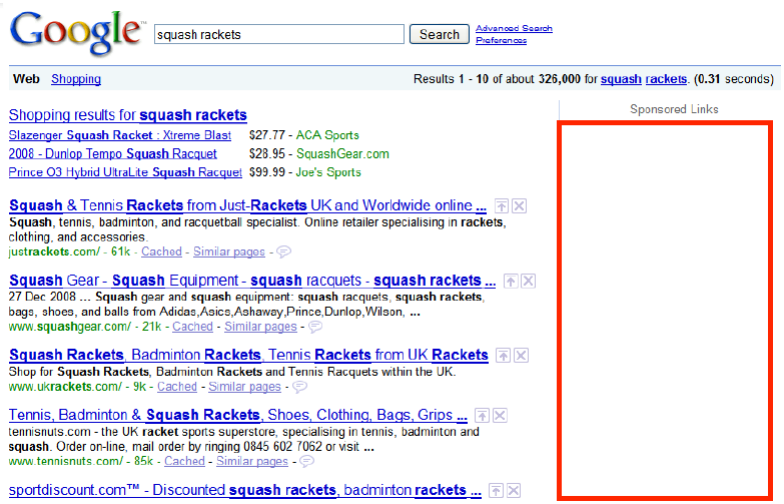
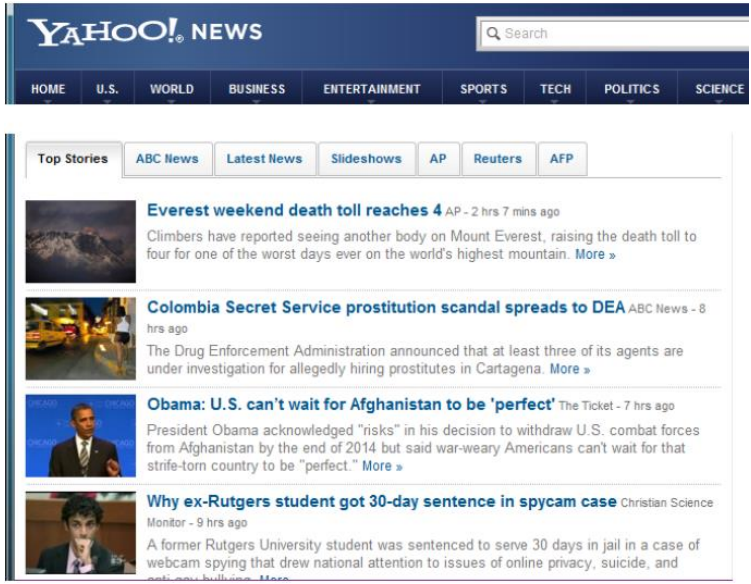
Multi-armed Bandits: Learning through Experimentation

CS246: Mining Massive Datasets
Caroline Lo, Stanford University
http://cs246.stanford.edu

Learning through Experimentation

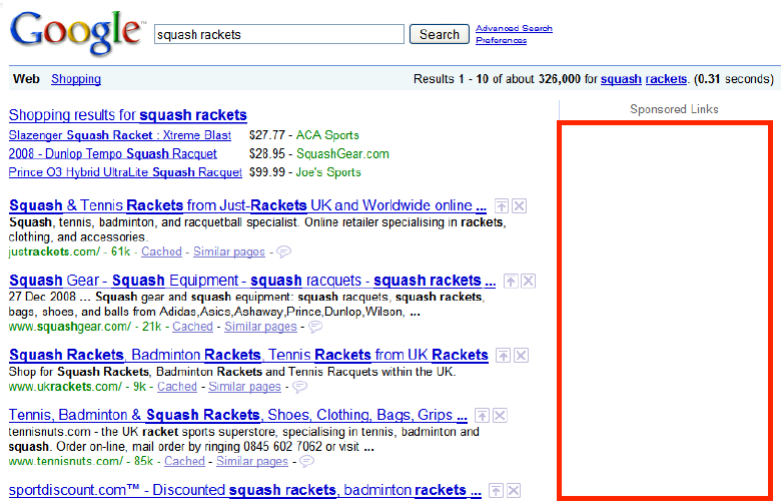
- **Web advertising**
  - We've learned how to match advertisers to queries in real-time
  - **But how to estimate the CTR (Click-Through Rate)?**
- **Recommendation engines**
  - We've learned how to build recommender systems
  - **But how to solve the cold-start problem?**

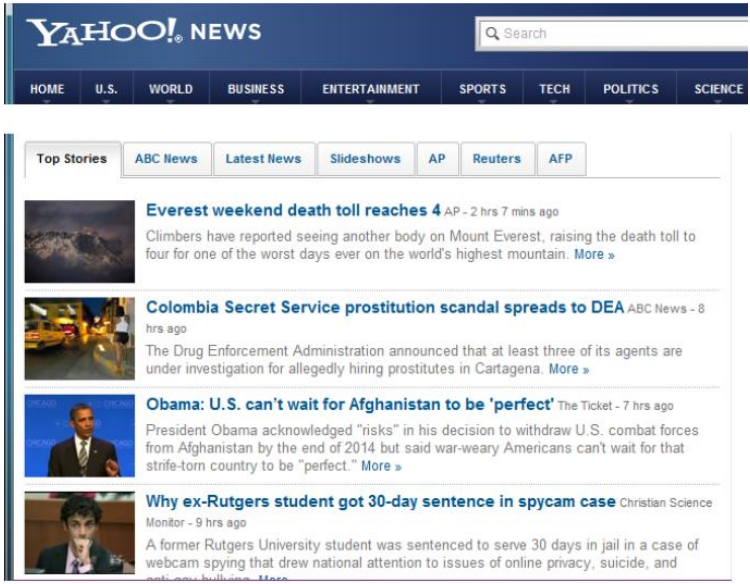




Learning through Experimentation

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





Example: Web Advertising

- **Google's goal: Maximize revenue**
- **The old way: Pay by impression (CPM)**

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 "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***?
  - **$E[revenue_{a,q}] = P(click_a \mid q) * amount_{a,q}$**

Prob. user will click on ad **a** given that she issues query **q**
(Unknown! Need to gather information)

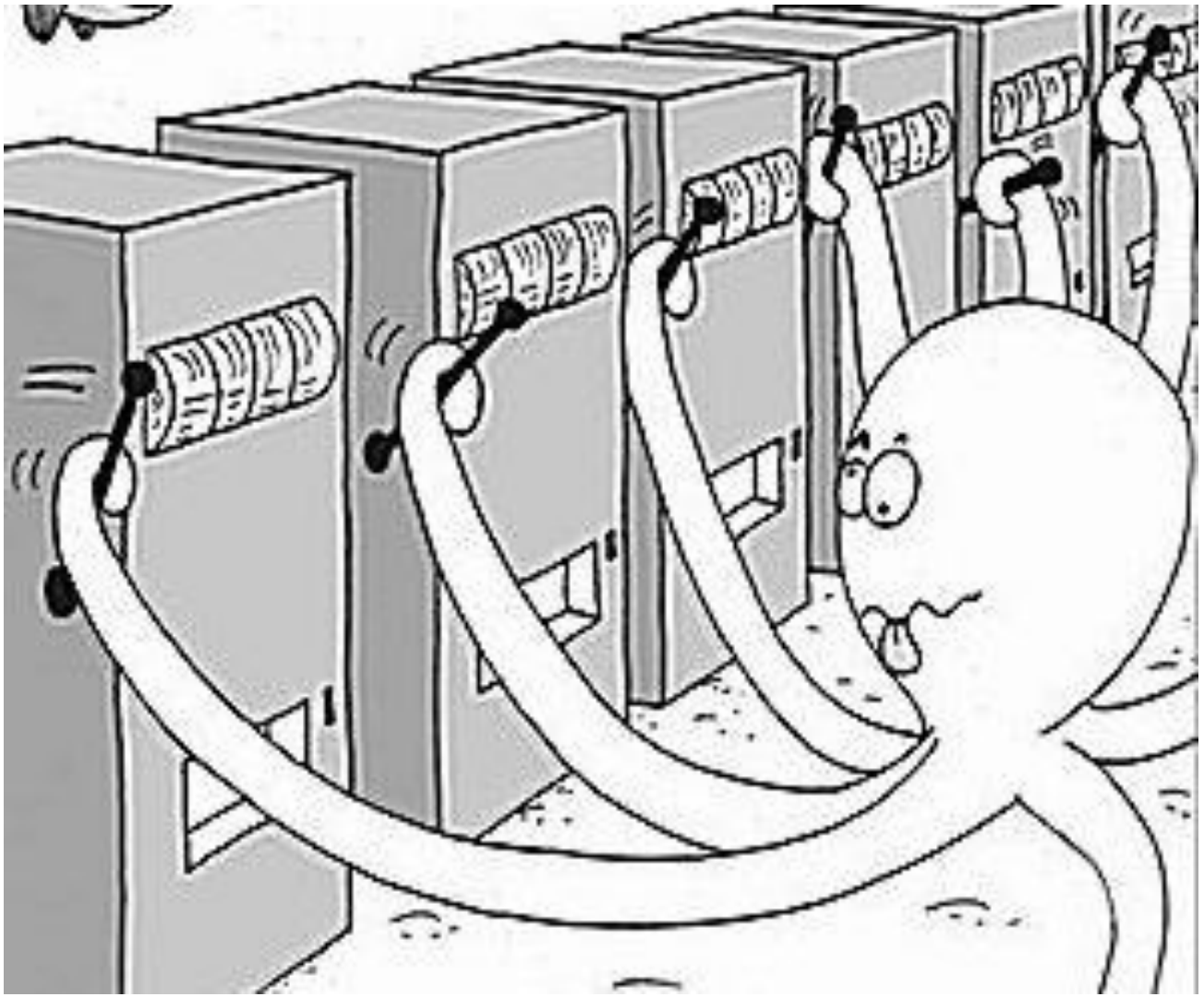
Bid amount for ad **a** on query **q**
(Known)

Other Applications

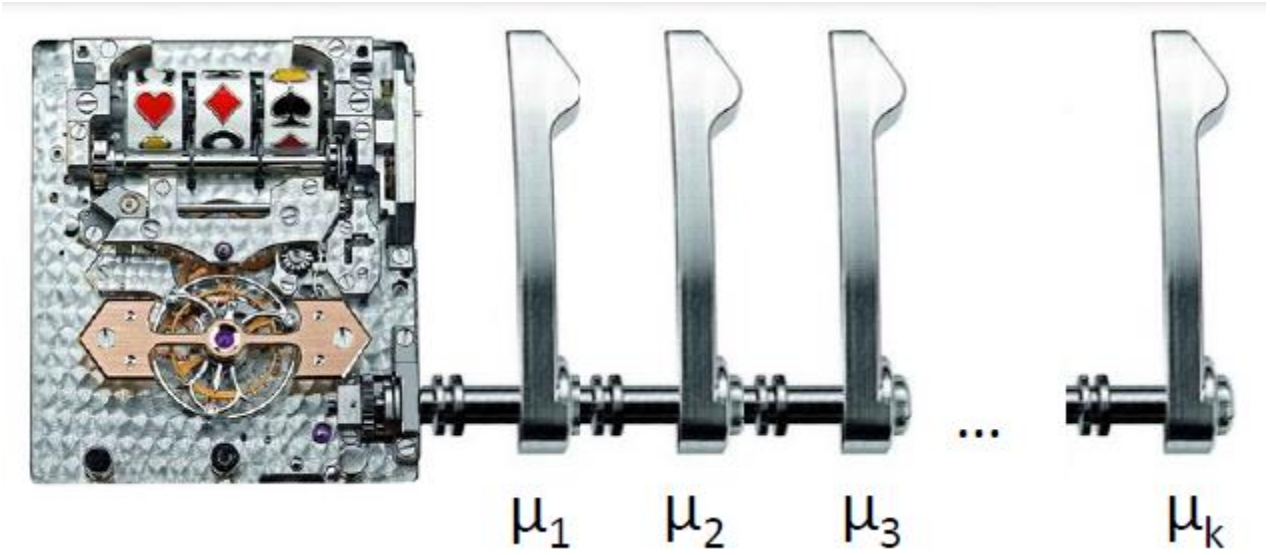
- **Clinical trials:**
  - Investigate effects of different treatments while minimizing patient losses
- **Adaptive routing:**
  - Minimize delay in the network by investigating different routes
- **Asset pricing:**
  - Figure out product prices while trying to make most money

Approach: Multiarmed Bandits

Approach: Multiarmed Bandits

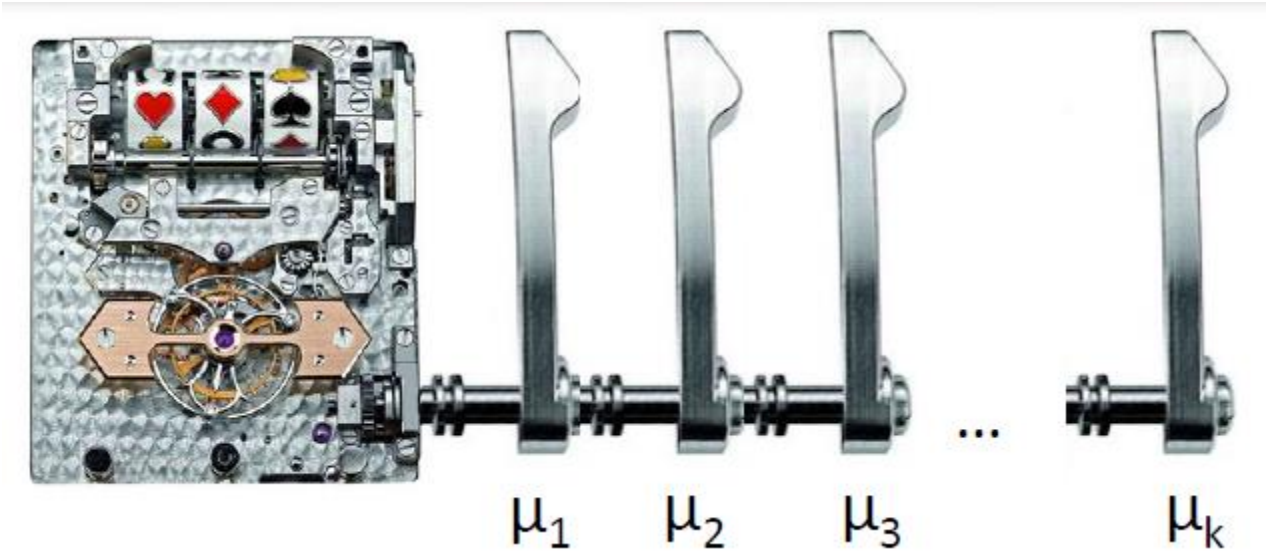


k-Armed Bandit



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

k-Armed Bandit



- **How does this map to our advertising example?**
- Each **query** is a **bandit**
- Each **ad** is an **arm**
- **We want to estimate the arm's probability of winning μ_a (i.e., the ad's CTR μ_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 tied to a probability distribution P_a with average reward/payoff μ_a (between [0, 1])
- We play the game for T rounds
- For each round t:
  - **(1)** We pick some arm j
  - **(2)** We win reward X_t drawn from P_j
    - Note reward is independent of previous draws
- **Our goal is to maximize** $\sum_{t=1}^{T} X_t$
- **We don't know μ_a!** But every time we pull some arm a we get to learn a bit about μ_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

- μ_a is the mean of P_a
- Payoff/reward of **best arm**: $\mu^* = \max_a \mu_a$
- Let $a_1, a_2 \ldots a_T$ 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:** $\frac{R_T}{T} \to 0$ **as** $T \to \infty$
  - Note: Ensuring $R_T/T \to 0$ is stronger than maximizing payoffs (minimizing regret), as it means that in the limit we discover the true best arm.

Allocation Strategies

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

$$\textbf{Pick} \quad \arg\max_{a} \mu_a$$

- We'd always pull the arm with the highest average reward.
- But we don't know which arm that is without **exploring**/experimenting with the arms first.

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

Exploration vs. Exploitation

- **Minimizing regret illustrates a classic problem in decision making:**
  - We need to trade off **exploration** (gathering data about arm payoffs) and **exploitation** (making decisions based on data already gathered)

  - **Exploration:** Pull an arm we never pulled before
  - **Exploitation:** Pull an arm a for which we currently have the highest estimate of μ_a

Algorithm: Epsilon-Greedy

Algorithm: Epsilon-Greedy

- **For t=1:T**
  - Set $\boldsymbol{\varepsilon_t = O(1/t)}$
  - **With prob. $\boldsymbol{\varepsilon_t}$: Explore** by picking an arm chosen uniformly at random
  - **With prob. $\boldsymbol{1-\varepsilon_t}$: Exploit** by picking an arm with highest empirical mean payoff
- **Theorem** [Auer et al. '02]

  For suitable choice of $\boldsymbol{\varepsilon_t}$ 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 with equal likelihood)
- **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!**

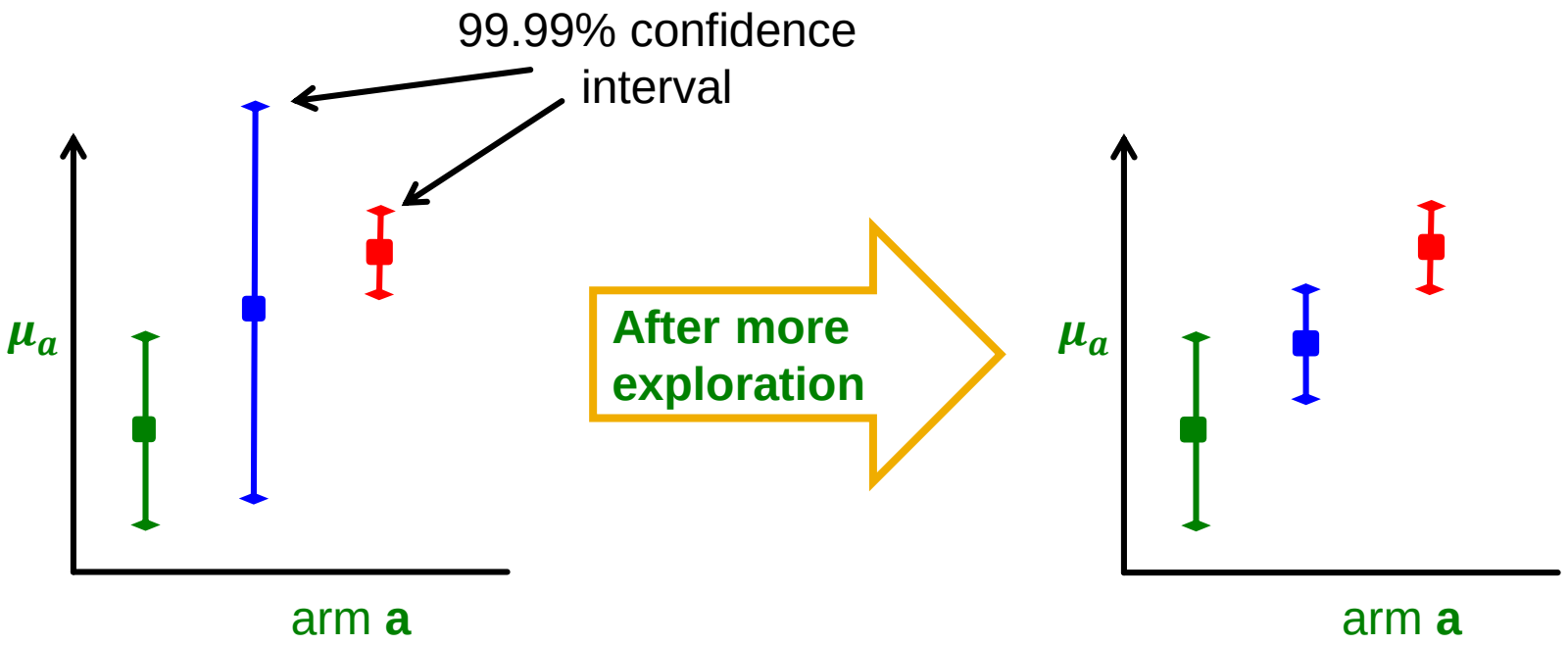
Confidence Intervals (1)

- **A confidence interval is a range of values within which we are sure the mean lies with a certain probability**
  - We could believe μ_a is within [0.2,0.5] with probability 0.95
  - If we 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 (i.e. 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 $Y_{a,1} \ldots Y_{a,m}$ be the payoffs of arm a in the first m trials
  - $Y_{a,1} \ldots Y_{a,m}$ are **i.i.d.** rnd. vars. with values in **[0,1]**
- **Expected mean payoff of arm *a*:** $\mu_a = E[Y_{a,m}]$
- **Our estimate:** $\widehat{\mu_{a,m}} = \frac{1}{m}\sum_{\ell=1}^{m} Y_{a,\ell}$
- Want to find confidence bound b such that with high probability $\left|\mu_a - \widehat{\mu_{a,m}}\right| \leq b$
  - Also want b to be as small as possible (**why?**)
- **Goal: Want to bound** $\mathbf{P}\left(\left|\mu_a - \widehat{\mu_{a,m}}\right| \leq b\right)$

Hoeffding's Inequality

- **Hoeffding's inequality bounds** $\mathbf{P}(|\mu_a - \widehat{\mu_{a,m}}| \leq b)$
  - Let $Y_1 \ldots Y_m$ be **i.i.d.** rnd. vars. with values between **[0,1]**
  - Let $\mu = E[Y]$ and $\widehat{\mu_m} = \frac{1}{m}\sum_{\ell=1}^{m} Y_\ell$
  - **Then:** $\mathbf{P}(|\mu - \widehat{\mu_m}| \geq b) \leq exp(-2b^2m) = \delta$
- **To find out the confidence interval b (for a given confidence level δ) we solve:**
  - $e^{-2b^2m} \leq \delta$ **then** $-2b^2m \leq \ln(\delta)$
  - **So:** $b \geq \sqrt{\frac{\ln(1/\delta)}{2m}}$

[Auer et al. '02]

UCB1 Algorithm

- **UCB1 (Upper confidence sampling) algorithm**
  - **Set:** $\widehat{\mu_1} = \cdots = \widehat{\mu_k} = 0$ and $m_1 = \cdots = m_k = 0$
    - $\widehat{\mu_a}$ **is our estimate of payoff of arm** i
    - m_a **is the number of pulls of arm** i **so far**
  - **For** $t = 1:T$
    - **For each arm** a **calculate:** $UCB(a) = \widehat{\mu_a} + \sqrt{\frac{2 \ln t}{m_a}}$
    - **Pick arm** $j = arg\ max_a\ UCB(a)$
    - **Pull arm** j **and observe** y_t
    - **Set:** $m_j \leftarrow m_j + 1$ **and** $\widehat{\mu_j} \leftarrow \frac{1}{m_j}\left(y_t + \left(m_j - 1\right)\widehat{\mu_j}\right)$

Upper confidence interval (Hoeffding's inequality)

UCB1: Discussion

- $UCB(a) = \widehat{\mu_a} + \sqrt{\frac{2 \ln t}{m_a}}$

$$b \geq \sqrt{\frac{\ln(1/\delta)}{2m}}$$

  - t impacts the value of δ: $t = f(1/\delta)$
  - 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

"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 $\boldsymbol{\mu^* = \max_a \mu_a}$
  - And for each arm let $\boldsymbol{\Delta_a = \mu^* - \mu_a}$
  - **Then it holds that**

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

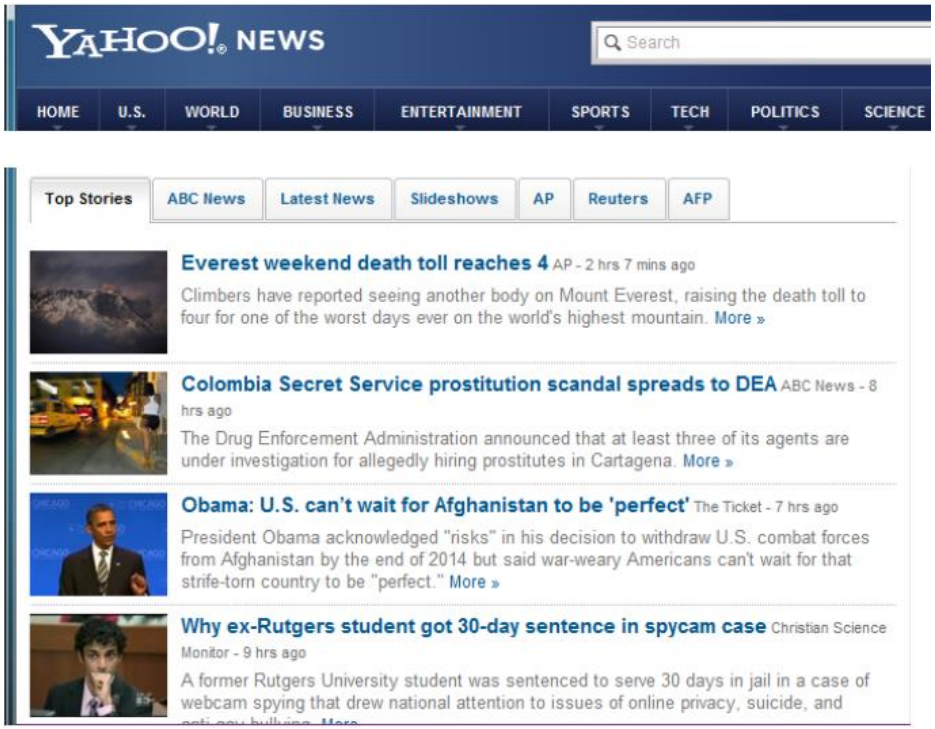
$O(k \ln T)$ $O(k)$

  - **So:** $O\left(\frac{R_T}{T}\right) \leq k \frac{\ln T}{T}$

Summary

- ***k*-armed bandit problem is a formalization of the exploration-exploitation tradeoff**
- **Simple algorithms are able to achieve no regret (limit towards infinity)**
  - Epsilon-greedy
  - UCB (Upper Confidence Sampling)

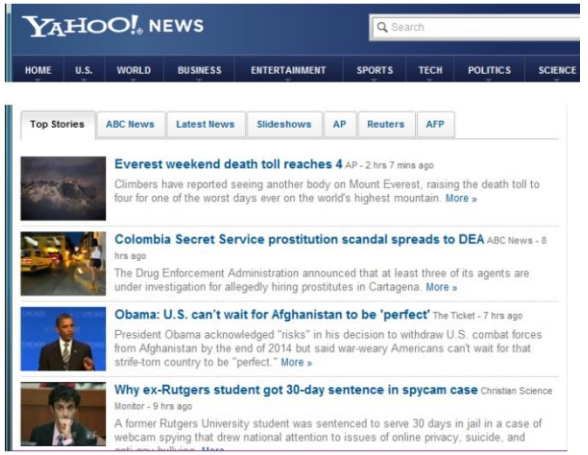
News Recommendation



- Every round receive **context** [Li et al., WWW '10]
  - **Context:** User features, articles view before
- **Model for each article's click-through rate**

News Recommendation

- **Feature-based exploration:**
  - **Select articles to serve users based on contextual information about the user and the articles**
  - **Simultaneously adapt article selection strategy based on user-click feedback to maximize total number of user clicks**



Example: A/B testing vs. Bandits

- 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 student's t-test to establish the sample size
  - Can bandits do better?

Example: Bandits vs. A/B testing

- **How long it does it take to discover A > B?**
  - **A/B test:** We need 22,330 observations. Assuming 100 observations/day, we need 223 days
  - **Bandits:** We use UCB1 and keep track of confidences for each version we stop as soon as A is better than B with 95% confidence.
    How much do we save?
    - 175 days on the average!
    - **48 days vs. 223 days**
    - **More at: http://bit.ly/1pywka4**