XCHAT FANS

X

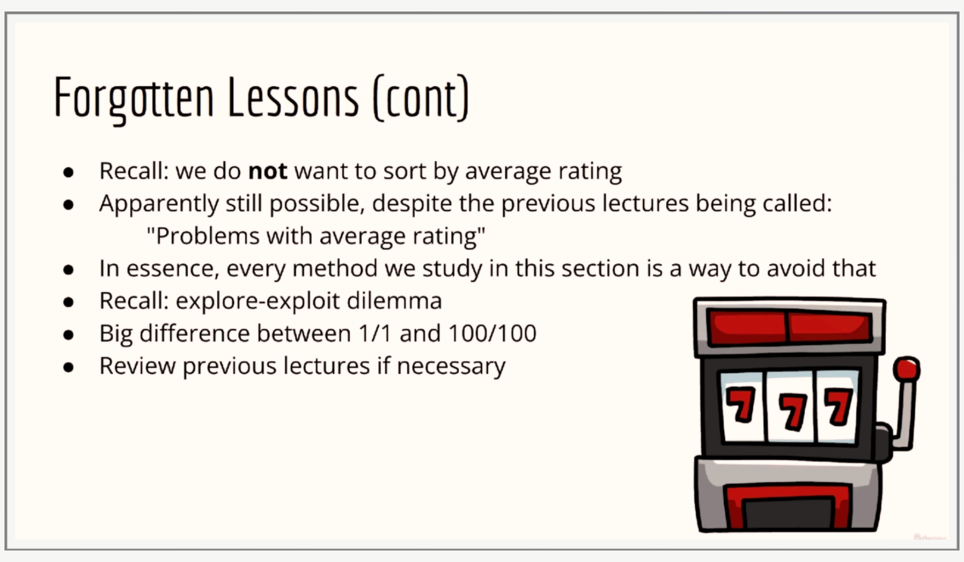
THE SPOT FOR YOUR X FRIEND EXPERIENCE

XChatFans Confidential

Bayesian Ranking

Bayesian Ranking for sorting user rankings

Well, let's identify one problem with ALL of the previous methods. They all got one point in common, they all rely on a ranking by a fixed number. We already know that sorting by the mean (average rating) is not suitable for reasons we just discussed i.e. Explore-Exploit-dilemma. In fact, this whole section exists because we do NOT want to sort by the mean. Well, one option we saw was to sort by the lower confidence bound. But still, it somehow feels like ranking by a fixed number is just not quite right.



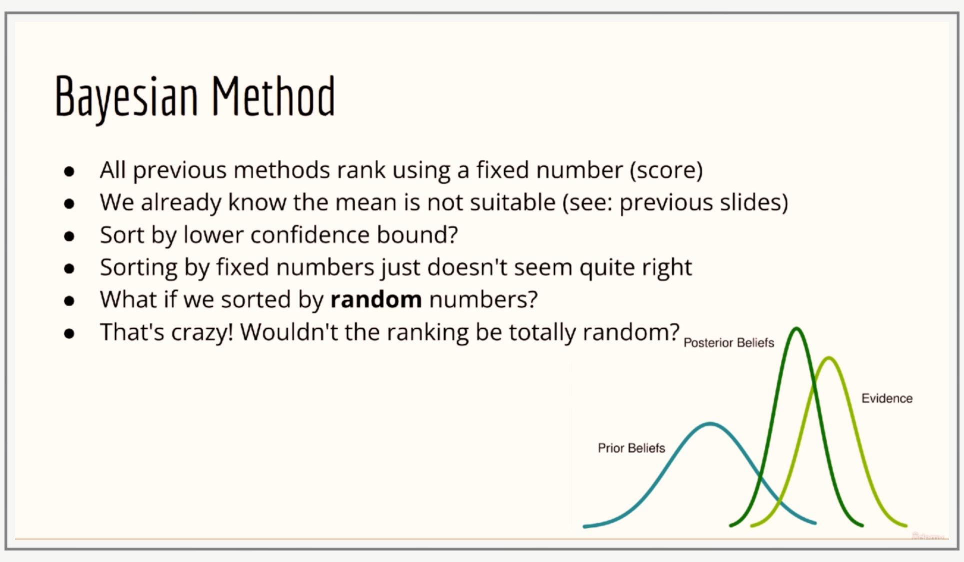
Forgotten Lessons (cont)

- Recall: we do **not** want to sort by average rating
- Apparently still possible, despite the previous lectures being called: "Problems with average rating"
- In essence, every method we study in this section is a way to avoid that
- Recall: explore-exploit dilemma
- Big difference between 1/1 and 100/100
- Review previous lectures if necessary

Normal Randomness

So, what is this **Bayesian method** all about?

What if instead, we ranked items by drawing random numbers? You might think this is ludicrous, if we just pick random scores for every item, wouldn't such a single random ranking be completely useless. Well, it depends on what you mean by random.

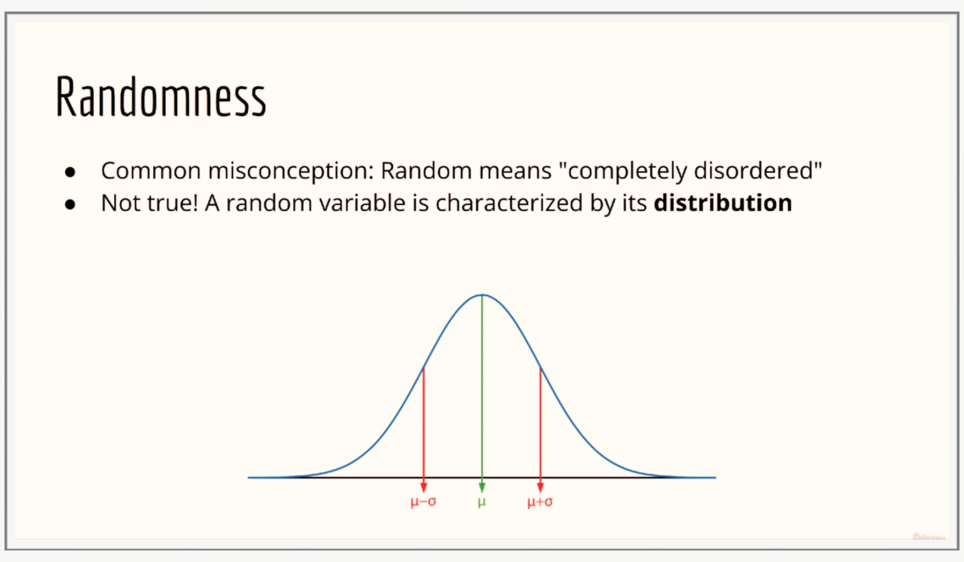


Bayesian Method

- All previous methods rank using a fixed number (score)
- We already know the mean is not suitable (see: previous slides)
- Sort by lower confidence bound?
- Sorting by fixed numbers just doesn't seem quite right
- What if we sorted by **random** numbers?
- That's crazy! Wouldn't the ranking be totally random?

Posterior Beliefs
Evidence
Prior Beliefs

One common misconception among some students who haven't studied probability is that there is only one kind of random—to them, random might simply mean completely disordered. This is NOT the case. In probability, we characterize the way in which something is random by its distribution. For example, consider the normal distribution. This distribution tells us certain information, like if we were to draw a random number. What is the most likely number?—a number close to the mean.



Randomness

- Common misconception: Random means "completely disordered"
- Not true! A random variable is characterized by its **distribution**

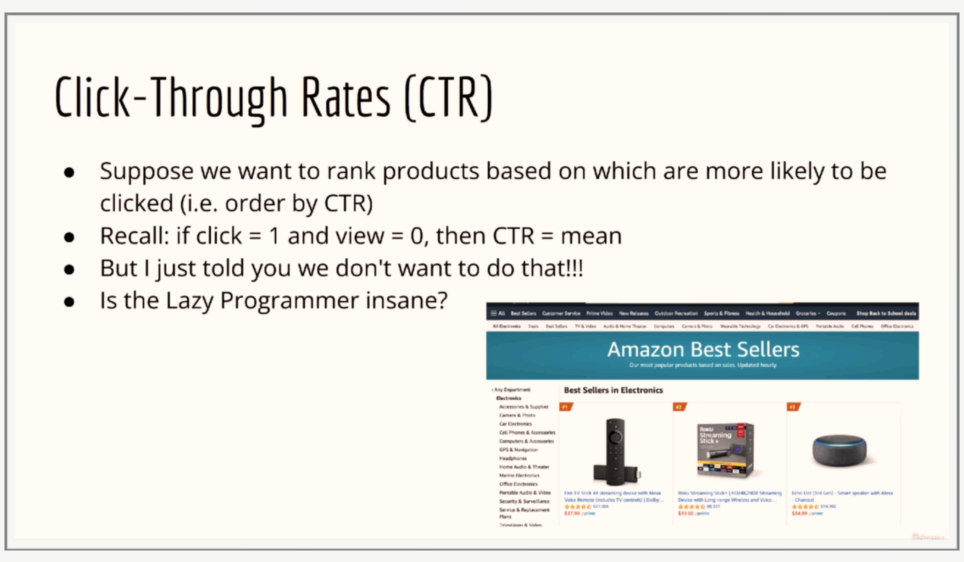
It tells us what values are more likely and what values are less likely. It also tells us the range of values we might get for the normal distribution. That's any number on the real line.

Click-Through Rates (CTR)

Click-Through Rate (**CTR**) : is the ratio of clicks on a specific link to the number of times a page is shown.

CTR = (#item_clicks / #page_visits) × 100%

Now, suppose we're dealing with **click through rates**, so perhaps we're trying to rank products on our homepage and we want to know in which order should we rank these products. Our goal is to get more people to click, which takes them to the product page from which they can presumably make a purchase.
So in essence, we want to order them by click through rate. Now, if we treat each click as a 1 in each view as a 0, then the click through rate is equivalent to the mean. As you recall—we do NOT want to sort by the mean for the reasons we described—explore exploit dilemma.
So we want to sort by click through rate—but I just told you it's a bad idea. This might sound like a paradox, but I assure you there is a solution.



Click-Through Rates (CTR)

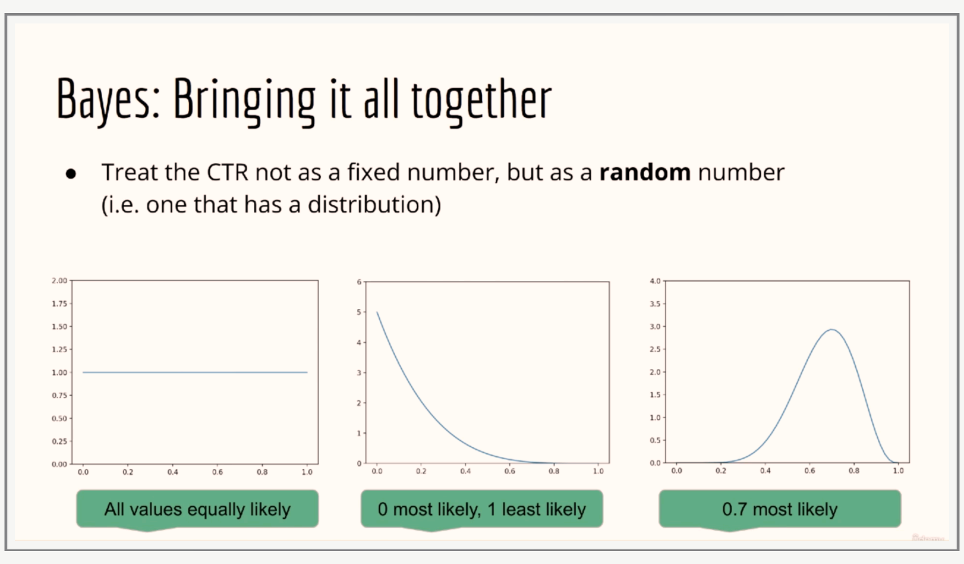
- Suppose we want to rank products based on which are more likely to be clicked (i.e. order by CTR)
- Recall: if click = 1 and view = 0, then CTR = mean
- But I just told you we don't want to do that!!!
- Is the Lazy Programmer insane?

Amazon Best Sellers
Best Sellers in Electronics

So what should we do instead?

Bayes Method

OK, so this is where we tie all these concepts together. You see, the **Bayesian method** works by treating the **Click-Through-Rate** NOT as a fixed number, but rather as a random number. That is to say it has a distribution—**random numbers have distribution**.

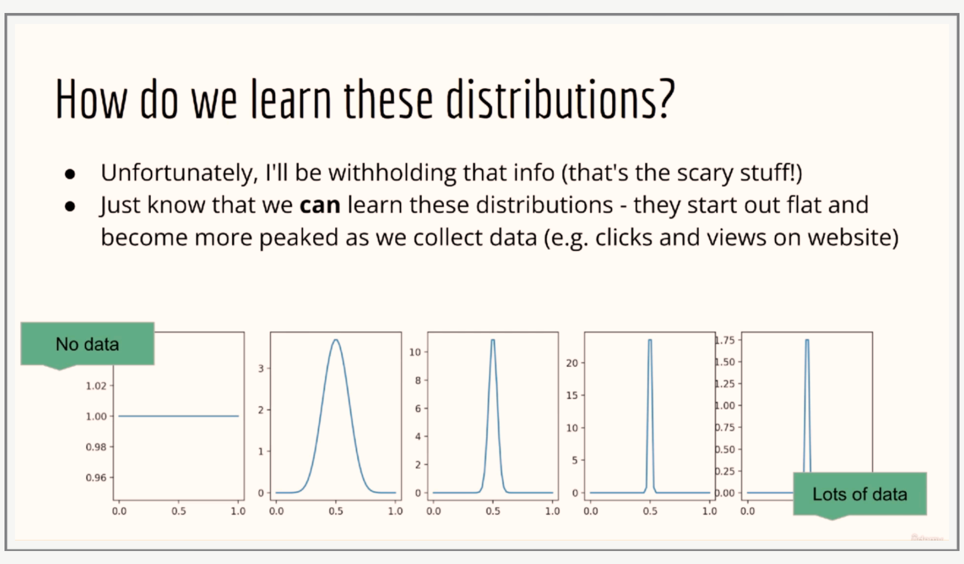


Bayes: Bringing it all together

- Treat the CTR not as a fixed number, but as a **random** number (i.e. one that has a distribution)

Now let's think about what these distributions tell us. Firstly, notice how all these distributions are bounded between zero and one. This makes sense. **Click-Through-Rates can only be between a zero and one**. So we'd better pick a distribution that matches this requirement.
So what else can we see?
(1)
Well, we can see that one of these distributions is completely flat. What does this mean? This means that we pretty much don't know anything about what the Click-Through-Rate is. We're saying that any Click-Through-Rate between zero and one is equally likely.
How about the next one?
(2)
This sort of looks like a ramp which starts off at zero, then goes down to one. So what does this say? Well, we can see that the highest point is at zero, implying that for this distribution, the most likely Click-Through-Rate is zero. But we're not completely sure which is why there is probability mass for other values of the CTR. However, we can see that for greater values of the CTR, they are less likely.
(3)
Finally, for the last chart, we can see a peaked distribution. It appears that in this case, we are pretty confident that the CTR is 0.7, but there is still some probability for the other values.

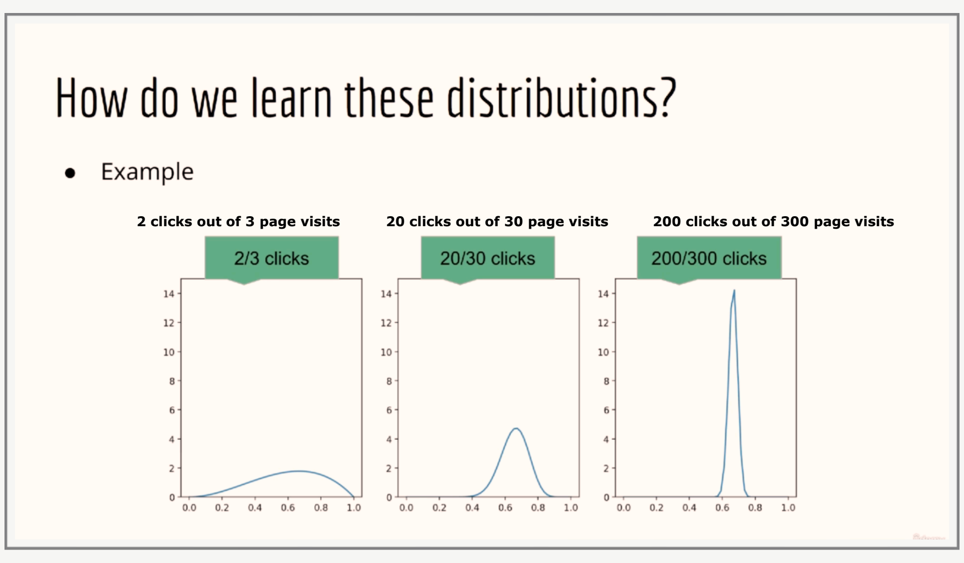
Now, the real question is, how do you get these distributions?
Unfortunately, this is the information I'll be withholding from you in this lecture. If you'd like to know how this works, you'll have to go to the **scary part** (of the course). Otherwise, I'll try to give you the main points. So suppose I have a bunch of products I want to rent. I'm going to give each of them their own distribution. These distributions will start out completely flat, meaning we assume nothing about their CTR. However, as time goes on and people visit our website, we're going to collect data. Every time we collect the data point, we're going to update these distributions. So for example, **every time a user sees a product that will count as a view, every time a user clicks on a product that will count as a click**.



How do we learn these distributions?

- Unfortunately, I'll be withholding that info (that's the scary stuff!)
- Just know that we **can** learn these distributions - they start out flat and become more peaked as we collect data (e.g. clicks and views on website)

So what happens as we collect this data?
Well, here's an example.

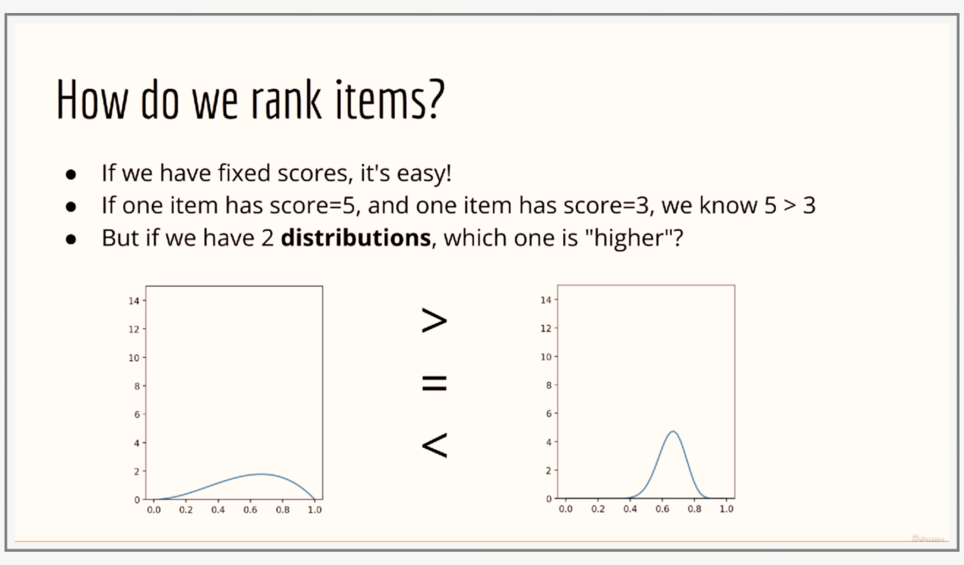


How do we learn these distributions?

- Example

The first plot shows two out of three clicks. This distribution is pretty wide and I'm not very confident about it's true value. The second plot shows 20 out of 30 clicks, this distribution is much skinnier, showing that as we collect more data, we become more confident that the true story really is ⅔. The third plot shows 200 out of 300 clicks. This distribution is even skinnier, so you can see that as we approach an infinite number of data points—our distribution approaches our point estimate.

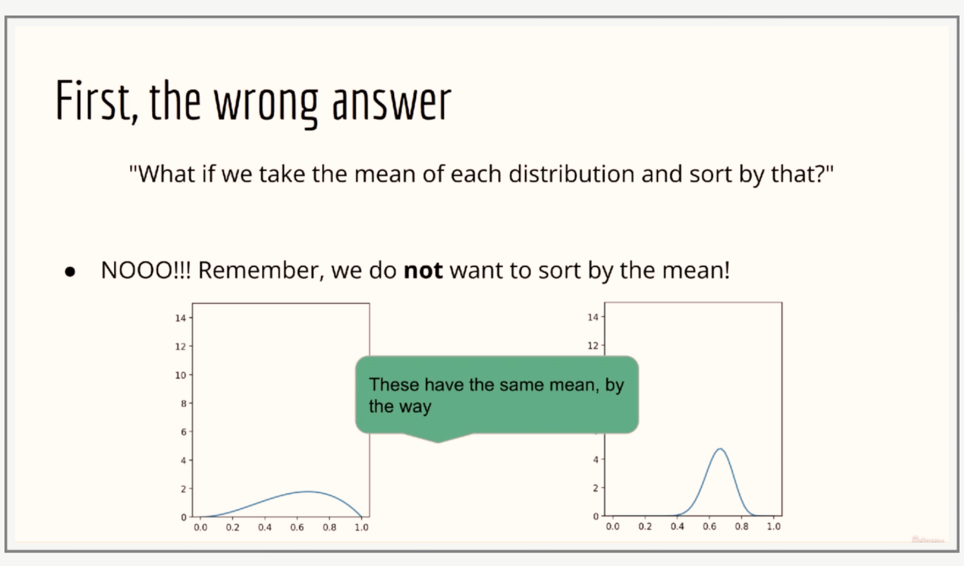
Now, what I just showed you is all well and good, but how do we actually use the distribution to rank items? This is not clear. When we have fixed numbers, this is easy. For example, if one item has a score of five in, another item has a score of three. Then we put the one with score five on top because five is bigger than three. But if we have two distributions, how do we decide which one should come first?



How do we rank items?

- If we have fixed scores, it's easy!
- If one item has score=5, and one item has score=3, we know 5 > 3
- But if we have 2 **distributions**, which one is "higher"?

Now you might think why not just take the mean of the distribution and a rank by that?
So again, this is what I warned you against. Remember, we do not want to sort by the mean.



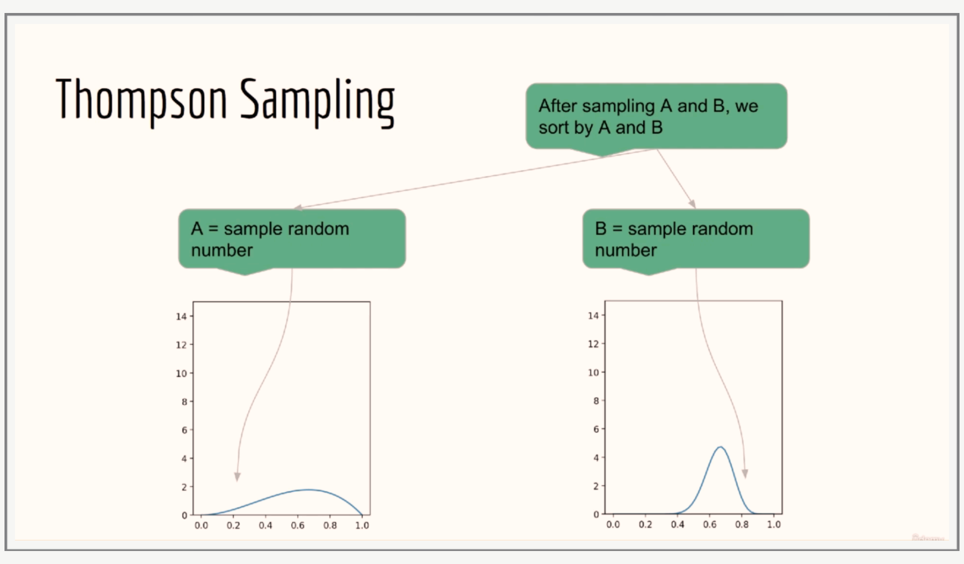
First, the wrong answer

"What if we take the mean of each distribution and sort by that?"

- NOOO!!! Remember, we do **not** want to sort by the mean!

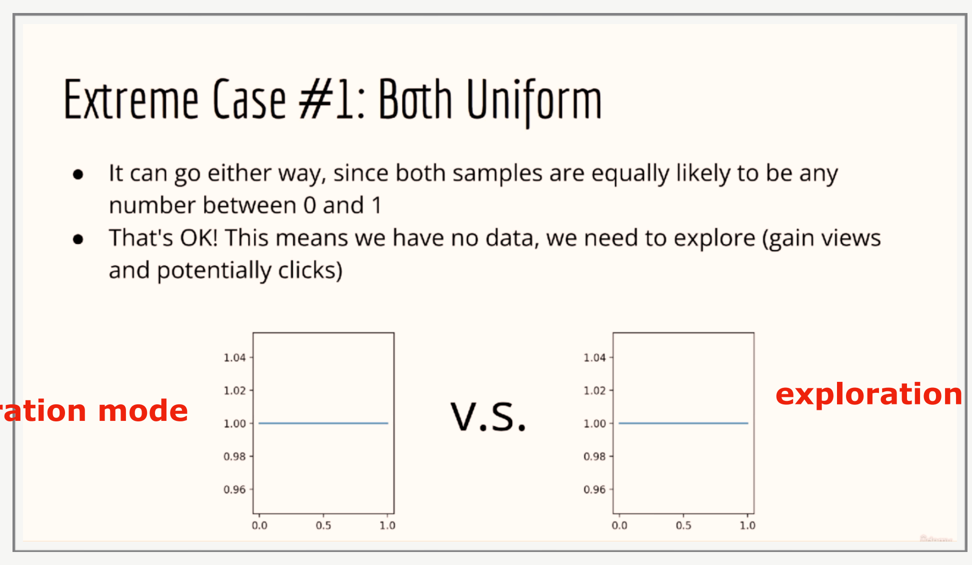
Thompson Sampling

As mentioned earlier in this lecture, we do this by **sampling random numbers**—in fact, this method is sometimes known as **Thompson sampling**.


Thompson Sampling

Scenario 1

To understand how this will work, it’s helpful to think of some extreme examples. Consider the case where both items are **uniform**—two **Uniform Distributions**. In this case, the samples we draw are completely random and it's NOT really predictable, which would come first. This is perfectly OK because remember that we're still at the stage where we want to explore and collect more data. If you don't know anything about which of these items are more likely to be clicked on, then we need to collect data to give us a better idea in the real world. This translates to the idea that we need to actually show these items to users to see if they will be clicked, or not only after users have seen the items. Will we have any idea of how good they are?

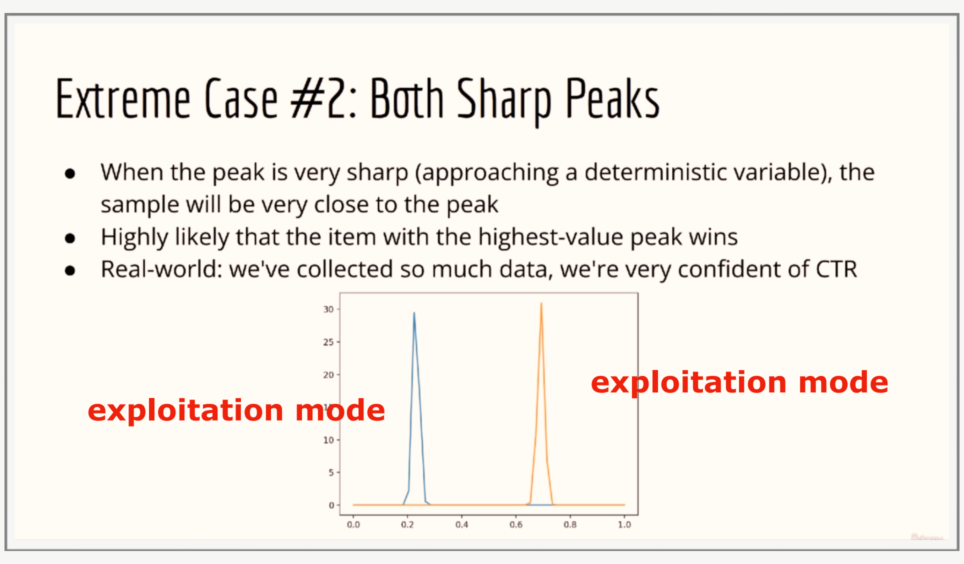

Extreme Case #1: Both Uniform

- It can go either way, since both samples are equally likely to be any number between 0 and 1
- That's OK! This means we have no data, we need to explore (gain views and potentially clicks)

exploration mode
V.S.
exploration mode

Scenario 2

Now, consider the case where both items have **very sharp peaks**. In this case, when we draw samples from these distributions, the samples are going to be very close to the peak. As an example, consider the peak at ⅔ (orange distribution), we might get a sample of 0.6665 or 0.6668. However, there's essentially NO chance we're going to draw a sample like, say, 0.2. Because of this, it's highly likely that the item whose peak is located at the highest position will win. To connect this back to the real world. This makes total sense. This is the situation where we've already collected millions of data points for both items. Because we’ve collected so much data, there’s essentially NO question about what the true story is, and because we're so confident that we know the true CTR, we have NO problem ranking these items by those CTR or equivalently by samples, which are very close to those CTR.

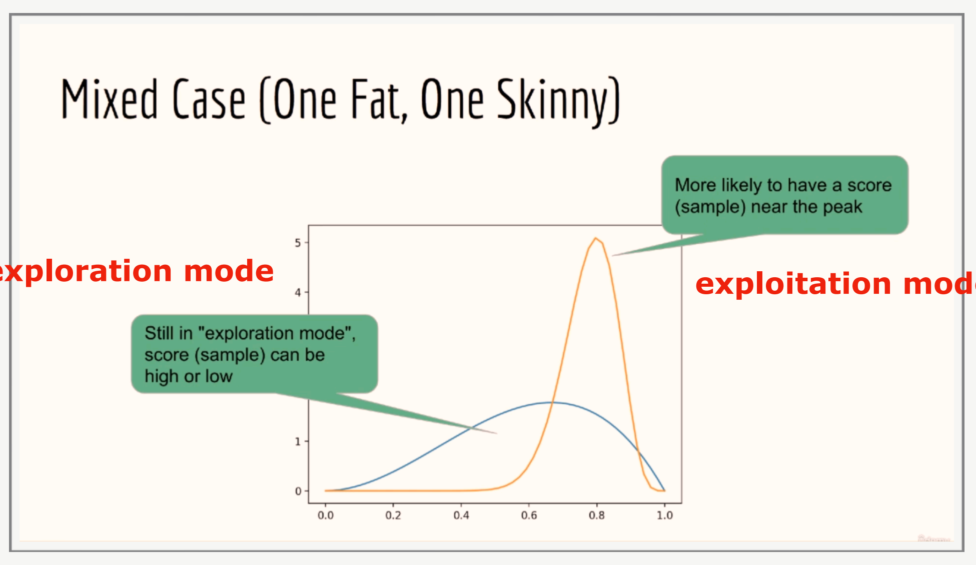

Extreme Case #2: Both Sharp Peaks

- When the peak is very sharp (approaching a deterministic variable), the sample will be very close to the peak
- Highly likely that the item with the highest-value peak wins
- Real-world: we've collected so much data, we're very confident of CTR

exploitation mode
exploitation mode

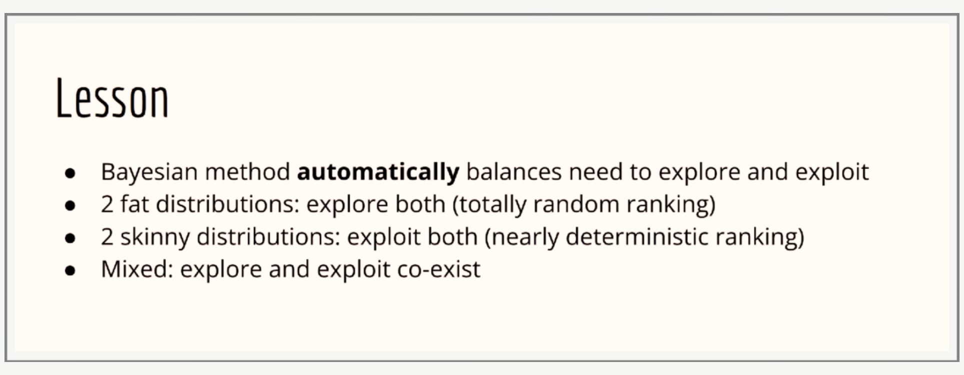
Scenario 3

Now, let's consider a mixed scenario where we have one item with a **very sharp peak** and one item with a **very fat peak**. In this case, the sample for the sharp peak will most likely be located at the peak. On the other hand, the item with the very fat peak could really have a sample anywhere.
As such, there is a good chance it may be ranked higher than the other item. But there's also a good chance it maybe ranks lower. And remember, many people are going to see this page, so there's going to be a mix of both. So sometimes item-one will be ranked higher than item-two and sometimes vice versa. And remember that this is good so that we can collect more data.
You can think of the fat item as being an **exploration mode** because we're still not very confident of its Click-Through-Rate.


Mixed Case (One Fat, One Skinny)

So to summarize what we have just learned, we've learned that the **Bayesian method automatically balances** the need to **explore** and the need to **exploit**.

- In **Scenario-1**, we saw that it handles the **need to explore** by essentially sorting items in a completely random fashion.
- In **Scenario-2**, we saw that it handles the **need to exploit** by essentially sorting items deterministically by their known stars.
- In **Scenario-3**, we saw that these two extremes i.e the **need to explore** and **need to exploit** can coexist—we have one item which needs more exploration because it needs more data, and one item which can be exploited because it already has enough data.


Lesson

- Bayesian method **automatically** balances need to explore and exploit
- 2 fat distributions: explore both (totally random ranking)
- 2 skinny distributions: exploit both (nearly deterministic ranking)
- Mixed: explore and exploit co-exist

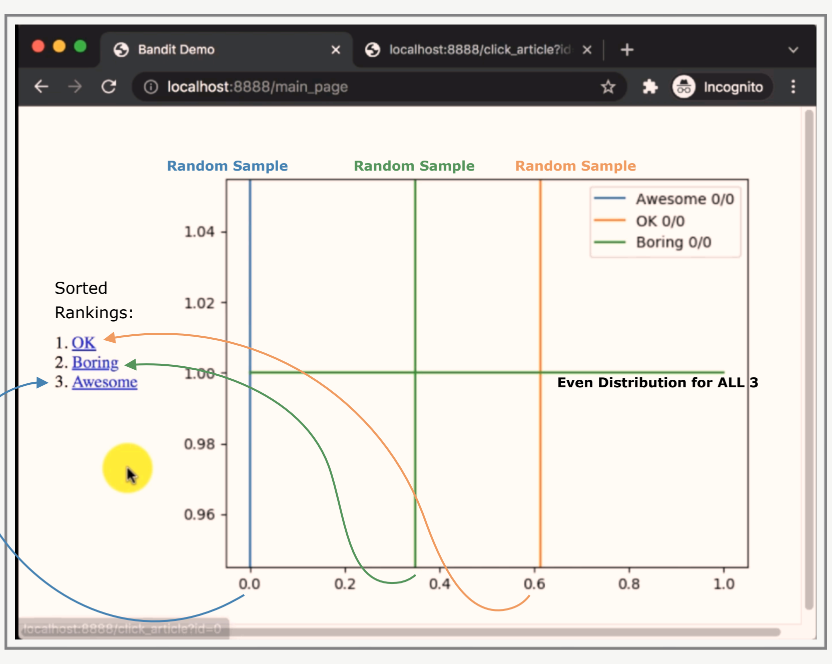
In other words, this solves the Explore Exploit dilemma. So the nice thing about this method is that it is completely automatic. You don't need to run an **A/B Test** to figure out which item is best. You can just run this algorithm and everything is ranked automatically in an optimal way.

Example

OK, so that was still pretty theoretical. But what really helps is to see this in action. I'll begin by telling you that we have three items. Awesome, OK and boring. Now, just to be clear, in case it's not obvious:

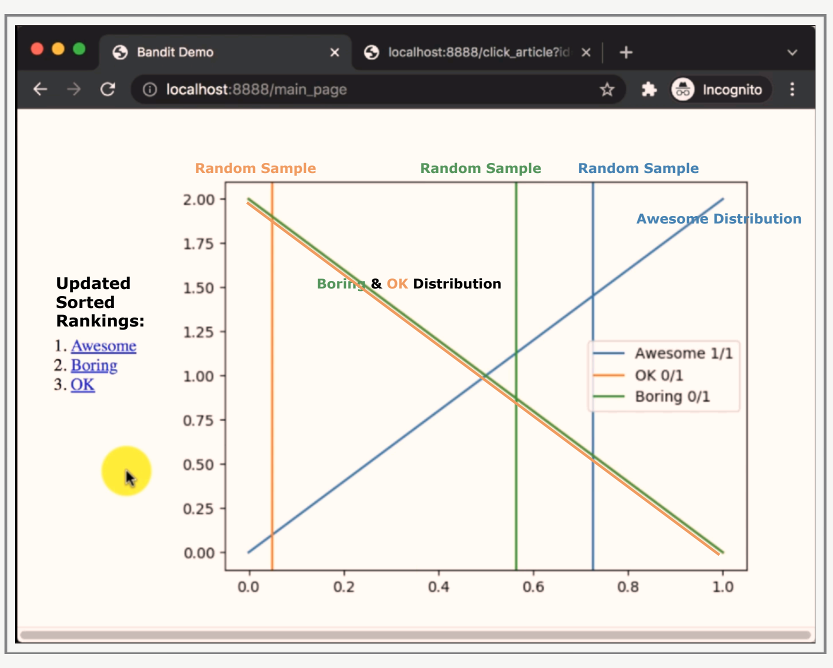
- the **Awesome product** is intended to be the products that people are most likely to click on.
- the **OK product** is a bit less likely to be clicked on.
- the **Boring product** is even less likely to be clicked on.

So what I've done here is I've created a dummy website which just shows these three items in an ordered list. You can pretend this is like Amazon.



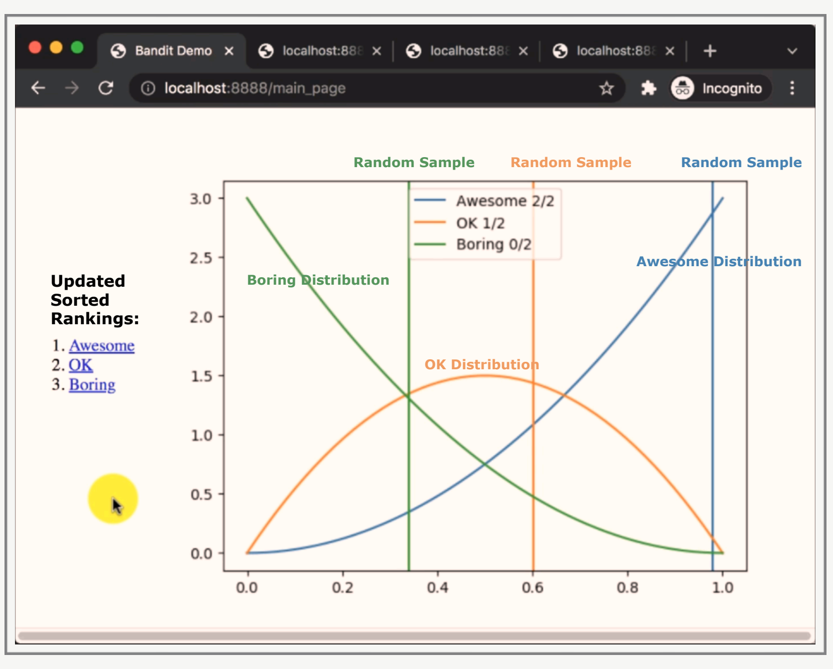
Beside the list is a graph showing us the current distribution of each item in addition, and notice that they also have a corresponding vertical bars which tell us the values of the samples. Remember that the samples are random, they are drawn from the corresponding distributions. So the green vertical line was sampled from the green distribution and so forth. So currently, what we can see is that the distribution for each item is flat. We know nothing about their CTRs because no one has seen this website yet. And notice how the items are ordered pretty randomly. OK is first, boring is second and awesome is last. Now, let's pretend we are the first user to ever visit this page—we click on the awesome item because we're interested in awesome products.

Now, let's refresh the page and pretend that we're the second person to ever visit this website.

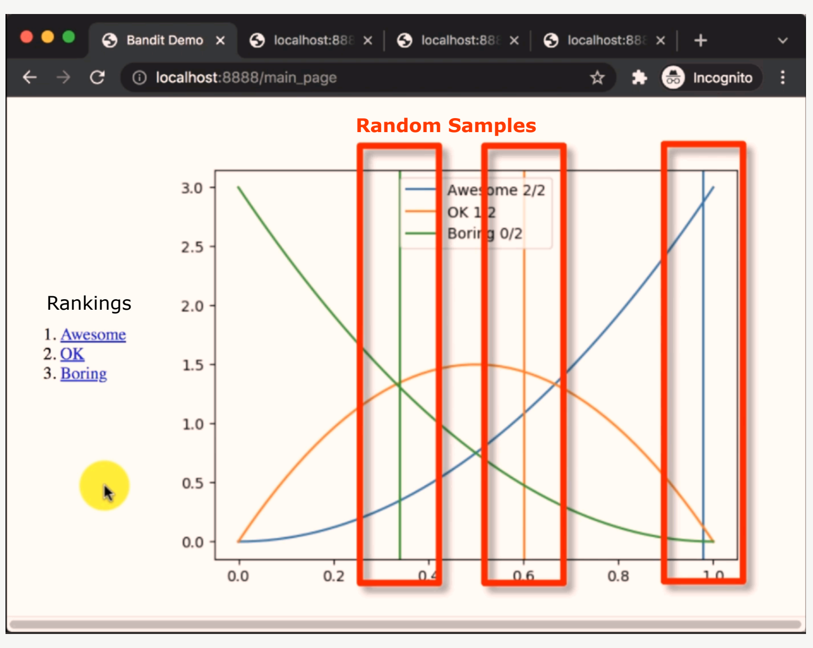


OK, so this is interesting. Now the distributions for all the items have changed. The awesome item is sloped upwards, while the OK and boring items are sloped downwards. This makes sense because nobody has clicked on the OK and boring items. We are completely confident that their CRTs cannot be 100 percent. But what about the rankings? This time, awesome is ranked at the top. Then we have boring and then OK. For boring and OK, they effectively have an equal chance of being in second and third place because they both have the same distribution.

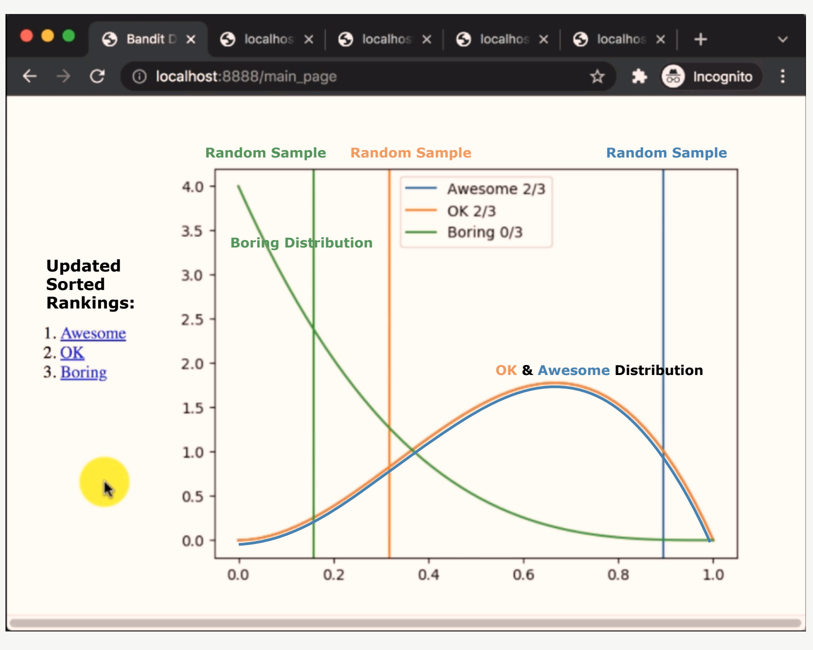
OK, so back to our user, let's say this user is interested in two products the awesome item and the OK item. So the user clicks on those. Now, let's refresh the page and pretend that we're the third user to ever visit this web site.



So now the distributions have been updated yet again, this time we can see that the boring item is still at zero, but with a steeper slope. Conversely, Awesome is still peaked at one, but with the opposite slope, OK has peaked in the middle, which makes sense because we saw it twice, but we only clicked at it once. Therefore, it's a maximum likelihood Click-Through-Rate is 0.5, but it's very wide because one out of two is not enough data to be confident.
In terms of rankings, they make sense given the distributions we have awesome, followed by OK, followed by boring.
Note that the samples we drew correspond to this ranking. Blue is the highest. Then orange, then green.

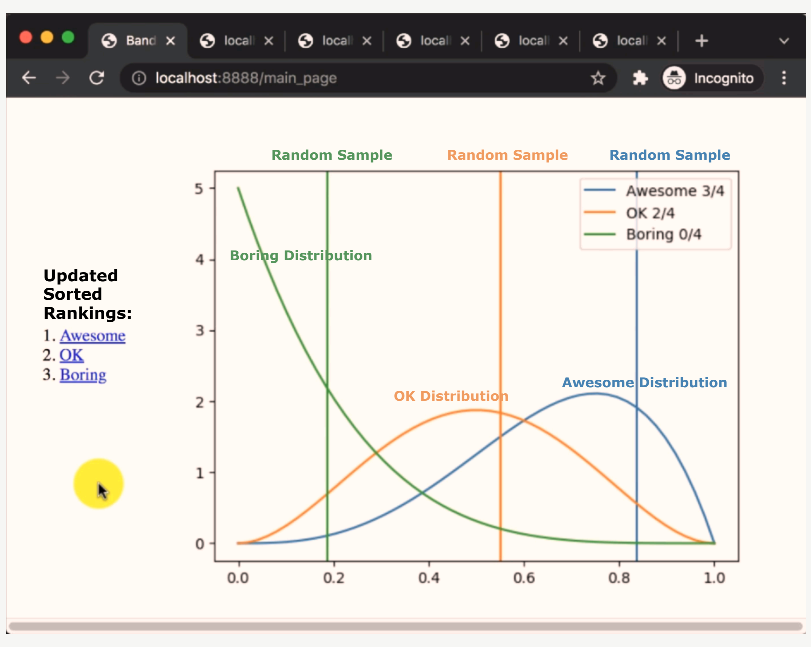


So now let's say this user is not interested in the awesome item, but is interested in the OK item. This is less probable, but it's not impossible. Not everyone will be interested in the awesome item. So let's refresh the page and check out what the next user would see.



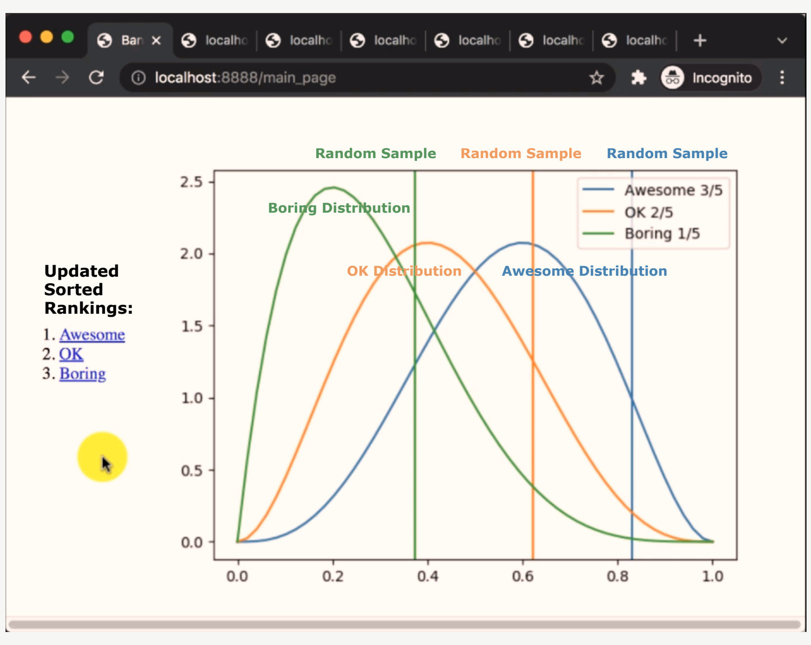
OK, so again, this makes sense. The distribution for both awesome and OK are the same now because they were both clicked. Two out of three times, the boring item has never been clicked and it's getting even more steep. Keeping its peak at zero. Notice how the rankings are in a sensible order: Awesome, OK, and then boring. But notice something interesting. The sample for OK is quite low. In fact, it's still very possible for the boring item to be ranked higher than the OK item, since their probability masses still overlap quite a bit.

In any case, let's pretend that this user is a typical user and they click on awesome. Now, let's refresh the page and see what the next user will see.

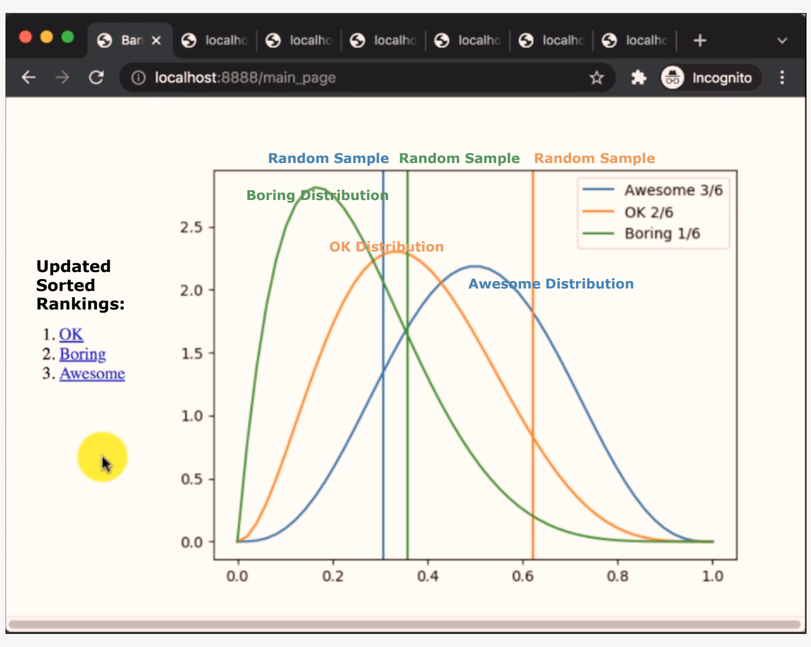


OK, so nothing really unexpected. The awesome peek is a bit higher than the OK peak since it's been clicked more often. The rankings are not unusual.

Now let's say this current user is atypical and they click on the boring item. Let's now refresh the page and see what the next user sees.

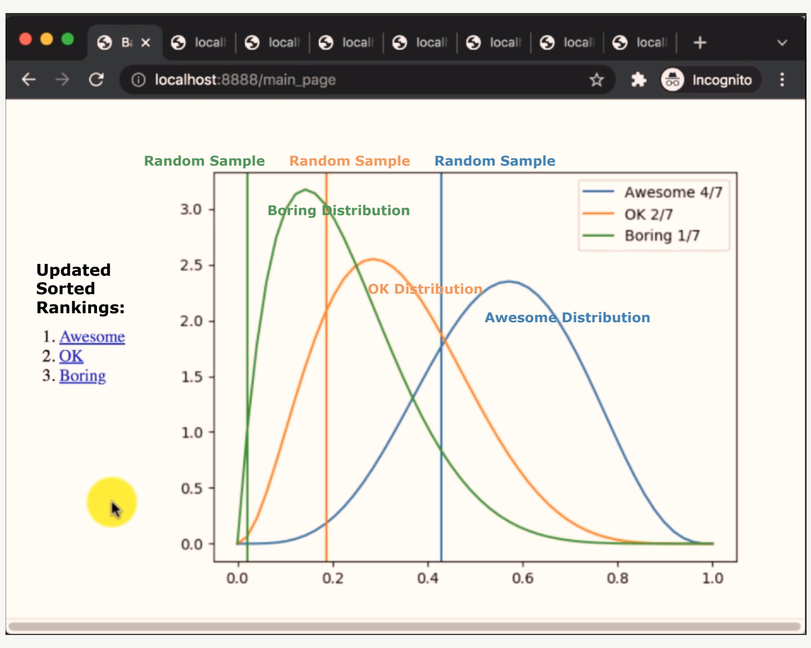


OK, so now we're entering interesting territory. There is a lot of overlap between the distributions, although the samples are in the not unusual order. There is a good chance they could be. So let's suppose this user doesn't click on anything. Let's refresh the page to see what the next user will see.

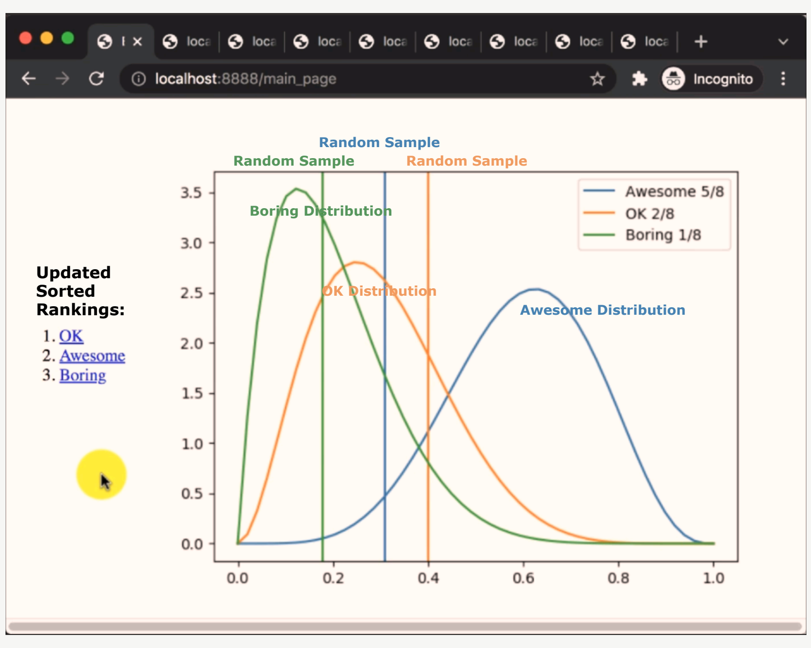


OK, so now the result is worth thinking about. Notice how awesome comes last. This is because these items are **not ordered by a deterministic ranking**. They are **ordered by random samples drawn from these distributions**. It just so happened that this time the awesome sample was very low, even lower than the boring sample. The OK sample turned out to be the highest.

Now, let's suppose that because the awesome item is so awesome, this user clicks on it. Now, let's refresh the page.

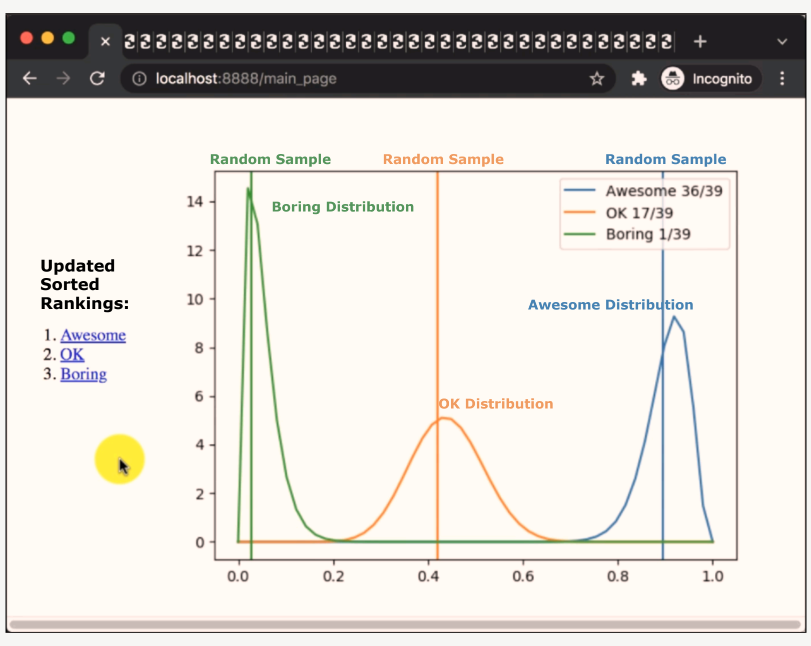


So this time the ordering is back to normal. Notice how all the distributions are moving. Let's click on the awesome item once again. And now let's refresh the page.



So importantly, notice that there is still enough overlap among the distributions that it's still possible for OK to overtake awesome.

OK, so what happens as we collect more and more clicks, what do we see is that these distributions get skinnier and skinnier as we collect more data? And because they are skinnier, **their samples have less variance and they are more likely to be close to their peaks**.



In other words, after we explore, we can then exploit by showing the true ranking of the items. **So over time, the rankings become more deterministic**.

Remember that we want the awesome item to be at the top because people want to click on this and purchase this product. This algorithm we just used happens to nicely balance exploration with exploitation so that we can collect data while sorting the items in an optimal way.