$ (usually $\gg K$) distributions, so need more complicated prior

Still can update distribution on $\{\nu^{(k)}(x)\}_{k\in\{1,\ldots,K\},x\in\mathcal{X}}$ after each reward $r_t \sim \nu^{(a_t)}(x_t)$

Still know posterior over $k^\star(x_t)$ that can draw from to choose a_t; this is $\pi_t(x_t)$

UCB for contextual bandits

UCB for contextual bandits

UCB algorithm also conceptually identical as long as $|\mathcal{X}|$ finite:

$$\pi_t(x_t) = \arg\max_k \hat{\mu}_t^{(k)}(x_t) + \sqrt{\ln(2TK|\mathcal{X}|/\delta)/2N_t^{(k)}(x_t)}$$

UCB for contextual bandits

UCB algorithm also conceptually identical as long as $|\mathcal{X}|$ finite:

$$\pi_t(x_t) = \arg\max_k \hat{\mu}_t^{(k)}(x_t) + \sqrt{\ln(2TK|\mathcal{X}|/\delta)/2N_t^{(k)}(x_t)}$$

- Added x_t argument to $\hat{\mu}_t^{(k)}$ and $N_t^{(k)}$ since we now keep track of the sample mean and number of arm pulls *separately* for each value of the context

UCB for contextual bandits

UCB algorithm also conceptually identical as long as $|\mathcal{X}|$ finite:

$$\pi_t(x_t) = \arg\max_k \hat{\mu}_t^{(k)}(x_t) + \sqrt{\ln(2TK|\mathcal{X}|/\delta)/2N_t^{(k)}(x_t)}$$

- Added x_t argument to $\hat{\mu}_t^{(k)}$ and $N_t^{(k)}$ since we now keep track of the sample mean and number of arm pulls *separately* for each value of the context
- Added $|\mathcal{X}|$ inside the log because our union bound argument is now over all arm mean estimates $\hat{\mu}_t^{(k)}(x)$, of which there are $K|\mathcal{X}|$ instead of just K

UCB for contextual bandits

UCB algorithm also conceptually identical as long as $|\mathcal{X}|$ finite:

$$\pi_t(x_t) = \arg\max_k \hat{\mu}_t^{(k)}(x_t) + \sqrt{\ln(2TK|\mathcal{X}|/\delta)/2N_t^{(k)}(x_t)}$$

- Added x_t argument to $\hat{\mu}_t^{(k)}$ and $N_t^{(k)}$ since we now keep track of the sample mean and number of arm pulls *separately* for each value of the context
- Added $|\mathcal{X}|$ inside the log because our union bound argument is now over all arm mean estimates $\hat{\mu}_t^{(k)}(x)$, of which there are $K|\mathcal{X}|$ instead of just K

But when $|\mathcal{X}|$ is really big (or even infinite), this will be really bad!

UCB for contextual bandits

UCB algorithm also conceptually identical as long as $|\mathcal{X}|$ finite:

$$\pi_t(x_t) = \arg\max_k \hat{\mu}_t^{(k)}(x_t) + \sqrt{\ln(2TK|\mathcal{X}|/\delta)/2N_t^{(k)}(x_t)}$$

- Added x_t argument to $\hat{\mu}_t^{(k)}$ and $N_t^{(k)}$ since we now keep track of the sample mean and number of arm pulls *separately* for each value of the context
- Added $|\mathcal{X}|$ inside the log because our union bound argument is now over all arm mean estimates $\hat{\mu}_t^{(k)}(x)$, of which there are $K|\mathcal{X}|$ instead of just K

But when $|\mathcal{X}|$ is really big (or even infinite), this will be really bad!

Solution: share information across contexts x_t, i.e., don't treat $\nu^{(k)}(x)$ and $\nu^{(k)}(x')$ as completely different distributions which have nothing to do with one another

UCB for contextual bandits

UCB algorithm also conceptually identical as long as $|\mathcal{X}|$ finite:

$$\pi_t(x_t) = \arg\max_k \hat{\mu}_t^{(k)}(x_t) + \sqrt{\ln(2TK|\mathcal{X}|/\delta)/2N_t^{(k)}(x_t)}$$

- Added x_t argument to $\hat{\mu}_t^{(k)}$ and $N_t^{(k)}$ since we now keep track of the sample mean and number of arm pulls *separately* for each value of the context
- Added $|\mathcal{X}|$ inside the log because our union bound argument is now over all arm mean estimates $\hat{\mu}_t^{(k)}(x)$, of which there are $K|\mathcal{X}|$ instead of just K

But when $|\mathcal{X}|$ is really big (or even infinite), this will be really bad!

Solution: share information across contexts x_t, i.e., don't treat $\nu^{(k)}(x)$ and $\nu^{(k)}(x')$ as completely different distributions which have nothing to do with one another

Example: showing an ad on a NYT article on politics vs a NYT article on sports:

UCB for contextual bandits

UCB algorithm also conceptually identical as long as $|\mathcal{X}|$ finite:

$$\pi_t(x_t) = \arg\max_k \hat{\mu}_t^{(k)}(x_t) + \sqrt{\ln(2TK|\mathcal{X}|/\delta)/2N_t^{(k)}(x_t)}$$

- Added x_t argument to $\hat{\mu}_t^{(k)}$ and $N_t^{(k)}$ since we now keep track of the sample mean and number of arm pulls *separately* for each value of the context
- Added $|\mathcal{X}|$ inside the log because our union bound argument is now over all arm mean estimates $\hat{\mu}_t^{(k)}(x)$, of which there are $K|\mathcal{X}|$ instead of just K

But when $|\mathcal{X}|$ is really big (or even infinite), this will be really bad!

Solution: share information across contexts x_t, i.e., don't treat $\nu^{(k)}(x)$ and $\nu^{(k)}(x')$ as completely different distributions which have nothing to do with one another

Example: showing an ad on a NYT article on politics vs a NYT article on sports: Not *identical* readership, but still both on NYT, so probably still *similar* readership!

Modeling in contextual bandits

Modeling in contextual bandits

Need a model for $\mu^{(k)}(x)$, e.g., a linear model: $\mu^{(k)}(x) = \theta_k^\top x$

Modeling in contextual bandits

Need a model for $\mu^{(k)}(x)$, e.g., a linear model: $\mu^{(k)}(x) = \theta_k^\top x$

E.g., placing ads on NYT or WSJ (encoded as 0 or 1 in the first entry of x), for articles on politics or sports (encoded as 0 or 1 in the second entry of x) $\Rightarrow x \in \{0,1\}^2$

Modeling in contextual bandits

Need a model for $\mu^{(k)}(x)$, e.g., a linear model: $\mu^{(k)}(x) = \theta_k^\top x$

E.g., placing ads on NYT or WSJ (encoded as 0 or 1 in the first entry of x), for articles on politics or sports (encoded as 0 or 1 in the second entry of x) $\Rightarrow x \in \{0,1\}^2$

$|\mathcal{X}| = 4 \Rightarrow$ w/o linear model, need to learn 4 different $\mu^{(k)}(x)$ values for each arm k

Modeling in contextual bandits

Need a model for $\mu^{(k)}(x)$, e.g., a linear model: $\mu^{(k)}(x) = \theta_k^\top x$

E.g., placing ads on NYT or WSJ (encoded as 0 or 1 in the first entry of x), for articles on politics or sports (encoded as 0 or 1 in the second entry of x) $\Rightarrow x \in \{0,1\}^2$

$|\mathscr{X}| = 4 \Rightarrow$ w/o linear model, need to learn 4 different $\mu^{(k)}(x)$ values for each arm k

With linear model there are just 2 parameters: the two entries of $\theta_k \in \mathbb{R}^2$

Modeling in contextual bandits

Need a model for $\mu^{(k)}(x)$, e.g., a linear model: $\mu^{(k)}(x) = \theta_k^\top x$

E.g., placing ads on NYT or WSJ (encoded as 0 or 1 in the first entry of x), for articles on politics or sports (encoded as 0 or 1 in the second entry of x) $\Rightarrow x \in \{0,1\}^2$

$|\mathcal{X}| = 4 \Rightarrow$ w/o linear model, need to learn 4 different $\mu^{(k)}(x)$ values for each arm k

With linear model there are just 2 parameters: the two entries of $\theta_k \in \mathbb{R}^2$

Lower dimension makes learning easier, but model could be wrong/biased

Modeling in contextual bandits

Need a model for $\mu^{(k)}(x)$, e.g., a linear model: $\mu^{(k)}(x) = \theta_k^\top x$

E.g., placing ads on NYT or WSJ (encoded as 0 or 1 in the first entry of x), for articles on politics or sports (encoded as 0 or 1 in the second entry of x) $\Rightarrow x \in \{0,1\}^2$

$|\mathscr{X}| = 4 \Rightarrow$ w/o linear model, need to learn 4 different $\mu^{(k)}(x)$ values for each arm k

With linear model there are just 2 parameters: the two entries of $\theta_k \in \mathbb{R}^2$

Lower dimension makes learning easier, but model could be wrong/biased

Choosing the best model, fitting it, and quantifying uncertainty are essentially problems of supervised learning (for another day)

Today

- Feedback from last lecture
- Recap
- Gittins index
- Contextual bandits

Today’s summary:

Today's summary:

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

Today's summary:

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

Contextual bandits

- Adds i.i.d. context/state variable to bandit
- Same algorithms apply, just need to model $\mu^{(k)}(x)$
- Modeling aspects = supervised learning

Today's summary:

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

Contextual bandits

- Adds i.i.d. context/state variable to bandit
- Same algorithms apply, just need to model $\mu^{(k)}(x)$
- Modeling aspects = supervised learning

Next time:

- Markov decision processes (= contextual bandit where context can depend on previous action and context)

Today's summary:

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

Contextual bandits

- Adds i.i.d. context/state variable to bandit
- Same algorithms apply, just need to model $\mu^{(k)}(x)$
- Modeling aspects = supervised learning

Next time:

- Markov decision processes (= contextual bandit where context can depend on previous action and context)

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