

# Learning through Experimentation

CS246: Mining Massive Datasets

Jure Leskovec, Stanford University

<http://cs246.stanford.edu>



# Learning through Experimentation

## ■ Web advertising

- We discussed how to match advertisers to queries in real-time
- **But we did not discuss how to estimate the CTR (Click-Through Rate)**

## ■ Recommendation engines

- We discussed how to build recommender systems
- **But we did not discuss the cold-start problem**

A screenshot of a Google search results page for the query "squash rackets". The results are categorized under "Web Shopping". The first result is a link to "Just Rackets UK and Worldwide online ...". Below it are links to "Squash Gear - Squash Equipment - squash racquets - squash rackets ...", "Squash Rackets Badminton Rackets Tennis Rackets from UK Rackets ...", and "Tennis, Badminton & Squash Rackets, Shoes, Clothing, Bags, Grips ...". A red box highlights the right-hand sidebar labeled "Sponsored Links".

A screenshot of the Yahoo! News homepage. The top navigation bar includes links for HOME, U.S., WORLD, BUSINESS, ENTERTAINMENT, SPORTS, TECH, POLITICS, and SCIENCE. Below the navigation, there's a "Top Stories" section featuring news items with images and titles. One story is about the Mount Everest death toll reaching four. Another is about a prostitution scandal at the DEA. A third is about President Obama's statement on Afghanistan. A fourth is about a Rutgers University student sentenced for webcam spying.

# Learning through Experimentation

- What do CTR and cold-start have in common?
- With every ad we show/ product we recommend we gather more data about the ad/product
- Theme: Learning through experimentation

A screenshot of a Google search results page for the query "squash rackets". The results are filtered by "Web Shopping". The first result is a sponsored link from "Squash & Tennis Rackets from Just-Rackets UK and Worldwide online...". Below it are several organic search results for various websites like Slazenger, Dunlop, Prince, and Squash Gear, showing product details and prices. A red box highlights the sponsored link at the top.

A screenshot of the Yahoo! News homepage. The main navigation bar includes links for HOME, U.S., WORLD, BUSINESS, ENTERTAINMENT, SPORTS, TECH, POLITICS, and SCIENCE. Below the navigation, there's a "Top Stories" section with four news items: "Everest weekend death toll reaches 4", "Colombia Secret Service prostitution scandal spreads to DEA", "Obama: U.S. can't wait for Afghanistan to be 'perfect'", and "Why ex-Rutgers student got 30-day sentence in spycam case". Each story has a thumbnail image, a title, and a brief description.

# Example: Web Advertising

- Google's goal: Maximize revenue
- The old way: Pay by impression (CPM)
  - Best strategy: Go with the highest bidder
    - But this ignores the “effectiveness” of an ad
- The new way: Pay per click! (CPC)
  - Best strategy: Go with expected revenue
  - What's the expected revenue of ad  $a$  for query  $q$ ?
  - $E[\text{revenue}_{a,q}] = P(\text{click}_a \mid q) * \text{amount}_{a,q}$

Prob. user will click on ad  $a$  given  
that she issues query  $q$

(Unknown! Need to gather information)

Bid amount for  
ad  $a$  on query  $q$   
(Known)

# Other Applications

- **Clinical trials:**

- Investigate effects of different treatments while minimizing adverse effects on patients

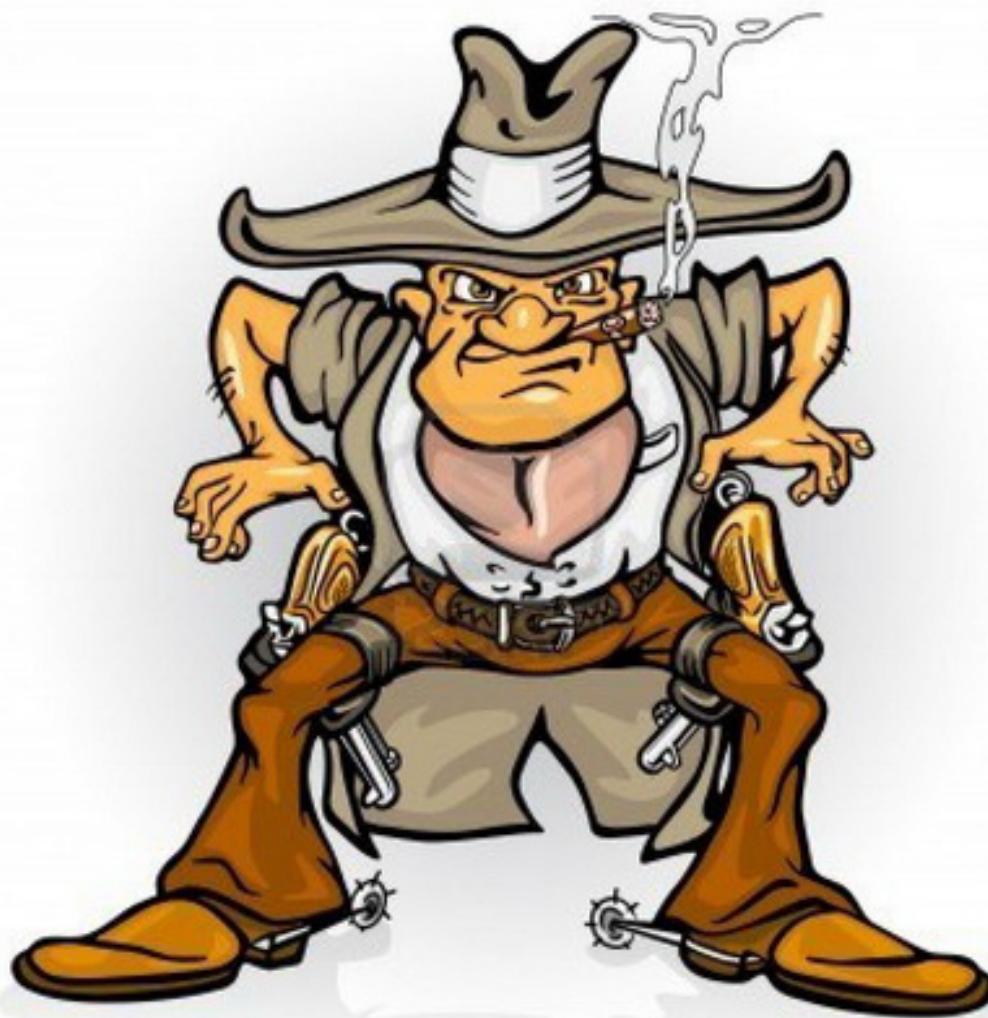
- **Adaptive routing:**

- Minimize delay in the network by investigating different routes

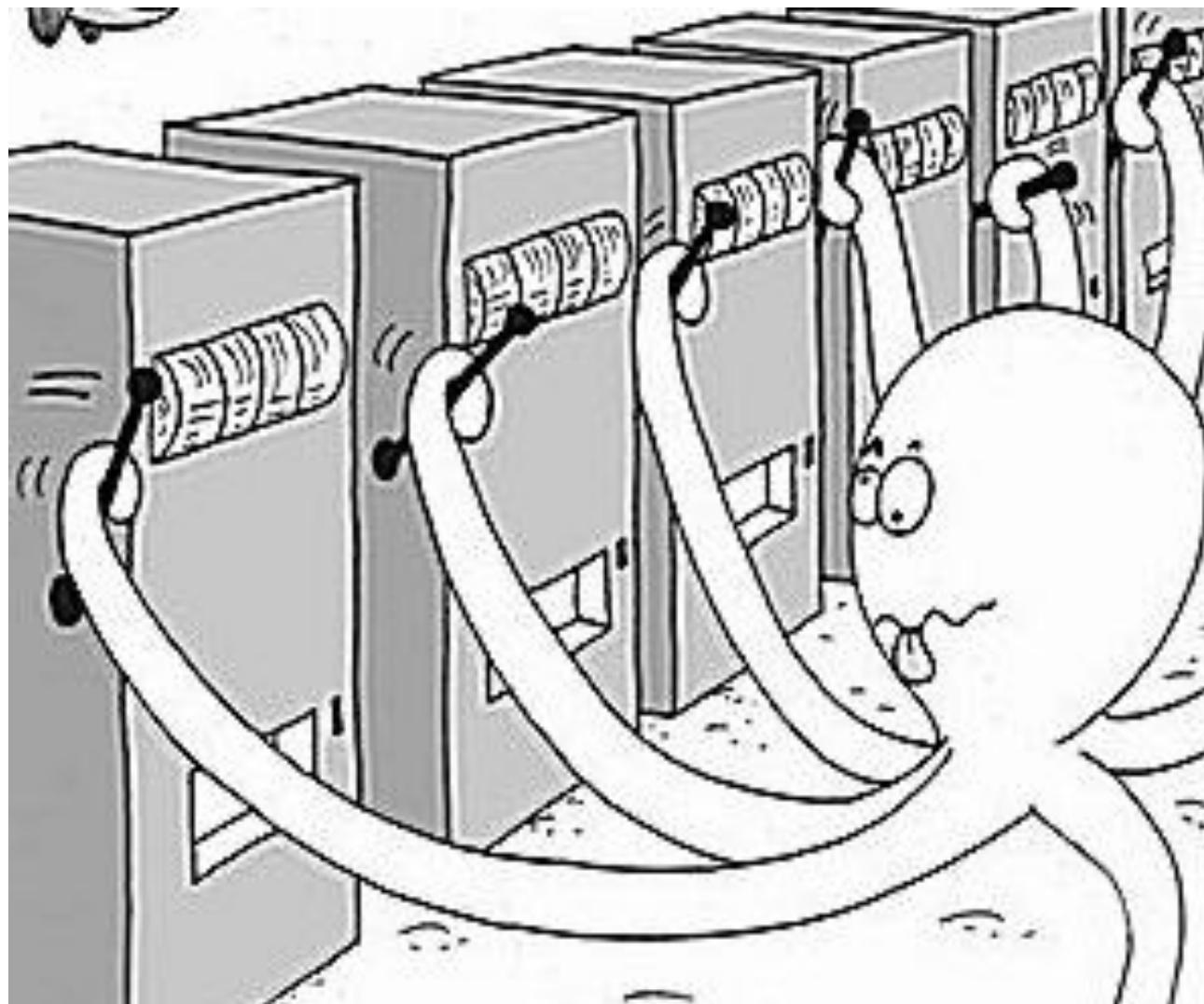
- **Asset pricing:**

- Figure out product prices while trying to make most money

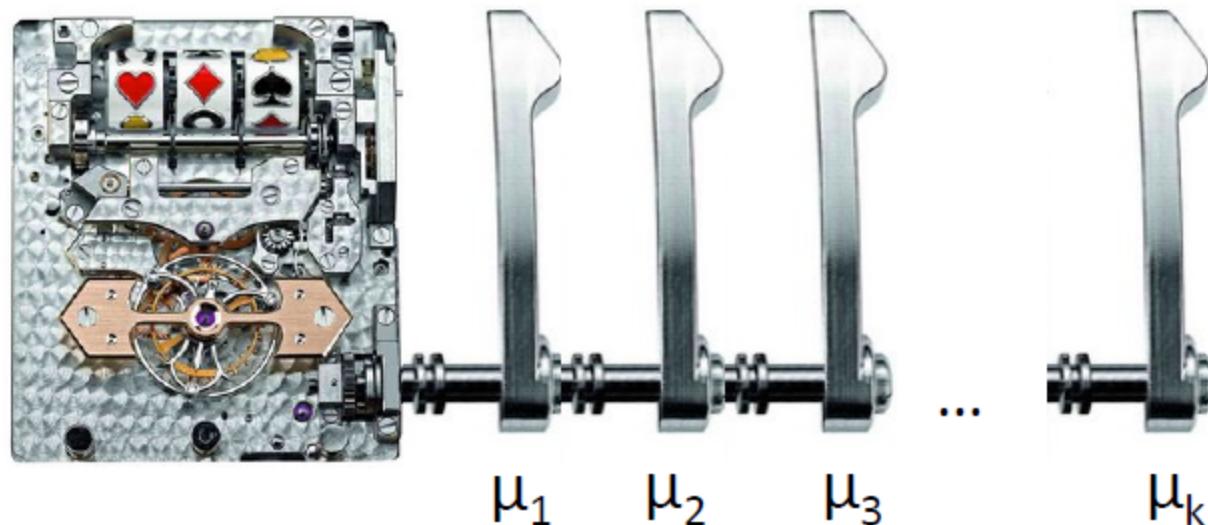
# Approach: Bandits



# Approach: Multiarmed Bandits

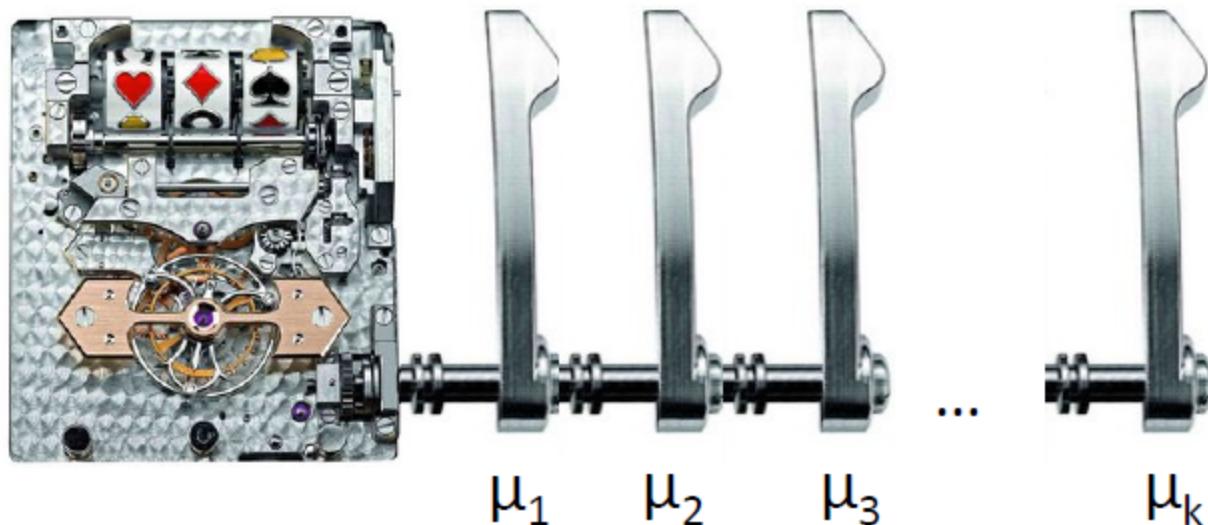


# k-Armed Bandit



- **Each arm  $a$** 
  - Wins (reward=1) with fixed (unknown) prob.  $\mu_a$
  - Loses (reward=0) with fixed (unknown) prob.  $1-\mu_a$
- All draws are independent given  $\mu_1 \dots \mu_k$
- **How to pull arms to maximize total reward?**

# k-Armed Bandit



- How does this map to our setting?
- Each query is a bandit
- Each ad is an arm
- We want to estimate the arm's probability of winning  $\mu_a$  (i.e., ad's CTR  $\mu_a$ )
- Every time we pull an arm we do an 'experiment'

# Stochastic k-Armed Bandit

## The setting:

- Set of  $k$  choices (arms)
- Each choice  $a$  is associated with unknown probability distribution  $P_a$  supported in  $[0,1]$
- We play the game for  $T$  rounds
- In each round  $t$ :
  - (1) We pick some arm  $a$
  - (2) We obtain random sample  $X_t$  from  $P_a$ 
    - Note reward is independent of previous draws
- Our goal is to maximize  $\sum_{t=1}^T X_t$
- But we don't know  $\mu_a$ ! But every time we pull some arm  $a$  we get to learn a bit about  $\mu_a$

# Online Optimization

- Online optimization with limited feedback

Choices	$X_1$	$X_2$	$X_3$	$X_4$	$X_5$	$X_6$	...
$a_1$					1	1	
$a_2$	0		1	0			
...							
$a_k$		0					

—————→ Time

- Like in online algorithms:

- Have to make a choice each time
- But we only receive information about the chosen action

# Solving the Bandit Problem

- **Policy:** a strategy/rule that in each iteration tells me which arm to pull
  - Hopefully policy depends on the history of rewards
- **How to quantify performance of the algorithm? Regret!**

# Performance Metric: Regret

- Let  $\mu_a$  be the mean of  $P_a$
- Payoff/reward of **best arm**:  $\mu^* = \max_a \mu_a$
- Let  $i_1, i_2 \dots i_T$  be the sequence of arms pulled
- Instantaneous **regret** at time  $t$ :  $r_t = \mu^* - \mu_{a_t}$
- **Total regret**:

$$R_T = \sum_{t=1}^T r_t$$

- **Typical goal:** Want a policy (arm allocation strategy) that guarantees:  $\frac{R_T}{T} \rightarrow 0$  as  $T \rightarrow \infty$ 
  - Note: Ensuring  $R_T/T \rightarrow 0$  is stronger than maximizing payoffs (minimizing regret), as it means that in the limit we discover the true best hand.

# Allocation Strategies

- If we knew the payoffs, which arm would we pull?

Pick  $\arg \max_a \mu_a$

- What if we only care about estimating payoffs  $\mu_a$ ?

- Pick each of  $k$  arms equally often:  $\frac{T}{k}$

- Estimate:  $\widehat{\mu}_a = \frac{k}{T} \sum_{j=1}^{T/k} X_{a,j}$

- Regret:  $R_T = \frac{T}{k} \sum_{a=1}^k (\mu^* - \widehat{\mu}_a)$

$X_{a,j}$  ... payoff received when pulling arm  $a$  for  $j$ -th time

# Bandit Algorithm: First try

- Regret is defined in terms of average reward
- So, if we can estimate avg. reward we can minimize regret
- Consider algorithm: *Greedy*  
Take the action with the highest avg. reward
  - Example: Consider 2 actions
    - A1 reward 1 with prob. 0.3
    - A2 has reward 1 with prob. 0.7
  - Play A1, get reward 1
  - Play A2, get reward 0 
  - Now avg. reward of A1 will never drop to 0, and we will never play action A2

# Exploration vs. Exploitation

- The example illustrates a classic problem in decision making:
  - We need to trade off between exploration (gathering data about arm payoffs) and exploitation (making decisions based on data already gathered)
- The Greedy algo does not explore sufficiently
  - Exploration: Pull an arm we never pulled before
  - Exploitation: Pull an arm  $a$  for which we currently have the highest estimate of  $\mu_a$

# Optimism

- The problem with our **Greedy** algorithm is that it is **too certain** in the estimate of  $\mu_a$ 
  - When we have seen a single reward of 0 we shouldn't conclude the average reward is 0
- **Greedy can converge to a suboptimal solution!**

# New Algorithm: Epsilon-Greedy

## Algorithm: Epsilon-Greedy

- For  $t=1:T$

- Set  $\varepsilon_t = O\left(\frac{1}{t}\right)$  (that is,  $\varepsilon_t$  decays over time  $t$  as  $1/t$ )
- With prob.  $\varepsilon_t$ : Explore by picking an arm chosen uniformly at random
- With prob.  $1 - \varepsilon_t$ : Exploit by picking an arm with highest empirical mean payoff
- Theorem [Auer et al. '02]

For suitable choice of  $\varepsilon_t$  it holds that

$$R_T = O(k \log T) \Rightarrow \frac{R_T}{T} = O\left(\frac{k \log T}{T}\right) \rightarrow 0$$

$k$ ...number  
of arms

# Issues with Epsilon-Greedy

- What are some issues with **Epsilon-Greedy**?
  - “**Not elegant**”: Algorithm explicitly distinguishes between exploration and exploitation
  - **More importantly**: Exploration makes **suboptimal choices** (since it picks any arm equally likely) ☰
- **Idea**: When exploring/exploiting we need to **compare** arms

# Comparing Arms

- Suppose we have done experiments:
  - Arm 1: 1 0 0 1 1 0 0 1 0 1
  - Arm 2: 1
  - Arm 3: 1 1 0 1 1 1 0 1 1 1
- Mean arm values:
  - Arm 1: 5/10, Arm 2: 1, Arm 3: 8/10
- Which arm would you pick next?
- Idea: Don't just look at the mean (that is, expected payoff) but also the confidence!

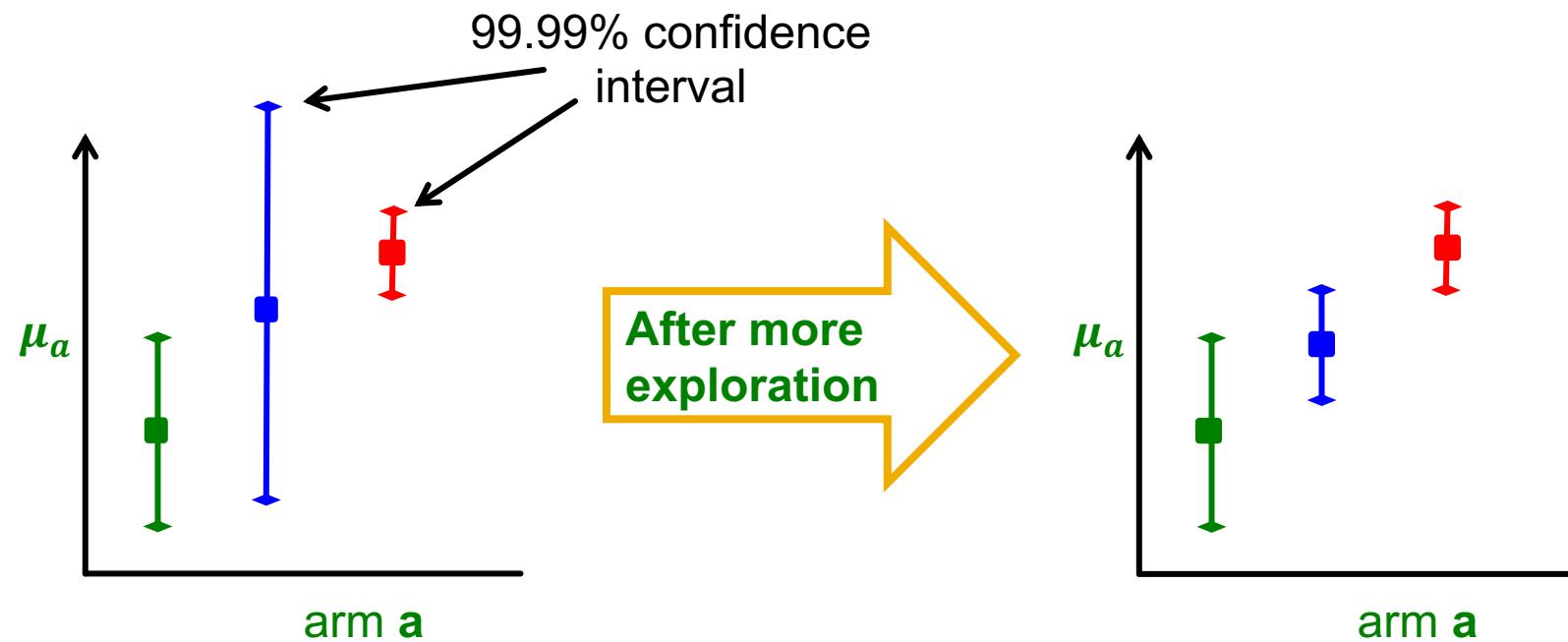
# Confidence Intervals (1)

- A confidence interval is a range of values within which we are sure the mean lies with a certain probability
  - We could believe  $\mu_a$  is within [0.2,0.5] with probability 0.95
  - If we would have tried an action less often, our estimated reward is less accurate so the confidence interval is larger
  - Interval shrinks as we get more information (try the action more often)

# Confidence Intervals (2)

- Assuming we know the confidence intervals
- Then, instead of trying the action with the highest mean we can try the action with the highest upper bound on its confidence interval
- This is called an optimistic policy
  - We believe an action is as good as possible given the available evidence

# Confidence Based Selection



# Calculating Confidence Bounds

Suppose we fix arm  $a$ :

- Let  $Y_{a,1} \dots Y_{a,m}$  be the payoffs of arm  $a$  in the first  $m$  trials
  - So,  $Y_{a,1} \dots Y_{a,m}$  are i.i.d. rnd. vars. taking values in  $[0,1]$
- Mean payoff of arm  $a$ :  $\mu_a = E[Y_{a,\cdot}]$
- Our estimate:  $\widehat{\mu}_{a,m} = \frac{1}{m} \sum_{\ell=1}^m Y_{a,\ell}$
- Want to find  $b$  such that with high probability  $|\mu_a - \widehat{\mu}_{a,m}| \leq b$ 
  - Want  $b$  to be as small as possible (so our estimate is close)
- Goal: Want to bound  $P(|\mu_a - \widehat{\mu}_{a,m}| \leq b)$

# Hoeffding's Inequality (1)

**Hoeffding's inequality** provides an upper bound on the probability that the average deviates from its expected value by more than a certain amount:

- Let  $X_1 \dots X_m$  be i.i.d. rnd. vars. taking values in  $[0,1]$
- Let  $\mu = E[X]$  and  $\widehat{\mu}_m = \frac{1}{m} \sum_{\ell=1}^m X_\ell$
- **Then:**  $P(|\mu - \widehat{\mu}_m| \geq b) \leq 2 \exp(-2b^2m) = 1 - \delta$ 
  - $\delta$ ... is the confidence level
- **To find out the confidence interval  $b$  (for a given confidence level  $\delta$ ) we solve:**
  - $2e^{-2b^2m} \leq \delta$  then  $-2b^2m \leq \ln(\delta/2)$
  - So:  $b \geq \sqrt{\frac{\ln(\frac{2}{\delta})}{2m}}$  

# Hoeffding's Inequality (2)

- $P(|\mu - \hat{\mu}_m| \geq b) \leq 2 \exp(-2b^2 m)$   
where  $b$  is our upper bound,  $m$  number of times we played the action
- Let's set  $b = b(a, T) = \sqrt{2 \log(T)/m_a}$
- Then:  $P(|\mu - \hat{\mu}_m| \geq b) \leq 2T^{-4}$  which converges to zero very quickly:
  - Notice:
    - If we don't play action  $a$ , its upper bound  $b$  increases
      - This means we never permanently rule out an action no matter how poorly it performs
    - Prob. our upper bound is wrong decreases with time  $T$

# UCB1 Algorithm

- **UCB1 (Upper confidence sampling) algorithm**

- Set:  $\widehat{\mu}_1 = \dots = \widehat{\mu}_k = 0$  and  $m_1 = \dots = m_k = 0$ 
  - $\widehat{\mu}_a$  is our estimate of payoff of arm  $a$
  - $m_a$  is the number of pulls of arm  $a$  so far
- For  $t = 1:T$ 
  - For each arm  $a$  calculate:  $UCB(a) = \widehat{\mu}_a + \alpha \sqrt{\frac{2 \ln t}{m_a}}$
  - Pick arm  $j = \arg \max_a UCB(a)$
  - Pull arm  $j$  and observe  $y_t$
  - Set:  $m_j \leftarrow m_j + 1$  and  $\widehat{\mu}_j \leftarrow \frac{1}{m_j} (y_t + (m_j - 1) \widehat{\mu}_j)$

Upper confidence  
interval (Hoeffding's  
inequality)

$\alpha$ ...is a free parameter trading off exploration vs. exploitation

# UCB1: Discussion

$$\blacksquare \quad UCB(a) = \widehat{\mu}_a + \alpha \sqrt{\frac{2 \ln t}{m_a}}$$

$$b \geq \sqrt{\frac{\ln\left(\frac{2}{\delta}\right)}{2m}}$$

- Confidence interval **grows** with the total number of actions  $t$  we have taken
- But **shrinks** with the number of times  $m_a$  we have tried arm  $a$
- This ensures each arm is tried infinitely often but still balances exploration and exploitation
- $\alpha$  plays the role of  $\delta$ :  $\alpha = f\left(\frac{2}{\delta}\right)$

$$P(|\mu - \widehat{\mu}_m| \geq b) = \delta$$

**“Optimism in face of uncertainty”:**

The algorithm believes that it can obtain extra rewards by reaching the unexplored parts of the state space

# Performance of UCB1

## ■ Theorem [Auer et al. 2002]

- Suppose optimal mean payoff is  $\mu^* = \max_a \mu_a$
- And for each arm let  $\Delta_a = \mu^* - \mu_a$
- Then it holds that

$$E[R_T] \leq \left[ 8 \sum_{a: \mu_a < \mu^*} \frac{\ln T}{\Delta_a} \right] + \left( 1 + \frac{\pi^2}{3} \right) \left( \sum_{i=a}^k \Delta_a \right)$$

$O(k \ln T)$

$O(k)$

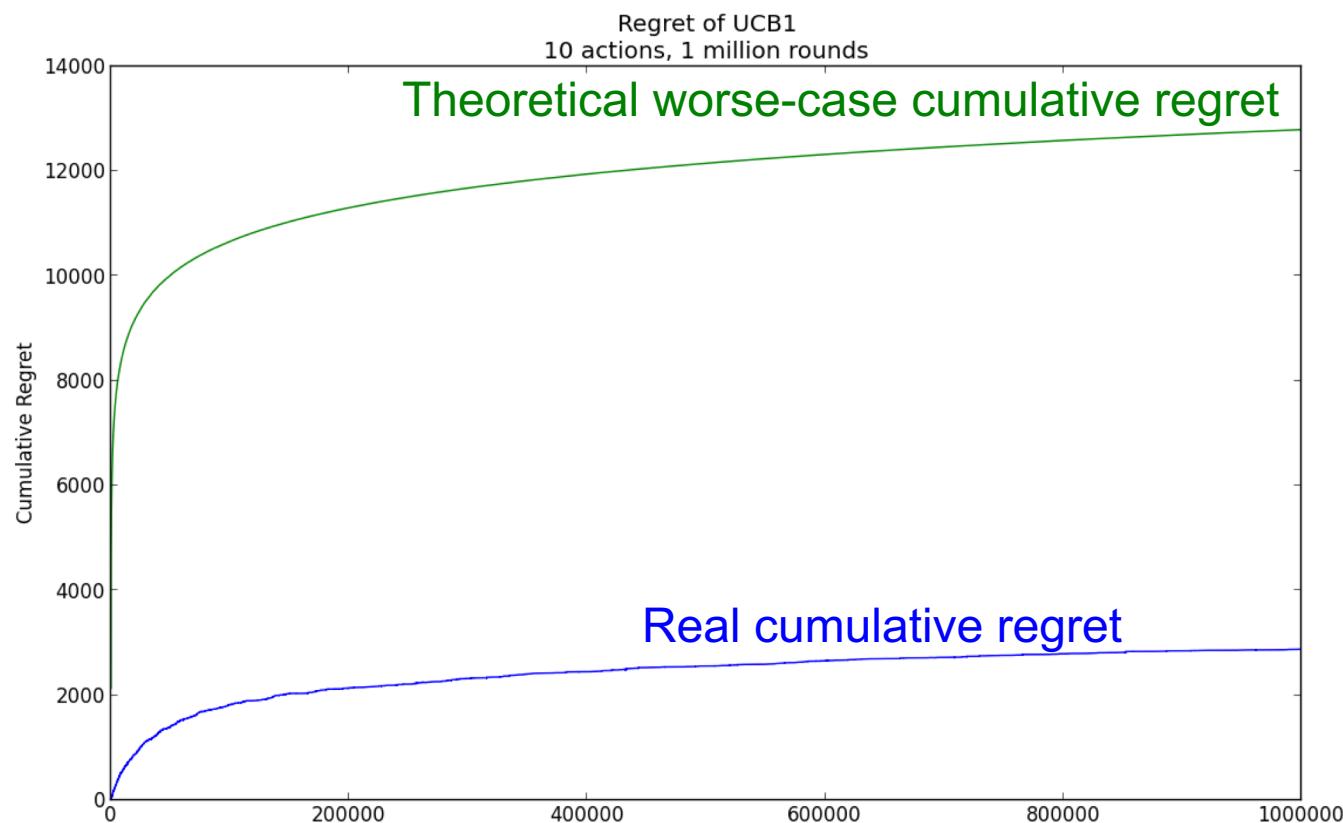
- So:  $O\left(\frac{R_T}{T}\right) \leq k \frac{\ln T}{T}$
- (note this is worst case regret)

# Summary so far

- $k$ -armed bandit problem as a formalization of the exploration-exploitation tradeoff
- Analog of online optimization (e.g., SGD, BALANCE), but with **limited feedback**
- **Simple algorithms are able to achieve no regret (in the limit)**
  - Epsilon-greedy
  - UCB (Upper Confidence Sampling)

# Example

- 10 actions, 1M rounds, uniform [0,1] rewards



# Use-case: Pinterest

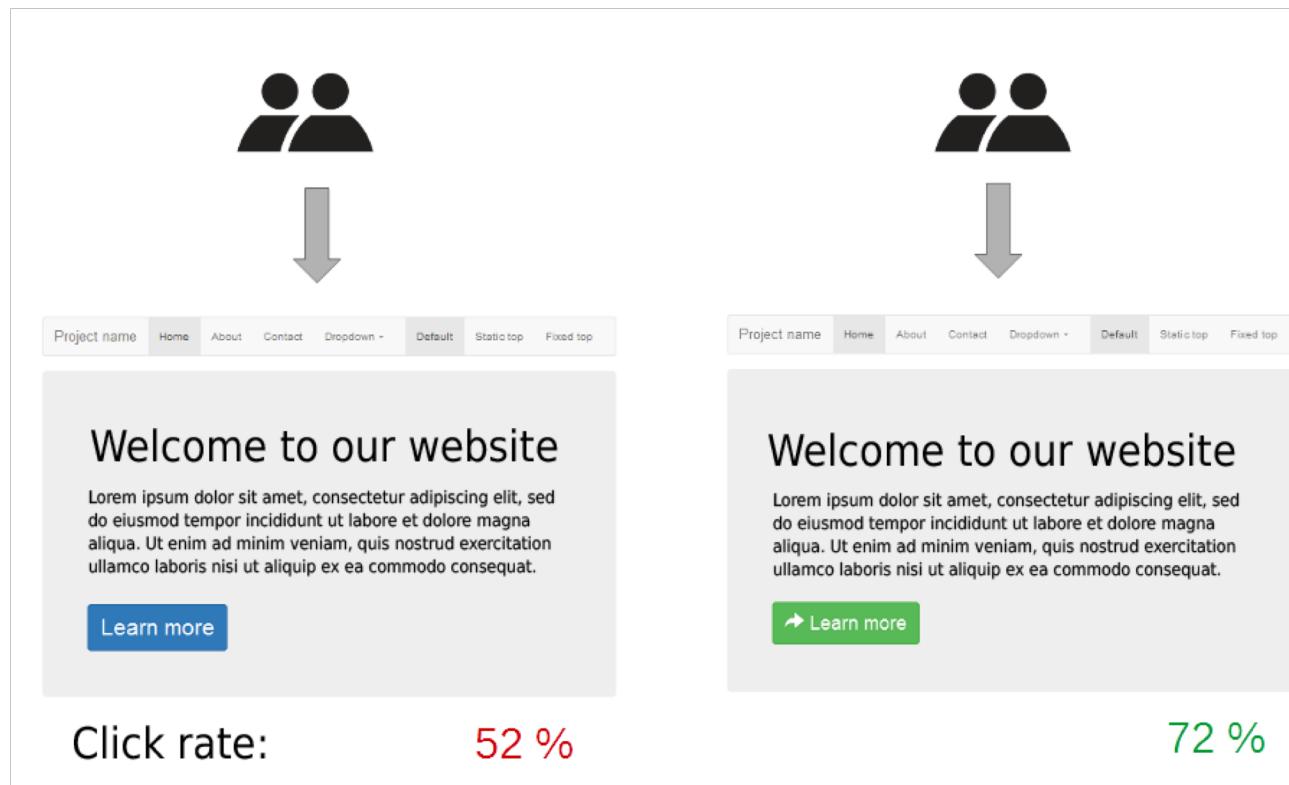
- **Problem:** For new pins/ads we do not have enough signal on how good they are
  - How likely are people to interact with them?
- **Idea:**
  - Try to maximize the rewards from several unknown slot machines by deciding which machines and the order to play
  - Each pin is regarded as an arm, user engagement are considered as rewards
  - Making tradeoff between exploration and exploitation, avoid keep showing the best known pins and trap the system into local optima

# Use-case: Pinterest

- **Solution: Bandit algorithm in round  $t$** 
  - (1) Algorithm observes user a set  $A$  of pins/ads
  - (2) Based on payoffs from previous trials, algorithm chooses arm  $a \in A$  and receives payoff  $r_{t,a}$ 
    - Note only feedback for the chosen  $a$  is observed
  - (3) Algorithm improves arm selection strategy with each observation  $(a, r_{t,a})$
- **If the score for a pin is low, filter it out**

# Use-Case: A/B testing

- A/B testing is a controlled experiment with two variants, A and B
- Part of the traffic sees variant A, part variant B



# Use Case: A/B testing

- Part of the traffic sees variant A, part variant B
- Hypothesis test, does variant A outperform variant B? What test to perform?

Assumed Distribution	Example	Standard Test
<a href="#">Gaussian</a>	Average Revenue Per Paying User	<a href="#">Welch's t-test (Unpaired t-test)</a>
<a href="#">Binomial</a>	Click Through Rate	<a href="#">Fisher's exact test</a>
<a href="#">Poisson</a>	Transactions Per Paying User	E-test
<a href="#">Multinomial</a>	Number of each product purchased	<a href="#">Chi-squared test</a>

- If A outperforms B, we want to stop the experiment as soon as possible

# Use Case: A/B testing

- Imagine you have two versions of the website and you'd like to test which one is better
  - Version A has engagement rate of 5%
  - Version B has engagement rate of 4%
- You want to establish with 95% confidence that version A is better
  - You'd need 22,330 observations (11,165 in each arm) to establish that
    - Use t-test to establish the sample size
- Can bandits do better?

# Example: Bandits vs. A/B testing

- **How long does it take to discover  $A > B$ ?**
  - **A/B test:** We need 22,330 observations. Assuming 100 observations/day, we need 223 days
- **The goal is to find the best action (A vs. B)**
- The randomization distribution (traffic to A vs. B) can be updated as the experiment progresses
- **Idea:**
  - Twice per day, examine how each of the variations/arms has performed
  - Adjust the fraction of traffic that each arm will receive going forward 
  - An arm that appears to be doing well gets more traffic, and an arm that is clearly underperforming gets less

# Thompson Sampling

- **Thompson sampling** assigns sessions to arms in proportion to the probability that each arm is optimal
- **Let:**
  - $\theta = (\theta_1, \theta_2, \dots, \theta_k)$  ... the vector of conversion rates for arms  $1, \dots, k$ .
    - $\theta_i = \text{#successes} / (\text{#successes} + \text{#failures})$
  - $y$  ... the data observed thus far in the experiment
  - $I_a(\theta)$  ... the indicator of the event that arm  $a$  is optimal
- **Then we can write:**

$$P(Ia) = \int Ia(\theta) p(\theta|y) d\theta$$

# Thompson Sampling

- Arm probabilities  $\theta$  can be computed using sampling:
  - Each element of  $\theta$  is an independent random variable from a Beta distribution ( $\alpha + \text{successes}, \beta + \text{failures}$ )

---

**Algorithm 2** Thompson sampling for the Bernoulli bandit

---

**Require:**  $\alpha, \beta$  prior parameters of a Beta distribution

$S_i = 0, F_i = 0, \forall i$ . {Success and failure counters}

**for**  $t = 1, \dots, T$  **do**

**for**  $i = 1, \dots, K$  **do**

        Draw  $\theta_i$  according to  $\text{Beta}(S_i + \alpha, F_i + \beta)$ .

**end for**

    Draw arm  $\hat{i} = \arg \max_i \theta_i$  and observe reward  $r$

**if**  $r = 1$  **then**

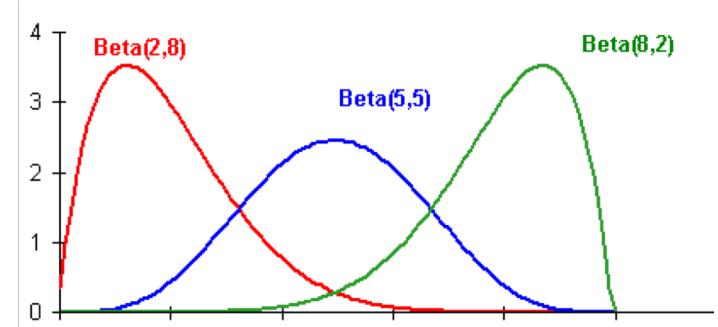
$S_{\hat{i}} = S_{\hat{i}} + 1$

**else**

$F_{\hat{i}} = F_{\hat{i}} + 1$

**end if**

**end for**



# Thompson Sampling

But, in our case we have to set the amount of traffic. Set it to be proportional to  $P(I_a)$ :

- (1) Simulate many draws from  $Beta(\alpha + S_a, \beta + F_a)$ :

Time	Arm 1	Arm 2	Arm 3
1	0.54	0.73	0.74
2	0.55	0.66	0.73
3	0.53	0.81	0.80
...			

- (2) The probability that arm  $a$  is optimal is the empirical fraction of rows for which arm  $a$  had the largest simulated value
- (3) Set traffic to arm  $a$  to be equal to % of wins

# Use Case: A/B testing

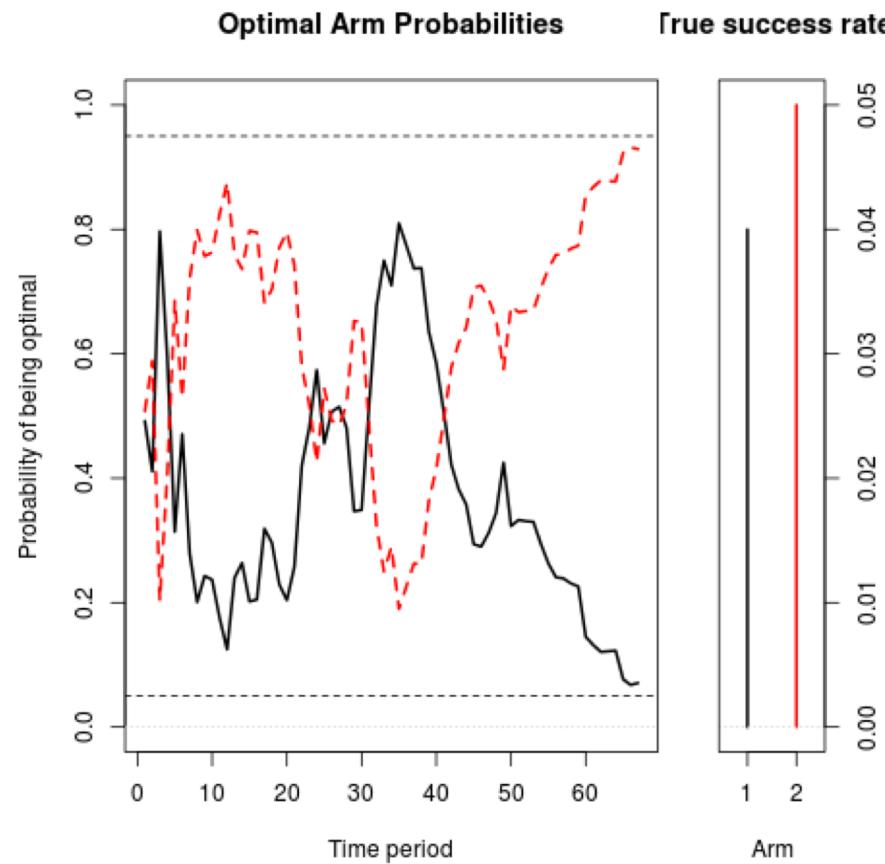
- **Imagine you have two versions of the website and you'd like to test which one is better**
  - Version A has engagement rate of 5%
  - Version B has engagement rate of 4%
- **You want to establish with 95% confidence that version A is better**
  - You'd need 22,330 observations (11,165 in each arm) to establish that
    - Use t-test to establish the sample size
- **Can bandits do better?**

# Example

**A/B test:** We need 22,330 observations. Assuming 100 observations/day, we need 223 days

- On 1<sup>st</sup> day about 50 sessions are assigned to each arm
- Suppose A got really lucky on the first day, and it appears to have a 70% chance of being superior
- Then we assign it 70% of the traffic on the second day, and the variant B gets 30%
- At the end of the 2nd day we accumulate all the traffic we've seen so far (over both days), and recompute the probability that each arm is best

# Simulation



- The experiment finished in 66 days, so it saved you 157 days of testing (66 vs 223)

# Generalization to multiple arms

- Easy to generalize to multiple arms:

