

Multi-Armed *Bandits*

CSE 312 26Su
Lecture 22



The Problem

There are K slot machines ("bandits" with "arms").

"Bandits" because they steal your money

You pull arms T times, where the arm you pull at time t (*arm pulled at time t is $a_t \in \{1, \dots, K\}$*) will return a random reward to you



The Problem

Your goal? Win as much money as possible! 🎰 That is, maximize your total expected reward after T pulls of an arm of the K slot machines.

Question: How do you know which arm to pull (based on information you have – the past) to achieve your goal?

Assumptions:

1. Rewards from each arm are independent.
2. The reward distribution of an arm does not change over time.

If you knew the reward distribution...

You have the below $K = 3$ slot machines, whose distribution of rewards you know. What strategy are you going to use to maximize your total (expected) reward?



$Poi(\lambda = 1.36)$



$Bin(n = 10, p = 0.4)$



$\mathcal{N}(\mu = -1, \sigma^2 = 4)$

If you knew the reward distribution...

You have the below $K = 3$ slot machines, whose distribution of rewards you know. What strategy are you going to use to maximize your total (expected) reward?

We compute **expected reward** per pull: 1.36, 4, -1 . So, pull arm 2 all times!



$Poi(\lambda = 1.36)$



$Bin(n = 10, p = 0.4)$



$\mathcal{N}(\mu = -1, \sigma^2 = 4)$

Neat! Problem solved! Or not...

Aren't you glad you learnt calculate expectations so early in this course?
Oh wait...why would we know the reward distributions of slot machines?

We don't 😞

We need to... *estimate all K expectations.*

WHILE maximizing the total (expected) reward.



Our Problem (summarized)

There are k “arms”. We get some reward when we pull the arm, but we don’t know the distribution with which the rewards are given. Each time we pull an arm, we can observe the reward we get.

Our goal: Derive a strategy for picking which arm to pull to maximize the total reward based on observations from your previous pulls.

Before we go on...why is this important?

Lot's of problems (reinforcement learning) can be phrased as a bandit problem!

A/B Testing: Experiment with releasing a new feature, or test ads to maximize click rate.
Arms: feature(s)/ad(s) to test *Maximize:* total ratings/click rate

Networks: Adaptive routings for picking best route for each pieces of data.
Arms: available routes *Maximize:* transmission speed

Recommendations: K movies options. Recommend each person a movie and observe.
Arms: available movies *Maximize:* clicks/rating recommendations

Clinical Trials: K treatment options. For each patient, give treatment and observe results.
Arms: different possible treatments *Maximize:* number of patients healed

Very real life: For each meal, you choose food, and track your happiness after eating.
Arms: possible food options *Maximize:* total/average happiness

Why is this a challenging problem?

There's a tradeoff between...

Exploitation (act accordingly to what you know is going to work)
Pulling arms that we know are "good" based on reward history.

Exploration (explore other options that *might* increase reward)
Pulling other arms in case they could be "good" or "better".

Regret: The difference between the optimal, best possible total (expected) reward and the actual reward from our choices of T pulls

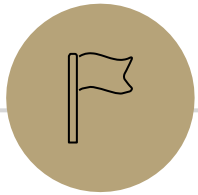
Simplification for the problem

In this case, let's say that each "arm" rewards either \$1, \$10, or \$100.

We don't know what the probabilities are though...so the PMF for the reward of arm i is:

$$p_{A_i}(k) = \begin{cases} 1 - \theta_1 - \theta_2 & k = 1 \\ \theta_1 & k = 10 \\ \theta_2 & k = 100 \end{cases}$$

Again, our goal is to find a strategy for picking an arm to pull that will maximize the total reward over the T pulls.



Strategy 1: Naïve, Greedy Approach

Strategy 1: Naïve, Greedy Approach

1. **Explore.** Pull each arm M times and record the reward from each
Based on data for each, estimate the parameters θ_1 and θ_2 for each arm
2. **Exploit.** Pick arm with highest estimated expected value and *only* use that arm for the remaining pulls.

1. Explore (*here, there are 3 arms, and $M = 2$*)

2. Exploit



Strategy 1: Naïve, Greedy Approach

1. **Explore.** Pull each arm M times and record the reward from each
Based on data for each, estimate the parameters θ_1 and θ_2 for each arm
How do we do this.....Maximum Likelihood Estimation!
2. **Exploit.** Pick arm with highest estimated expected value and *only* use that arm for the remaining pulls.

1. Explore (here, there are 3 arms, and $M = 2$)

2. Exploit



Strategy 1: Naïve, Greedy Approach

Pull arm 1 35 times and record the reward from each.

We see rewards $x_1, x_2, \dots, x_{35}$. Out of these, we get \$1 5 times, \$10 10 times and \$100 20 times. What is the MLE for θ_1 and θ_2 for arm 1?

1. Likelihood Function

$$p_{A_i}(k) = \begin{cases} 1 - \theta_1 - \theta_2 & k = 1 \\ \theta_1 & k = 10 \\ \theta_2 & k = 100 \end{cases}$$

2. Log-likelihood Function

3. Derivative(s) of Log-likelihood Function

....

4. Set the derivative(s) to 0 and solve for MLE(s)

5. Second derivative Test

- > Next step would be to repeat this process and get estimates all arms
- > Use the estimates to compute the expected reward from each
- > Pick the arm with highest expected reward for the remaining pulls

Strategy 1: Naïve, Greedy Approach

Pull arm 1 35 times and record the reward from each.

We see rewards $x_1, x_2, \dots, x_{35}$. Out of these, we get \$1 5 times, \$10 10 times and \$100 20 times. What is the MLE for θ_1 and θ_2 for arm 1?

1. Likelihood Function

$$\mathcal{L}(x_1, \dots, x_n; \theta_1, \theta_2) = (\theta_1)^{10} (\theta_2)^{20} (1 - \theta_1 - \theta_2)^5$$

$$p_{A_i}(k) = \begin{cases} 1 - \theta_1 - \theta_2 & k = 1 \\ \theta_1 & k = 10 \\ \theta_2 & k = 100 \end{cases}$$

2. Log-likelihood Function

$$\ln(\mathcal{L}(x_1, \dots, x_n; \theta_1, \theta_2)) = 10 \cdot \ln(\theta_1) + 20 \cdot \ln(\theta_2) + 5 \cdot \ln(1 - \theta_1 - \theta_2)$$

3. Derivative(s) of Log-likelihood Function

$$\text{Partial derivative w.r.t. } \theta_1: \frac{\partial}{\partial \theta_1} \left(\ln(\mathcal{L}(x_1, \dots, x_n; \theta_1, \theta_2)) \right) = \frac{10}{\theta_1} - \frac{5}{1 - \theta_1 - \theta_2}$$

$$\text{Partial derivative w.r.t. } \theta_2: \frac{\partial}{\partial \theta_2} \left(\ln(\mathcal{L}(x_1, \dots, x_n; \theta_1, \theta_2)) \right) = \frac{20}{\theta_2} - \frac{5}{1 - \theta_1 - \theta_2}$$

4. Set the derivative(s) to 0 and solve for MLE(s)

$$\frac{10}{\widehat{\theta}_1} - \frac{5}{1 - \widehat{\theta}_1 - \widehat{\theta}_2} = 0 \text{ and } \frac{20}{\widehat{\theta}_2} - \frac{5}{1 - \widehat{\theta}_1 - \widehat{\theta}_2} = 0. \text{ Solving system of equations... } \widehat{\theta}_1 = \frac{10}{35}, \widehat{\theta}_2 = \frac{20}{35}$$

5. Second derivative Test

Strategy 1: Naïve, Greedy Approach

Pull arm 1 35 times and record the reward from each.

We see rewards $x_1, x_2, \dots, x_{35}$. Out of these, we get \$1 5 times, \$10 10 times and \$100 20 times. What is the MLE for θ_1 and θ_2 for arm 1?

$$\widehat{\theta}_1 = \frac{10}{35}, \widehat{\theta}_2 = \frac{20}{35}$$

How good is this estimator? Is it biased?

$$p_{A_i}(k) = \begin{cases} 1 - \theta_1 - \theta_2 & k = 1 \\ \theta_1 & k = 10 \\ \theta_2 & k = 100 \end{cases}$$

1. Write a generalized form of the estimator

2. Check if it's unbiased

Strategy 1: Naïve, Greedy Approach

Pull arm 1 35 times and record the reward from each.

We see rewards $x_1, x_2, \dots, x_{35}$. Out of these, we get \$1 5 times, \$10 10 times and \$100 20 times. What is the MLE for θ_1 and θ_2 for arm 1?

$$\widehat{\theta}_1 = \frac{10}{35}, \widehat{\theta}_2 = \frac{20}{35}$$

$$p_{A_i}(k) = \begin{cases} 1 - \theta_1 - \theta_2 & k = 1 \\ \theta_1 & k = 10 \\ \theta_2 & k = 100 \end{cases}$$

How good is this estimator? Is it biased?

1. Write a generalized form of the estimator

$$\text{Let } X_i \sim \text{Ber}(\theta_1) \text{ and } Y_i \sim \text{Ber}(\theta_2) \rightarrow \widehat{\theta}_1 = \frac{\sum_{i=1}^{35} X_i}{35}, \widehat{\theta}_2 = \frac{\sum_{i=1}^{35} Y_i}{35}$$

2. Check if it's unbiased

$$\mathbb{E}[\widehat{\theta}_1] = \mathbb{E}\left[\frac{\sum_{i=1}^{35} X_i}{35}\right] = \frac{\sum_{i=1}^{35} \mathbb{E}[X_i]}{35} = \theta_1 \quad \checkmark$$

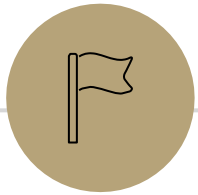
$$\mathbb{E}[\widehat{\theta}_2] = \mathbb{E}\left[\frac{\sum_{i=1}^{35} Y_i}{35}\right] = \frac{\sum_{i=1}^{35} \mathbb{E}[Y_i]}{35} = \theta_2 \quad \checkmark$$

Problems with this approach

We may not get an accurate idea from our first M pulls of each arm

- > If we choose the wrong best arm, we'd regret it for the rest of time!
- > If we increase M , we are spending more time on sub-optimal arms

Problem: We did all exploration, and then all exploitation.
Why don't we blend the two a bit more?



Strategy 2: Epsilon-Greedy

Strategy 2: Epsilon Greedy

1. **Explore.** Pull each arm M times and record the reward from each (same as before)
2. **Exploit (with a mix of exploration!).** With a small probability of ε , try a random other arm. Otherwise calculate "best arm" *Based on data for each, estimate the parameters θ_1 and θ_2 for each arm and pick best arm with highest estimated expected value.*

Better!

- > continuously updates estimated expected reward when it is pulled
- > explores with some probability ε which allows *you* to choose how to balance exploration and exploitation.

Still some problems!

Is uniform exploration the optimal policy?
Is a higher estimate *always* a better choice?

We have our estimates $\hat{\theta}_1$ and $\hat{\theta}_2$ for each arm that we use to find the expected reward for each arm. As we see more data, we can update these estimates.

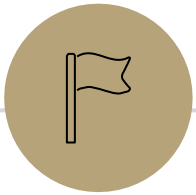
But, for example...

> After 300 samples from arm 1, $\hat{\theta}_1 = 0.3$

> After 3 samples from arm 2, $\hat{\theta}_1 = 0.2$

are very different! In this case, arm 2 still has a *potential* to have a much higher true probability of getting \$100, but with arm 1, it's less likely

We don't want to explore these equally!



Strategy 3: Upper Confidence Bound

Explore arms that have a higher *potential* of a better expectation

Confidence Interval for Our Estimates

Instead of picking the arm with the highest expected value, pick the arm with the **highest *potential* for expected value?**

How to calculate “potential”? Use a confidence interval!

For each of the estimated parameters, what range can we be 95% sure the *true parameter* lies in?

$$\begin{aligned} \text{E.g. for arm 1, } \widehat{\theta}_1 = 0.2 &\rightarrow \theta_1 \in [0.2 - 0.15, 0.2 + 0.15] = [0.05, 0.35] \\ \widehat{\theta}_2 = 0.3 &\rightarrow \theta_2 \in [0.3 - 0.1, 0.3 + 0.1] = [0.2, 0.4] \end{aligned}$$

On the next slide, we’re going to just look at the estimate for one of the parameters, θ_2 , but ideally, we would also look at the other parameter, and use it to estimate the expected value $(1 - \widehat{\theta}_1 - \widehat{\theta}_2) + (10 \cdot \widehat{\theta}_1) + (100 \cdot \widehat{\theta}_2)$

Confidence Interval for Our Estimates

For an estimate $\widehat{\theta}_2$ on arm i after seeing 35 samples, what is the smallest value of a such that the distance between the true θ_2 and the estimate $\widehat{\theta}_2$ is at most a with at least 95% confidence?

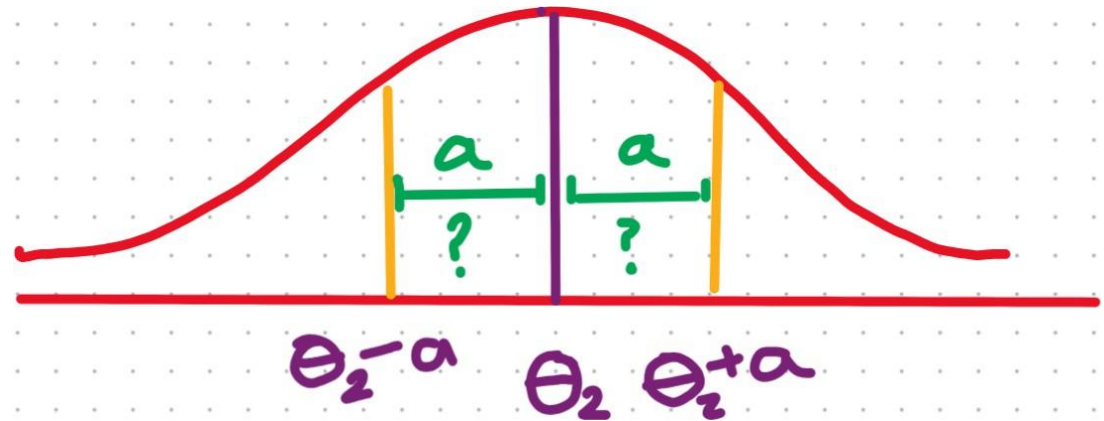
Translating to math notation...

$$\mathbb{P}(\theta_2 - a \leq \widehat{\theta}_2 \leq \theta_2 + a) \geq 0.95$$

And what exactly is $\widehat{\theta}_2$ again?

$$\widehat{\theta}_2 = \frac{\sum_{i=1}^{35} Y_i}{35} = \sum_{i=1}^{35} \frac{Y_i}{35}, \text{ where } Y_i \sim \text{Ber}(\theta_2)$$

$\widehat{\theta}_2$ is a sum of i.i.d RVs! So, what can we use to solve for a ?



Outline of CLT steps

1. **Setup the problem** (e.g., $X = \sum_{i=1}^n X_i$, X_i are i.i.d., and we want $\mathbb{P}(X \leq k)$)

Write event you are interested in, in terms of sum of random variables.

★ Apply *continuity correction* here if RVs are discrete.

2. **Apply CLT** (e.g., approx X as $Y \sim N(n\mu, n\sigma^2)$ -> $\mathbb{P}(X \leq k) \approx \mathbb{P}(Y \leq k)$)

Approximate sum of RVs as normal with appropriate mean and variance

from here, we're working with a normal distribution, which we've worked with before!

3. **Compute probability approximation using Phi table**

> *Standardize* ($Z = \frac{Y-\mu}{\sigma}$) -> $\mathbb{P}(Y \leq k) = \mathbb{P}\left(\frac{Y-\mu}{\sigma} \leq \frac{k-\mu}{\sigma}\right) = \mathbb{P}\left(Z \leq \frac{k-\mu}{\sigma}\right)$

> *Write in terms of $\Phi(z) = \mathbb{P}(Z \leq z)$*

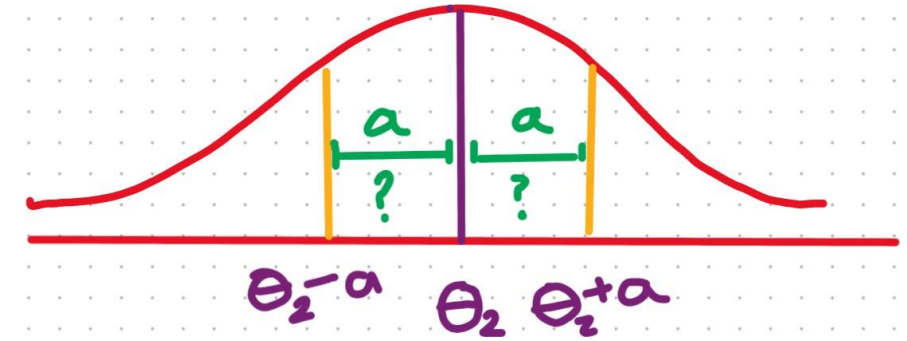
> *Look up in table*

Confidence Interval for Our Estimates

We have this MLE estimate: $\widehat{\theta}_2 = \frac{\sum_{i=1}^{35} Y_i}{35} = \sum_{i=1}^{35} \frac{Y_i}{35}$, where $Y_i \sim \text{Ber}(\theta_2)$

What is the value of a such that: $\mathbb{P}(\theta_2 - a \leq \widehat{\theta}_2 \leq \theta_2 + a) \geq 0.95$

1. Setup the problem



2. Apply CLT

3. Compute probability approximation using Phi table

Confidence Interval for Our Estimates

We have this MLE estimate: $\widehat{\theta}_2 = \frac{\sum_{i=1}^{35} Y_i}{35} = \sum_{i=1}^{35} \frac{Y_i}{35}$, where $Y_i \sim \text{Ber}(\theta_2)$

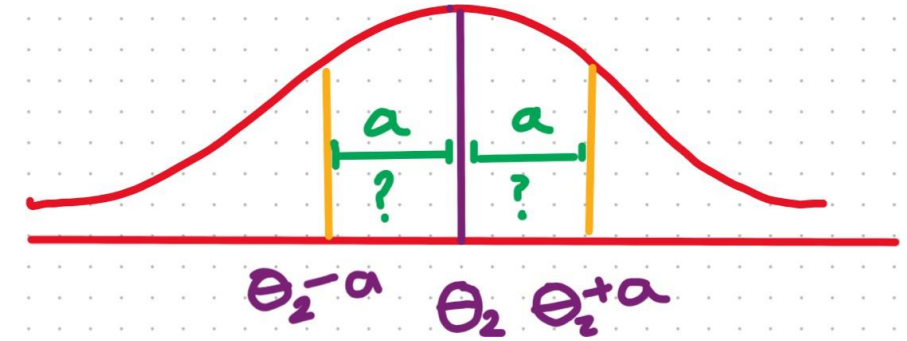
What is the value of a such that: $\mathbb{P}(\theta_2 - a \leq \widehat{\theta}_2 \leq \theta_2 + a) \geq 0.95$

1. Setup the problem

$$\widehat{\theta}_2 = \sum_{i=1}^{35} \frac{Y_i}{35},$$

where $E[Y_i] = \theta_2$, $\text{Var}(Y_i) = \theta_2(1 - \theta_2)$

$$\text{so } E\left[\frac{Y_i}{35}\right] = \frac{\theta_2}{35}, \text{Var}\left(\frac{Y_i}{35}\right) = \frac{\theta_2(1-\theta_2)}{35^2}$$



2. Apply CLT

Let $W \sim N\left(\theta_2, \frac{\theta_2(1-\theta_2)}{35}\right)$

By CLT, $\widehat{\theta}_2 \approx W$

3. Compute probability approximation using Phi table

$$\text{Let } \sigma = \frac{\theta_2(1-\theta_2)}{35}$$

$$\begin{aligned} P(\theta_2 - a \leq W \leq \theta_2 + a) &= P\left(\frac{\theta_2 - a - \theta_2}{\sigma} \leq Z \leq \frac{\theta_2 + a - \theta_2}{\sigma}\right) \\ &= \Phi\left(\frac{a}{\sigma}\right) - \Phi\left(-\frac{a}{\sigma}\right) \\ &= \Phi\left(\frac{a}{\sigma}\right) - \left(1 - \Phi\left(\frac{a}{\sigma}\right)\right) \\ &= 2\Phi\left(\frac{a}{\sigma}\right) - 1 \end{aligned}$$

$$2\Phi\left(\frac{a}{\sigma}\right) - 1 \geq 0.95$$

$$\Phi\left(\frac{a}{\sigma}\right) \geq 0.975$$

$$\frac{a}{\sigma} \geq 1.96$$

$$a \geq 1.96 \cdot \sqrt{\frac{\theta_2(1-\theta_2)}{35}}$$

Handling $\sqrt{\theta_2(1 - \theta_2)}$

Justification 1: If we make a mistake, we want it to be making n bigger. (since we're trying to say "take n at least this big, and you'll be safe").

The bigger the standard deviation, the bigger n will need to be to control it. So assume the biggest possible standard deviation.

Justification 2:

As $\sqrt{\theta_2(1 - \theta_2)}$ gets bigger, the interval gets smaller (it's in the denominator), so assuming the biggest value of $\sqrt{\theta_2(1 - \theta_2)}$ gives us the most restricted interval. So no matter what the true interval is we have a subset of it. And if our probability is at least .95 then the true probability is at least .95.

What's the maximum of $\sqrt{\theta_2(1 - \theta_2)}$?

Worst value of p

Calculus time!

$$\text{Set } \frac{d}{dp} \sqrt{p - p^2} = 0$$

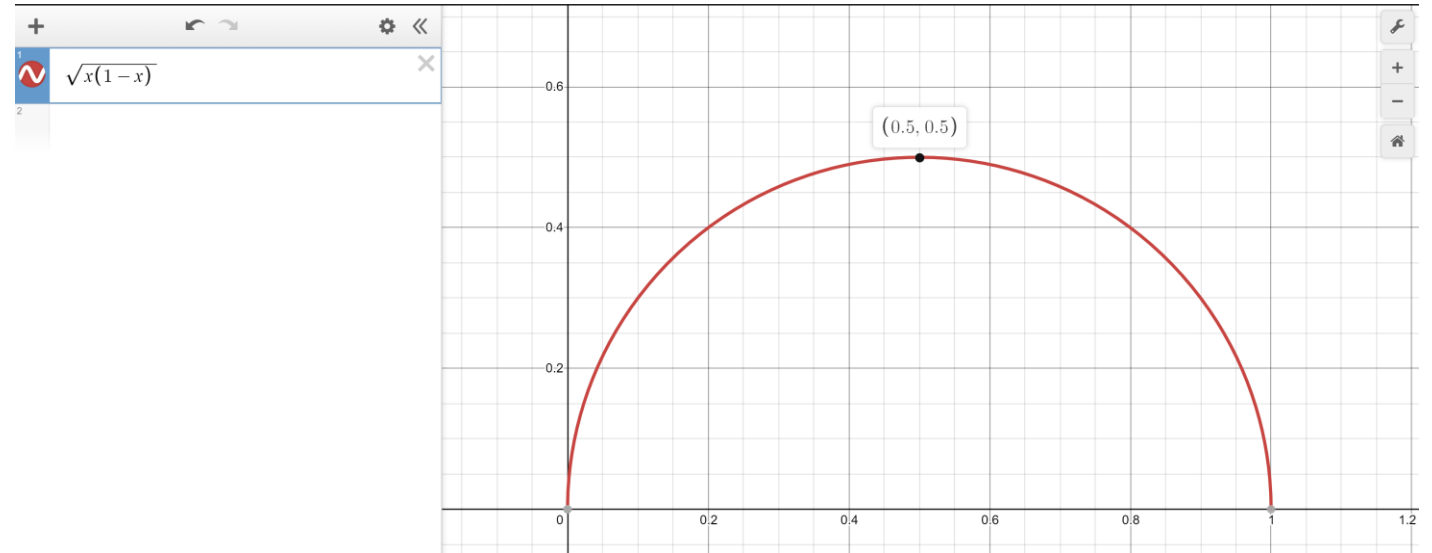
$$\frac{1}{\sqrt{p - p^2}} (1 - 2p) = 0$$

$$1 - 2p = 0 \rightarrow p = 1/2$$

Second derivative test will confirm $p = \frac{1}{2}$ is a maximizer

Or just plot it.

$$\sqrt{\frac{1}{2} \left(1 - \frac{1}{2}\right)} = \sqrt{1/4}.$$



Side note: A similar process can be used when we're trying to figure out things like how many people do we need to poll to be confident in our result when we know *NOTHING* about the true mean or variance of the population's votes!

Φ Table: $\mathbb{P}(Z \leq z)$ when $Z \sim \mathcal{N}(0, 1)$

z	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
0.0	0.5	0.50399	0.50798	0.51197	0.51595	0.51994	0.52392	0.5279	0.53188	0.53586
0.1	0.53983	0.5438	0.54776	0.55172	0.55567	0.55962	0.56356	0.56749	0.57142	0.57535
0.2	0.57926	0.58317	0.58706	0.59095	0.59483	0.59871	0.60257	0.60642	0.61026	0.61409
0.3	0.61791	0.62172	0.62552	0.6293	0.63307	0.63683	0.64058	0.64431	0.64803	0.65173
0.4	0.65542	0.6591	0.66276	0.6664	0.67003	0.67364	0.67724	0.68082	0.68439	0.68793
0.5	0.69146	0.69497	0.69847	0.70194	0.7054	0.70884	0.71226	0.71566	0.71904	0.7224
0.6	0.72575	0.72907	0.73237	0.73565	0.73891	0.74215	0.74537	0.74857	0.75175	0.7549
0.7	0.75804	0.76115	0.76424	0.7673	0.77035	0.77337	0.77637	0.77935	0.7823	0.78524
0.8	0.78814	0.79103	0.79389	0.79673	0.79955	0.80234	0.80511	0.80785	0.81057	0.81327
0.9	0.81594	0.81859	0.82121	0.82381	0.82639	0.82894	0.83147	0.83398	0.83646	0.83891
1.0	0.84134	0.84375	0.84614	0.84849	0.85083	0.85314	0.85543	0.85769	0.85993	0.86214
1.1	0.86433	0.8665	0.86864	0.87076	0.87286	0.87493	0.87698	0.879	0.881	0.88298
1.2	0.88493	0.88686	0.88877	0.89065	0.89251	0.89435	0.89617	0.89796	0.89973	0.90147
1.3	0.9032	0.9049	0.90658	0.90824	0.90988	0.91149	0.91309	0.91466	0.91621	0.91774
1.4	0.91924	0.92073	0.9222	0.92364	0.92507	0.92647	0.92785	0.92922	0.93056	0.93189
1.5	0.93319	0.93448	0.93574	0.93699	0.93822	0.93943	0.94062	0.94179	0.94295	0.94408
1.6	0.9452	0.9463	0.94738	0.94845	0.9495	0.95053	0.95154	0.95254	0.95352	0.95449
1.7	0.95543	0.95637	0.95728	0.95818	0.95907	0.95994	0.9608	0.96164	0.96246	0.96327
1.8	0.96407	0.96485	0.96562	0.96638	0.96712	0.96784	0.96856	0.96926	0.96995	0.97062
1.9	0.97128	0.97193	0.97257	0.9732	0.97381	0.97441	0.975	0.97558	0.97615	0.9767
2.0	0.97725	0.97778	0.97831	0.97882	0.97932	0.97982	0.9803	0.98077	0.98124	0.98169
2.1	0.98214	0.98257	0.983	0.98341	0.98382	0.98422	0.98461	0.985	0.98537	0.98574
2.2	0.9861	0.98645	0.98679	0.98713	0.98745	0.98778	0.98809	0.9884	0.9887	0.98899
2.3	0.98928	0.98956	0.98983	0.9901	0.99036	0.99061	0.99086	0.99111	0.99134	0.99158
2.4	0.9918	0.99202	0.99224	0.99245	0.99266	0.99286	0.99305	0.99324	0.99343	0.99361
2.5	0.99379	0.99396	0.99413	0.9943	0.99446	0.99461	0.99477	0.99492	0.99506	0.9952
2.6	0.99534	0.99547	0.9956	0.99573	0.99585	0.99598	0.99609	0.99621	0.99632	0.99643
2.7	0.99653	0.99664	0.99674	0.99683	0.99693	0.99702	0.99711	0.9972	0.99728	0.99736
2.8	0.99744	0.99752	0.9976	0.99767	0.99774	0.99781	0.99788	0.99795	0.99801	0.99807
2.9	0.99813	0.99819	0.99825	0.99831	0.99836	0.99841	0.99846	0.99851	0.99856	0.99861
3.0	0.99865	0.99869	0.99874	0.99878	0.99882	0.99886	0.99889	0.99893	0.99896	0.999

Upper Confidence Bound

Our strategy now is:

- > First sample K of each arm to get some initial data
- > Now, for every choice, we will use the previous data to figure out confidence intervals for each arm, use this to get ranges for the potential expectations of each arm, and pick the arm with the *highest upper bound*

Upper Confidence Bound

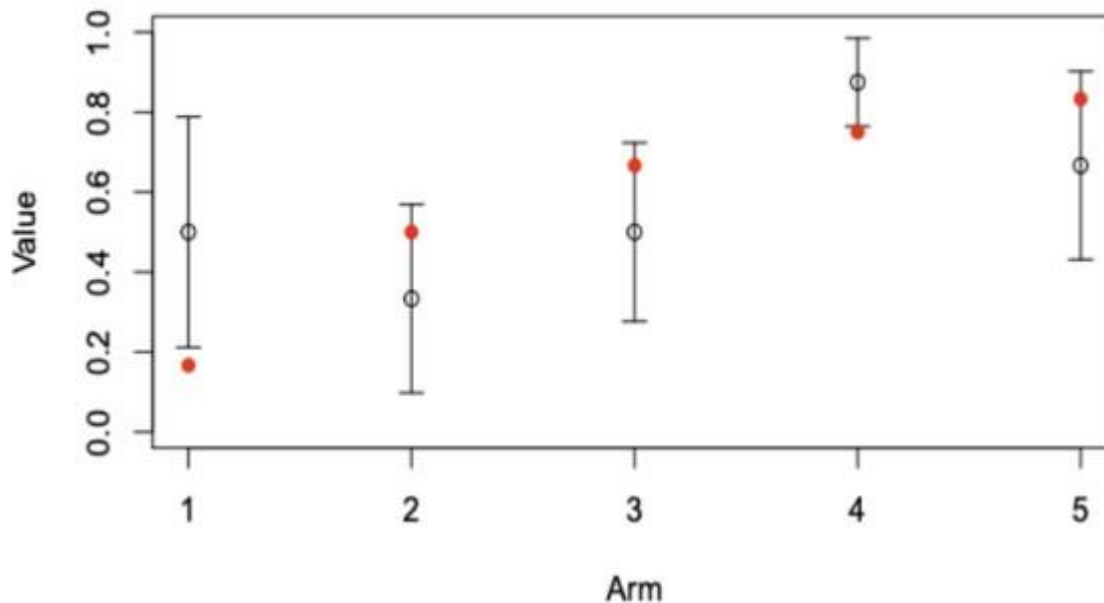
More frequent pulls from an arm --> smaller upper confidence bound

After seeing more observations, the variance of distribution decreases

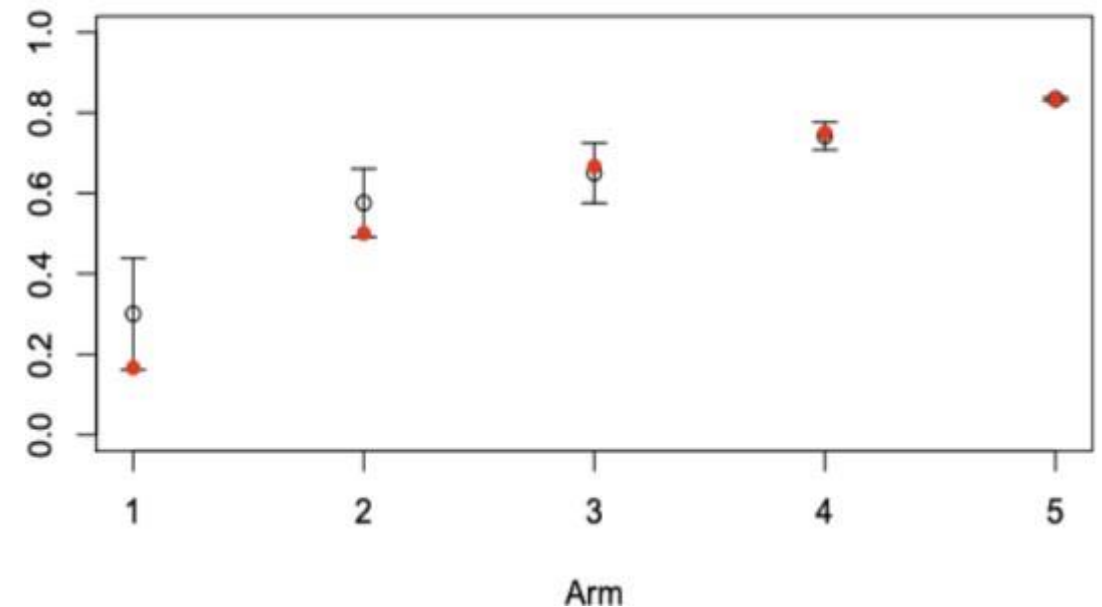
Less frequent pulls from an arm --> larger upper confidence bound

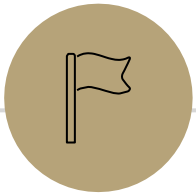
After not seeing many observations, more variation is still possible

Confidence Intervals for Mean of Each Arm: $t=10$



Confidence Intervals for Mean of Each Arm: $t=10000$





Strategy 4: Thompson Sampling

See Alex Tsun's textbook for details about this 😊

Muti-Armed Bandits aren't *perfect*

- > They don't personalize!
they only find overall "winners", but sometimes, different customers need different things. There are other external factors that affect what the best choice is
- > When there are too many arms, MABs don't work as fast and as well
e.g., if the arms are the time of day to send an email, if you find that if you send an email at 3pm, the customer buys the product, that tells you that 2pm and 4pm are also more likely to be better times

Contextual Bandits are the next step! If you're curious, here are some great resources for learning about this.

[article](#), [another article](#), [another *another* article](#)

If you like these topics, this is also the basis for *reinforcement learning*