

# Artificial Intelligence

CS 165A

Nov 17, 2020

Instructor: Prof. Yu-Xiang Wang

T  
o  
d  
a  
y

→ Bandits and exploration

# A few logistic notes

- Finishing grading
  - Individual midterm results posted on Gradescope
- HW3 due Tuesday next week
  - Discussion class this Wednesday will help you with MDPs
- HW3 solutions won't be discussed on Wednesday discussion class.
  - So please try submitting on time if possible
- HW4 will be released on Thursday

# Recap: Tabular MDP

- **Discrete** State, **Discrete** Action, Reward and Observation

$$S_t \in \mathcal{S} \quad A_t \in \mathcal{A} \quad R_t \in \mathbb{R} \quad \text{--- } O_t \in \mathcal{O}$$

- Policy:

– When the state is observable:  $\pi : \mathcal{S} \rightarrow \mathcal{A}$

~~– Or when the state is not observable~~

$$\text{--- } \pi_t : (\mathcal{O} \times \mathcal{A} \times \mathbb{R})^{t-1} \rightarrow \mathcal{A}$$

- Learn the best policy that maximizes the expected reward

– Finite horizon (episodic) RL:  $\pi^* = \arg \max_{\pi \in \Pi} \mathbb{E} \left[ \sum_{t=1}^T R_t \right]$  **T: horizon**

– Infinite horizon RL:  $\pi^* = \arg \max_{\pi \in \Pi} \mathbb{E} \left[ \sum_{t=1}^{\infty} \gamma^{t-1} R_t \right]$

**$\gamma$ : discount factor**

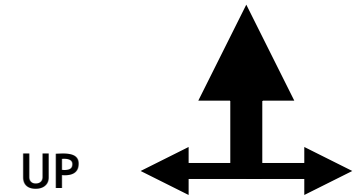
# Question from Piazza: Is this an infinite horizon or finite-horizon MDP?

			+1
			-1
START			

- reward +1 at [4,3], -1 at [4,2]
- reward -0.04 for each step

actions: UP, DOWN, LEFT, RIGHT

e.g.,



State-transitions with action **UP**:

- 80% move up
- 10% move left
- 10% move right

\*If you bump into a wall,  
you stay where you are.

# Recap: Reward function and Value functions

- Immediate reward function  $r(s,a,s')$

- **expected immediate** reward

$$r(s, a, s') = \mathbb{E}[R_1 | S_1 = s, A_1 = a, S_2 = s']$$

$$r^\pi(s) = \mathbb{E}_{a \sim \pi(a|s)}[R_1 | S_1 = s]$$

- state value function:  $V^\pi(s)$

- **expected long-term** return when starting in  $s$  and following  $\pi$

$$V^\pi(s) = \mathbb{E}_\pi[R_1 + \gamma R_2 + \dots + \gamma^{t-1} R_t + \dots | S_1 = s]$$

- state-action value function:  $Q^\pi(s,a)$

- **expected long-term** return when starting in  $s$ , performing  $a$ , and following  $\pi$

$$Q^\pi(s, a) = \mathbb{E}_\pi[R_1 + \gamma R_2 + \dots + \gamma^{t-1} R_t + \dots | S_1 = s, A_1 = a]$$

# Recap: Bellman equations – the fundamental equations of MDP and RL

- An alternative, recursive and more useful way of defining the V-function and Q function

# Recap: Bellman equations – the fundamental equations of MDP and RL

- An alternative, recursive and more useful way of defining the V-function and Q function
  - $V^\pi$  function Bellman equation

$$V^\pi(s) = \sum_a \pi(a|s) \sum_{s'} P(s'|s, a) [r(\underline{s, a, s'}) + \gamma \underline{V^\pi(s')}]$$

# Recap: Bellman equations – the fundamental equations of MDP and RL

- An alternative, recursive and more useful way of defining the V-function and Q function

- $V^\pi$  function Bellman equation

$$V^\pi(s) = \sum_a \pi(a|s) \sum_{s'} P(s'|s, a) [r(s, a, s') + \gamma V^\pi(s')]$$

- $Q^\pi$  function Bellman equation

$$Q^\pi(s, a) = \sum_{s'} P(s'|s, a) [r(s, a, s') + \gamma \sum_{a'} \pi(a'|s') Q^\pi(s', a')]$$



# Recap: Bellman equations – the fundamental equations of MDP and RL

- An alternative, recursive and more useful way of defining the V-function and Q function

- $V^\pi$  function Bellman equation

$$V^\pi(s) = \sum_a \pi(a|s) \sum_{s'} P(s'|s, a) [r(s, a, s') + \gamma V^\pi(s')]$$

- $Q^\pi$  function Bellman equation

$$Q^\pi(s, a) = \sum_{s'} P(s'|s, a) [r(s, a, s') + \gamma \sum_{a'} \pi(a'|s') Q^\pi(s', a')]$$

- $V^*$  function Bellman (optimality) equation

$$V^*(s) = \max_a \sum_{s'} P(s'|s, a) [r(s, a, s') + \gamma V^*(s')]$$

# Recap: Bellman equations – the fundamental equations of MDP and RL

- An alternative, recursive and more useful way of defining the V-function and Q function

- V<sup>π</sup> function Bellman equation

$$V^\pi(s) = \sum_a \pi(a|s) \sum_{s'} P(s'|s, a) [r(s, a, s') + \gamma V^\pi(s')]$$

- Q<sup>π</sup> function Bellman equation

$$Q^\pi(s, a) = \sum_{s'} P(s'|s, a) [r(s, a, s') + \gamma \sum_{a'} \pi(a'|s') Q^\pi(s', a')]$$

- V\* function Bellman (optimality) equation

$$V^*(s) = \max_a \sum_{s'} P(s'|s, a) [r(s, a, s') + \gamma V^*(s')]$$

- Q\* function Bellman (optimality) equation

$$Q^*(s, a) = \sum_{s'} P(s'|s, a) [r(s, a, s') + \gamma \max_{a'} Q^*(s', a')]$$

# Recap: Policy Iterations and Value Iterations

- What are these algorithms for?
  - Algorithms of computing the  $V^*$  and  $Q^*$  functions from MDP parameters

Policy Eval: calculate  $V^\pi$

- Policy Iterations

$$\pi_0 \rightarrow \overset{\uparrow}{\text{E}} V^{\pi_0} \rightarrow \overset{\downarrow}{\text{I}} \pi_1 \rightarrow \overset{\uparrow}{\text{E}} V^{\pi_1} \rightarrow \text{I} \dots \rightarrow \text{I} \pi^* \rightarrow \overset{\uparrow}{\text{E}} V^*$$

- Value iterations

for all  $s$

$$\underline{V_{k+1}(s)} \leftarrow \max_a \sum_{s'} P(s'|s, a) [r(s, a, s') + \gamma V_k(s')] \quad V_0 \in \mathcal{R}^{|S|}$$

- How do we make sense of them?
  - Recursively applying the Bellman equations until convergence.

# Recap: Matrix-form of Bellman Equations and Policy Evaluation

$$V^\pi(s) = \sum_a \pi(a|s) \sum_{s'} P(s'|s, a) [r(s, a, s') + \gamma V^\pi(s')]$$



$$V^\pi = r^\pi + \gamma P_\pi^T V^\pi$$

$\boxed{V^\pi} = \boxed{r^\pi} + \gamma \boxed{P_\pi^T} \boxed{V^\pi}$

$$V_{k+1}^\pi(s) \leftarrow \sum_a \pi(a|s) \sum_{s'} P(s'|s, a) [r(s, a, s') + \gamma V_k^\pi(s')]$$



$$V_{k+1}^\pi \leftarrow r^\pi + \gamma P_\pi^T V_k^\pi$$

$\underbrace{\hspace{15em}}$

A reasonable question for the final: Matrix-form Bellman equation with  $Q^\pi$

# Matrix-form of Bellman Optimality Equation

$$V^*(s) = \max_a \sum_{s'} P(s'|s, a) [r(s, a, s') + \gamma V^*(s')]$$



?



$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} P(s'|s, a) [r(s, a, s') + \gamma V_k(s')]$$



?

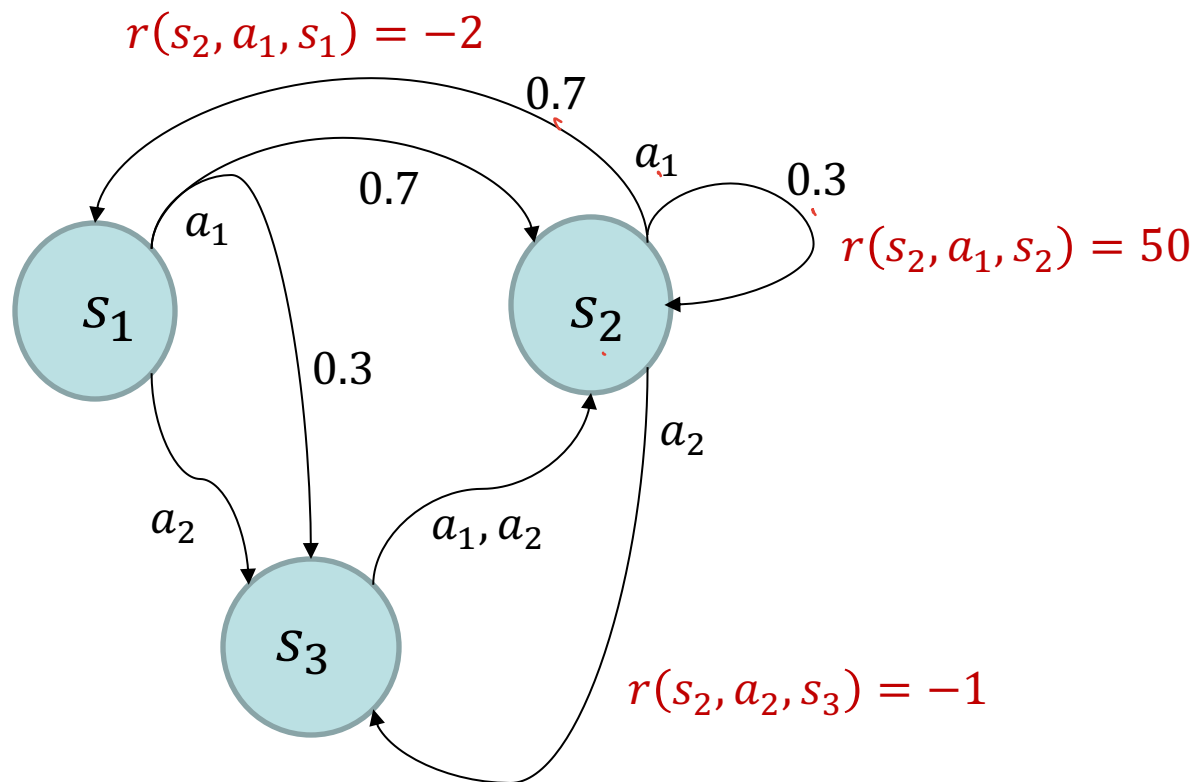
# Today's topic

- Exploration in the (multi-armed) Bandits problem
- Bandits algorithms
  - Explore-first
  - epsilon-greedy
  - Upper confidence bound

# Solving MDP with VI or PI is offline planning

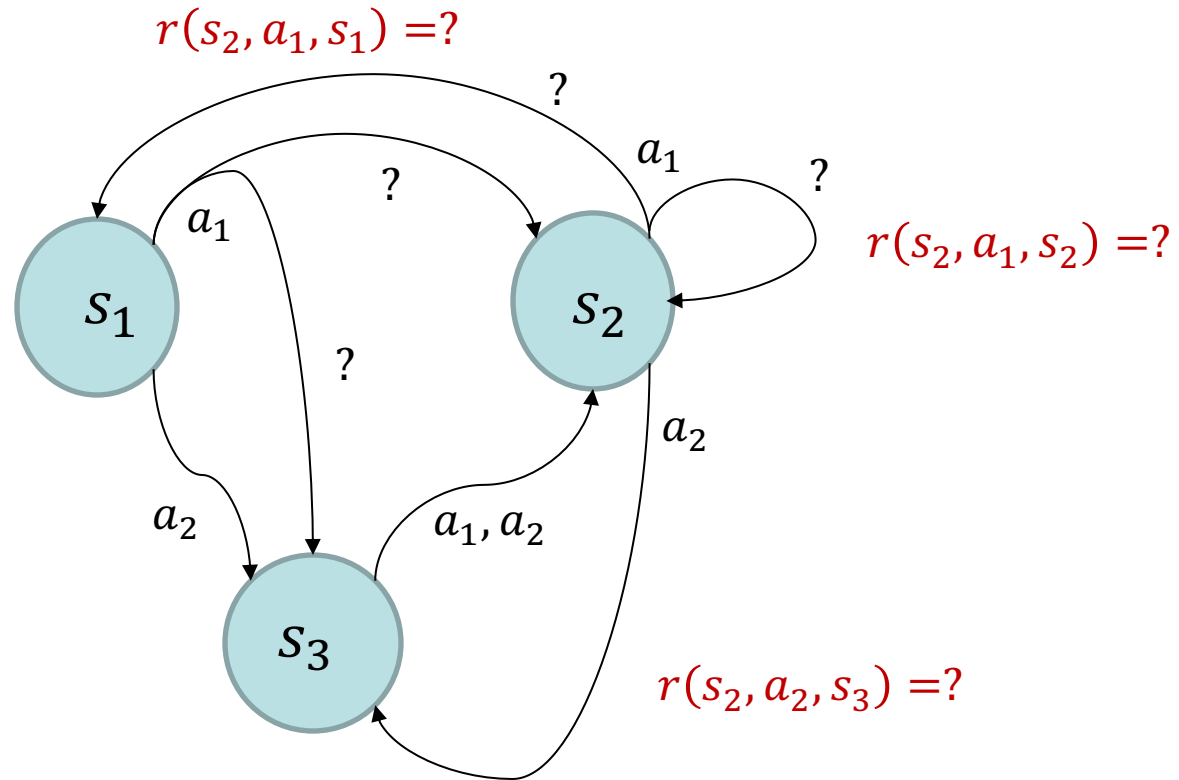
- The agent is given how the environment works
- The agent works out the optimal policy in its mind.
- The agent never really starts to play at all.
- No learning is happening.

# State-space diagram representation of an MDP: An example with 3 states and 2 actions.

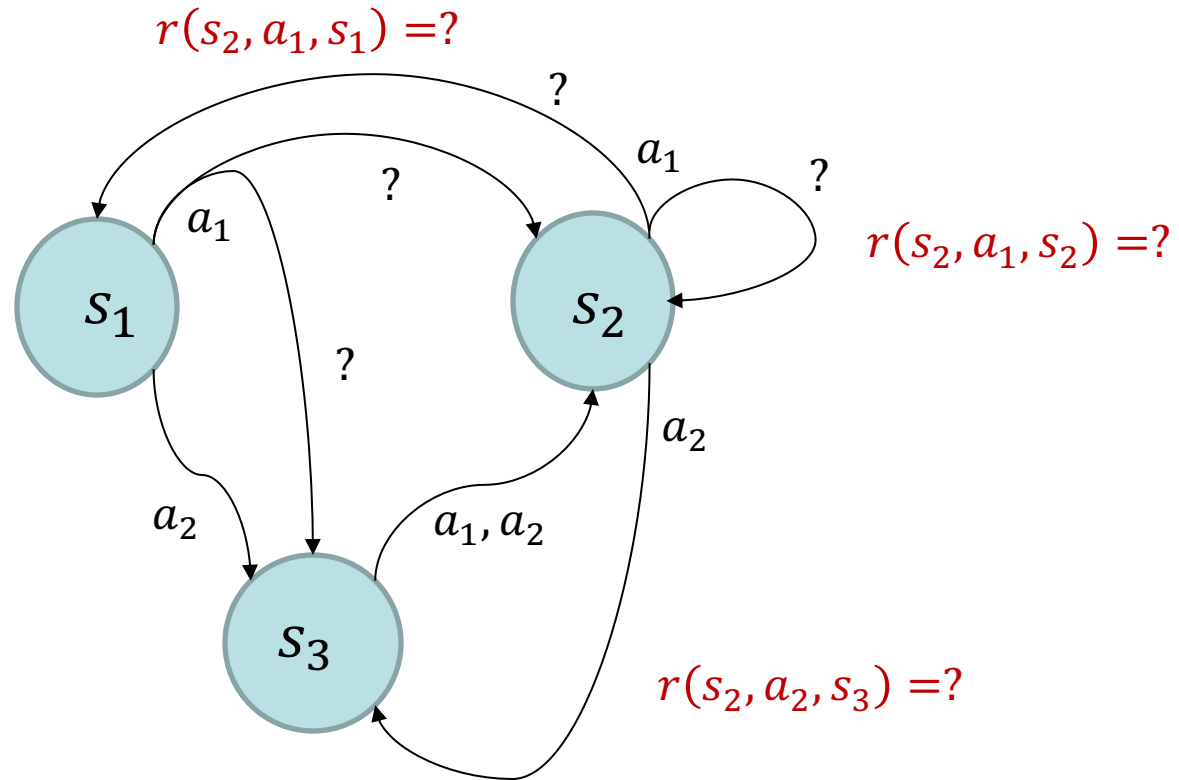




# What happens if you do not know the rewards / transition probabilities?



# What happens if you do not know the rewards / transition probabilities?



Then you have to learn by interacting with the unknown environment.

You cannot use only offline planning!

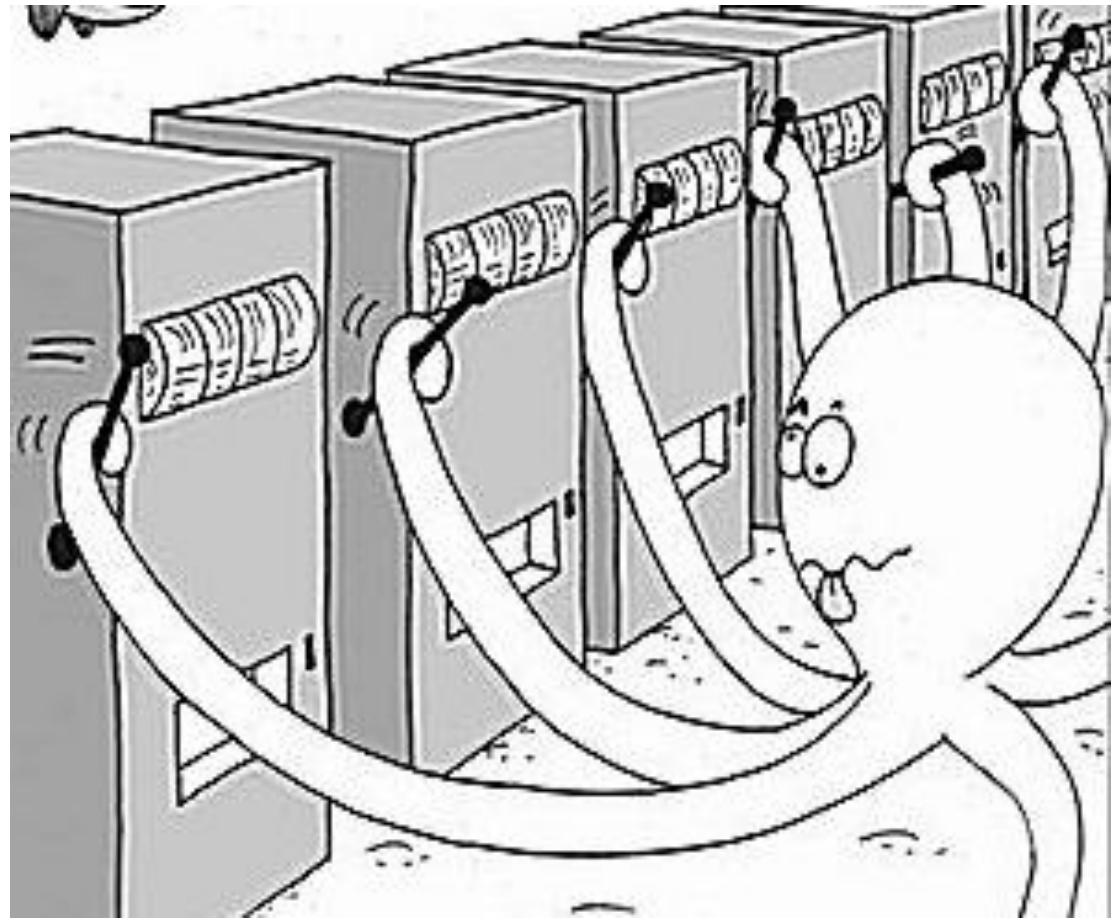
**Exploration:** Try unknown actions to see what happens.

**Exploitation:** Maximize utility using what we know.

# Let us tackle different aspects of the RL problem one at a time

- Markov Decision Processes:
  - Dynamics are given no need to learn
- **Bandits: Explore-Exploit in simple settings**
  - RL without dynamics
- Full Reinforcement Learning
  - Learning MDPs

# Slot machines and Multi-arm bandits



# Multi-arm bandits: Problem setup

$S = \emptyset$

- No state. k-actions  $a \in \mathcal{A} = \{1, 2, \dots, k\}$

- You decide which arm to pull in every iteration

$$A_1, A_2, \dots, A_T$$

- You collect a cumulative payoff of  $\sum_{t=1}^T R_t$

- The goal of the agent is to maximize the expected payoff.
  - For future payoffs?
  - For the expected cumulative payoff?

# Key differences from MDPs

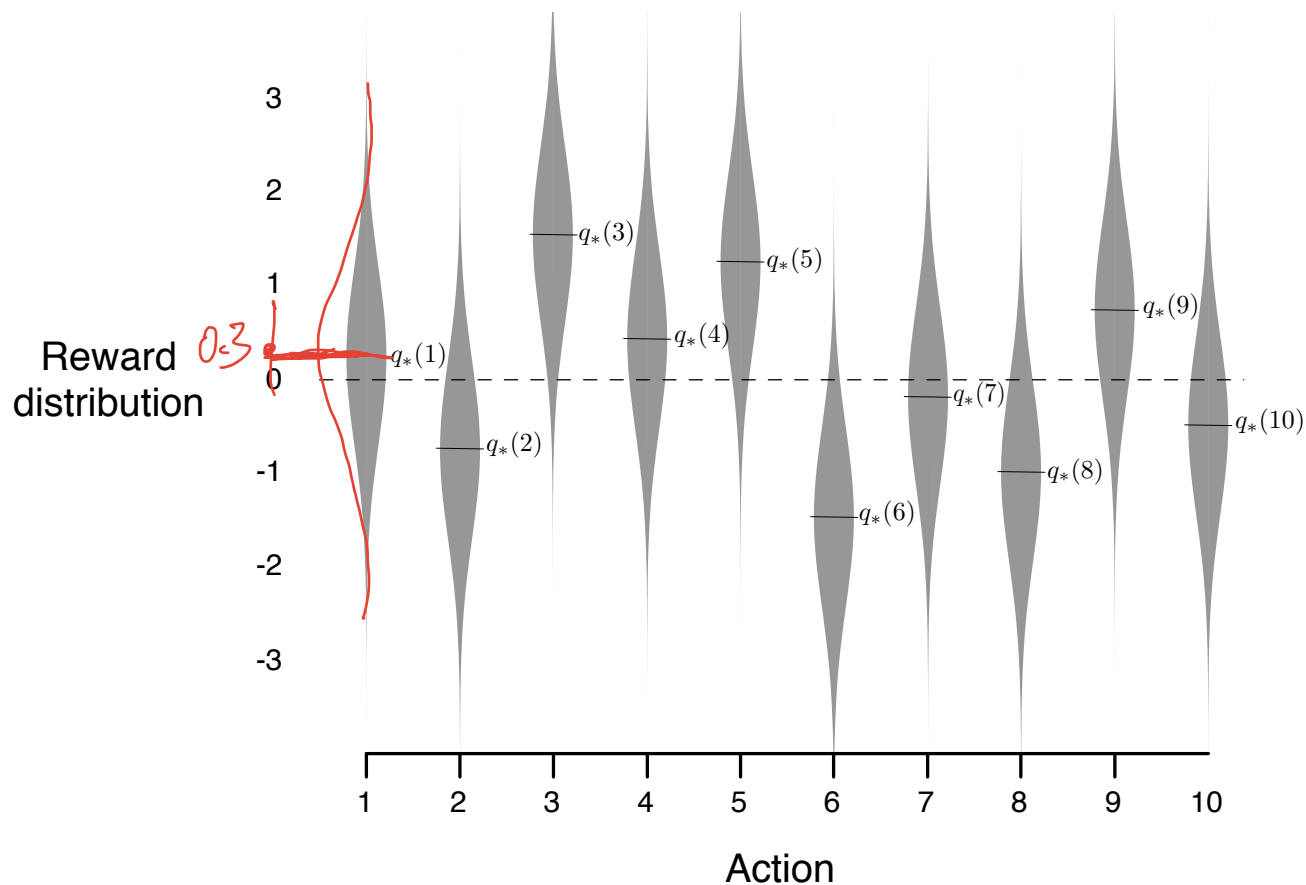
- Simplified:
  - No state-transitions

- But:

- We are not given the expected reward  $r(s, a, s')$
- We need to learn the optimal policy by trials-and-errors.

$$r(a) = \mathbb{E}[R_t | A_t = a]$$

# A 10-armed bandits example



**Figure 2.1:** An example bandit problem from the 10-armed testbed. The true value  $q_*(a)$  of each of the ten actions was selected according to a normal distribution with mean zero and unit variance, and then the actual rewards were selected according to a mean  $q_*(a)$  unit variance normal distribution, as suggested by these gray distributions.

# How do we measure the performance of an online learning agent?

- The notion of “Regret”:
  - I wish I have done things differently.
  - Comparing to the best actions in the hindsight, how much worse did I do.

- For MAB, the regret is defined as follow

$$\text{Regret}(\bar{\pi}) = T \max_{a \in [k]} \mathbb{E}[R_t | a] - \sum_{t=1}^T \mathbb{E}_{a \sim \pi} [\mathbb{E}[R_t | a]]$$

$[k] = \{1, 2, 3, \dots, k\}$

if  $O(T^{-\alpha})$   
 $0 < \alpha < 1$   
 then  
 $\frac{1}{T} \text{Regret} \xrightarrow{T \rightarrow \infty} 0$

$$\text{Regret}(A_1, \dots, A_T) = T \max_{a \in [k]} \mathbb{E}[R_t | a] - \sum_{t=1}^T \mathbb{E}[R_t | A_t = a_t]$$



# Greedy strategy

- Expected reward

$$q_*(a) \doteq \mathbb{E}[R_t \mid A_t = a].$$

- Estimate the expected reward

$Q_t(a)$   $\doteq$   $\frac{\text{sum of rewards when } a \text{ taken prior to } t}{\text{number of times } a \text{ taken prior to } t}$

$$= \frac{\sum_{i=1}^{t-1} R_i \cdot \mathbb{1}_{A_i=a}}{\sum_{i=1}^{t-1} \mathbb{1}_{A_i=a}}$$

$Q_*(1) = 0.6$   $R|_{A=1} \sim \text{Ber}(0.5)$

$Q_*(2) = 0.7$   $R|_{A=2} \sim \text{Ber}(0.5)$

$Q_t(a) = 0.5 \forall a \in \{1, 2\}$

- Choose  $A_t \doteq \arg \max_a Q_t(a)$ ,

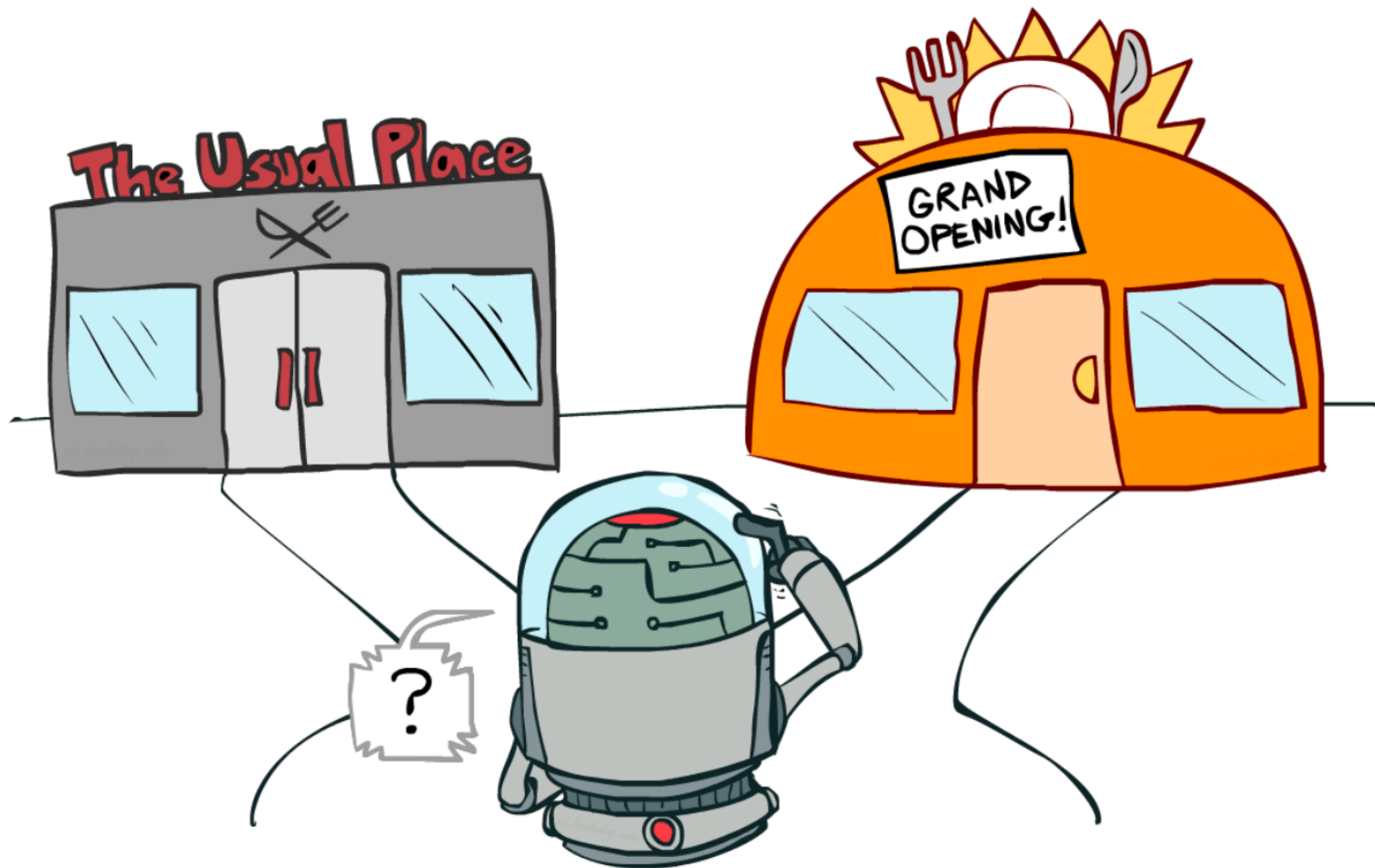
$R_1 = 1$ .  $Q_2(1) = \frac{R_1}{1} = 1$ .

$Q_2(2) = 0.5$

$Q_2 \rightarrow 1 > 0.5$

What is the issue with this strategy?

# Exploration vs. Exploitation



(Illustration from Dan Klein and Pieter Abbeel's course in UC Berkeley)

# Exploration first strategy

- Let's spend the first  $N$  step exploring.
  - Play each action for  $N/k$  times.

$$Q_t(a) = \frac{\sum_{i=1}^{t-1} R_i \cdot \mathbb{1}_{A_i=a}}{\sum_{i=1}^{t-1} \mathbb{1}_{A_i=a}} = N/k$$

as  $N/k \rightarrow \infty$   
 $Q_t(a) \rightarrow Q_*(a)$

- For  $t = N+1, N+2, \dots, T$ :

$$A_t \doteq \operatorname{argmax}_a \underline{Q_t(a)},$$

All (you need to know) about Statistics in one slide, two theorems.

# All (you need to know) about Statistics in one slide, two theorems.

- Statistics is about using **samples** from a distribution to infer the properties of the distribution itself (**population**)
  - $X_1, X_2, X_3, \dots, X_n \stackrel{iid}{\sim} P$

# All (you need to know) about Statistics in one slide, two theorems.

- Statistics is about using **samples** from a distribution to infer the properties of the distribution itself (**population**)

- $X_1, X_2, X_3, \dots, X_n \sim P$

- Law of large number

- Average  $\rightarrow$  Mean

$$\bar{X} := \frac{1}{n} \sum_{i=1}^n X_i \xrightarrow{n \rightarrow \infty} \mathbb{E}[X_1]$$

# All (you need to know) about Statistics in one slide, two theorems.

- Statistics is about using **samples** from a distribution to infer the properties of the distribution itself (**population**)

- $X_1, X_2, X_3, \dots, X_n \sim P$

- Law of large number

- Average  $\rightarrow$  Mean

$$\bar{X} := \frac{1}{n} \sum_{i=1}^n X_i \rightarrow \mathbb{E}[X_1]$$

- Central limit theorem

- The rate is  $\sqrt{1/n}$

$$\sqrt{n} \cdot \left( \frac{1}{n} \sum_{i=1}^n X_i - \mathbb{E}[X_1] \right) \rightarrow \underline{N(0, \text{Var}(X_1))}$$

$$\left| \bar{X} - \mathbb{E}[X_1] \right| = O\left(\frac{1}{\sqrt{n}}\right) \in$$

Random variables are difficult to work with, but with statistics, a lot of the properties are more-or-less deterministic, thanks to LLN.



Random variables are difficult to work with, but with statistics, a lot of the properties are more-or-less deterministic, thanks to LLN.

- Statistics is about using **samples** from a distribution to infer the properties of the distribution itself (**population**)
  - $X_1, X_2, X_3, \dots, X_n \sim P$

Random variables are difficult to work with, but with statistics, a lot of the properties are more-or-less deterministic, thanks to LLN.

- Statistics is about using **samples** from a distribution to infer the properties of the distribution itself (**population**)

- $X_1, X_2, X_3, \dots, X_n \sim P$

- **(Simplified version of the) Hoeffding's Inequality:**

- with high probability,

$$\left| \frac{1}{n} \sum_{i=1}^n X_i - \mathbb{E}[X_1] \right| = O(1/\sqrt{n})$$

Random variables are difficult to work with, but with statistics, a lot of the properties are more-or-less deterministic, thanks to LLN.

- Statistics is about using **samples** from a distribution to infer the properties of the distribution itself (**population**)
  - $X_1, X_2, X_3, \dots, X_n \sim P$
- **(Simplified version of the) Hoeffding's Inequality:**
  - with high probability,

$$\left| \frac{1}{n} \sum_{i=1}^n X_i - \mathbb{E}[X_1] \right| = O(1/\sqrt{n})$$

\*If you try each arm multiple times, then you have a good idea what the expected reward is.

# How does Exploration-First work?

- Assume:  $0 < \text{reward} < 1$
- After  $N$  rounds, *with high probability*
  
- Do this for all  $k$  arms.
  
- (w.h.p.) The regret is smaller than

$$N + C(T - N)\sqrt{k/N}$$

# How does Exploration-First work?

- Assume:  $0 < \text{reward} < 1$
- After  $N$  rounds, *with high probability*

$$Q_t(a) \approx q_*(a) \pm C \sqrt{\frac{k}{N}}$$

- Do this for all  $k$  arms.
- *(w.h.p.)* The regret is smaller than

$$N + C(T - N) \sqrt{k/N}$$

# How does Exploration-First work?

- Assume:  $0 < \text{reward} < 1$
- After  $N$  rounds, *with high probability*

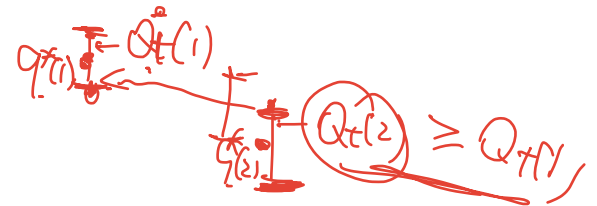
$$Q_t(a) \approx q_*(a) \pm C \sqrt{\frac{k}{N}}$$

$n = \frac{N}{k}$   
 $\sqrt{\frac{k}{N}} = \sqrt{\frac{1}{n}}$

**Hoeffding's inequality!**

- Do this for all  $k$  arms.
- (w.h.p.) The regret is smaller than

$$N + C(T - N) \sqrt{k/N}$$



# How does Exploration-First work?

- (High probability) Regret bound:

$$N + C(T - N)\sqrt{k/N}$$

# How does Exploration-First work?

- (High probability) Regret bound:

$$N + C(T - N)\sqrt{k/N} \leq N + \frac{CT\sqrt{k}}{\sqrt{N}}$$

- Choose N to minimize the regret

$$N = O(T^{2/3} k^{1/3})$$

$$N = \frac{CT\sqrt{k}}{\sqrt{N}}$$
$$N^{1.5} = CT\sqrt{k}$$
$$N = O\left(T^{\frac{2}{3}} k^{\frac{1}{3}}\right)$$



# How does Exploration-First work?

- (High probability) Regret bound:

$$N + C(T - N)\sqrt{k/N}$$

- Choose  $N$  to minimize the regret

$$N = O(T^{2/3} k^{1/3})$$

- Final regret bound:

$$O(T^{2/3} k^{1/3})$$

# $\epsilon$ -Greedy strategy: one way to balance exploration and exploitation

- You choose with probability  $1 - \epsilon$

$$A_t \doteq \underset{a}{\operatorname{argmax}} \underbrace{Q_t(a)},$$

- With probability  $\epsilon$ , choose an action **uniformly at random!**
  - Including the argmax.
- Carefully choose  $\epsilon$  parameter.

# Let's analyze $\epsilon$ -Greedy!

- By Hoeffding's inequality, at every  $t$ 
  - w.h.p. each arm is chosen at least  $0.5 \epsilon t / k$  times.
  - w.h.p.,

# Let's analyze $\epsilon$ -Greedy!

- By Hoeffding's inequality, at every  $t$

– w.h.p. each arm is chosen at least  $0.5 \epsilon t / k$  times.

– w.h.p.,

$$Q_t(a) \approx q_*(a) \pm C \sqrt{k/\epsilon t}$$

$$n = \frac{\epsilon t}{k}$$

$$\sqrt{\frac{1}{n}} = \sqrt{\frac{k}{\epsilon t}}$$

# Let's analyze $\epsilon$ -Greedy!

- By Hoeffding's inequality, at every  $t$

$$\mathbb{E}[\text{\# of times } a \text{ is taken}] = \frac{\epsilon t}{k}$$

– w.h.p. each arm is chosen at least  $0.5 \epsilon t / k$  times.

– w.h.p.,

$$\begin{aligned} \text{\# of times } a \text{ is taken} \\ \geq 0.5 \frac{\epsilon t}{k} \\ \text{w.h.p.} \end{aligned}$$

$$Q_t(a) \approx q_*(a) \pm C \sqrt{k/\epsilon t}$$

- Regret bound is

$$\underbrace{\epsilon T}_{\uparrow} + \underbrace{\sum_{t=1}^T C \sqrt{\frac{k}{\epsilon t}}}_{\uparrow} \quad \begin{matrix} \leq 1 \\ (1-\epsilon) \end{matrix}$$

$$1 + \frac{1}{\sqrt{2}} + \frac{1}{\sqrt{3}} + \dots + \frac{1}{\sqrt{T}}$$

# Convergent and divergent series

$$1 + 1/2^2 + 1/3^2 + \dots + 1/T^2 =? \leq 2$$

---

Harmonic Series

$$1 + 1/2 + 1/3 + \dots + 1/T =? O(\log T)$$

---

On the side, check out Riemann's Zeta Function...

# Convergent and divergent series

$$1 + 1/2^2 + 1/3^2 + \dots + 1/T^2 = ?$$

$$1 + 1/2 + 1/3 + \dots + 1/T = ?$$

$$1 + \frac{1}{\sqrt{2}} + \frac{1}{\sqrt{3}} + \dots + \frac{1}{\sqrt{T}} = ?$$

$\leq O(\sqrt{T})$

On the side, check out Riemann's Zeta Function...

# Let's analyze $\epsilon$ -Greedy!

- Regret bound

$$\epsilon T + \sum_{t=1}^T C \sqrt{\frac{k}{\epsilon t}} \leq \epsilon T + O\left(\sqrt{\frac{Tk}{\epsilon}}\right)$$

- Choose the optimal  $\epsilon$  to minimize the bound?
- Work it out yourself that you get  $T^{\frac{2}{3}}$  just like in Explore-First.

$$\frac{1}{T} T^{\frac{2}{3}} = T^{-\frac{1}{3}}$$



# Upper Confidence Bound algorithm (UCB)

# Upper Confidence Bound algorithm (UCB)

- At time  $t$ , choose the action

$$A_t \leftarrow \operatorname{argmax}_a \left[ Q_t(a) + c \sqrt{\frac{\log(1+t)}{N_t(a)}} \right]$$

exploration bonus

$$Q_t(a) = \bar{Q}_t(a) \pm \frac{c}{\sqrt{N_t(a)}}$$

# Upper Confidence Bound algorithm (UCB)

- At time  $t$ , choose the action

$$A_t \leftarrow \operatorname{argmax}_a \left[ \underbrace{Q_t(a)} + c \sqrt{\frac{\log(1+t)}{N_t(a)}} \right]$$

- Idea: Be optimistic
  - Choose an option that maximizes the upper confidence bound.

# Upper Confidence Bound algorithm (UCB)

- At time  $t$ , choose the action

$$A_t \leftarrow \operatorname{argmax}_a \left[ Q_t(a) + c \sqrt{\frac{\log(1+t)}{N_t(a)}} \right]$$

- Idea: Be optimistic
  - Choose an option that maximizes the upper confidence bound.

$$\mathbb{E}[\text{Regret}] = O(\sqrt{Tk})$$

- The proof is out of the scope of this course. For those who are interested, please look up. It's not difficult.

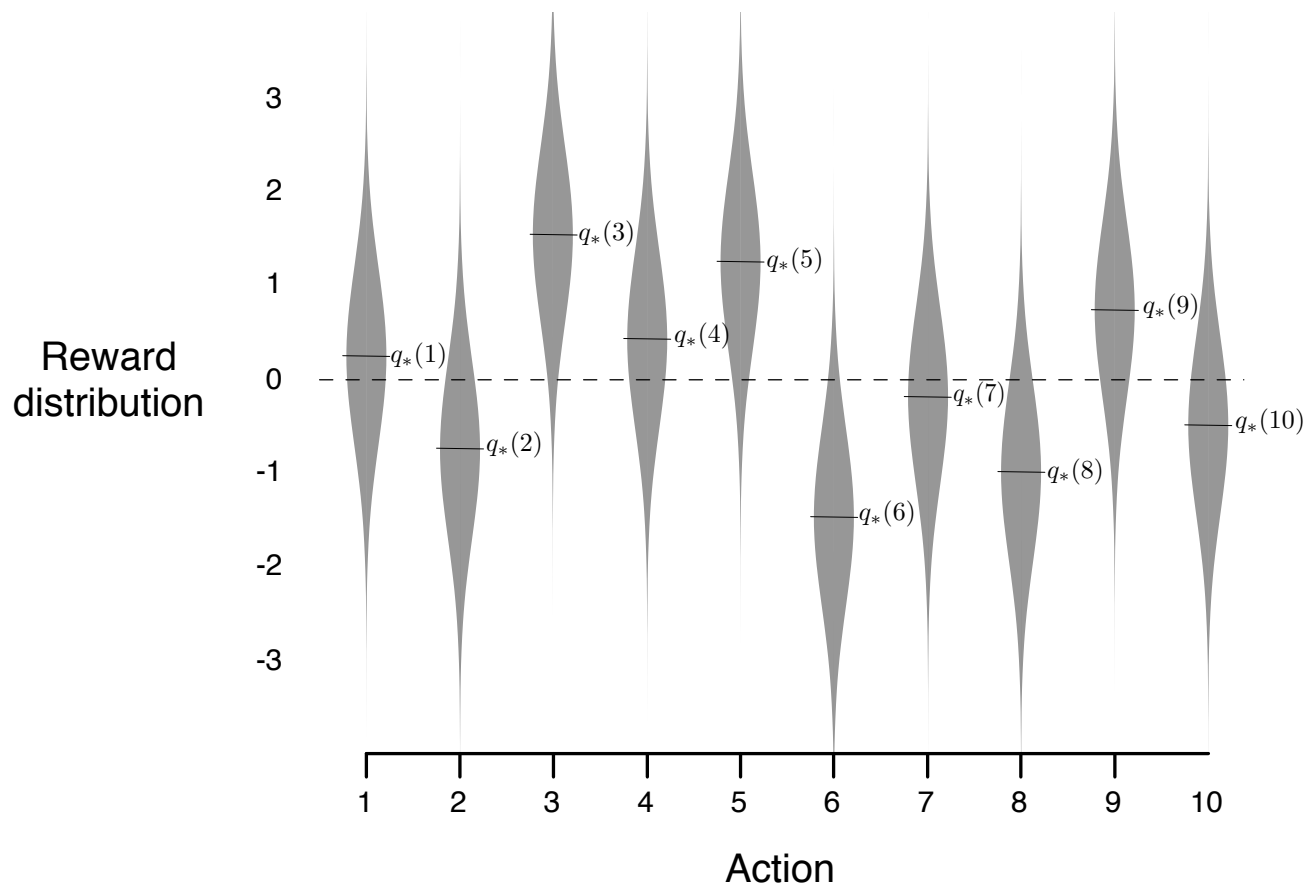
# Example: Two-Armed Bandits

- $k=2$ . Expected reward  $r(a = 1) = 0.8$ ,  $r(a = 2) = 0.5$
- Question: what is the Q-function for this problem?
  - If episodic with horizon  $T=1$ .  $Q_{(s,a)}^* = r(a)$
  - If discounted with  $0 \leq \gamma < 1$   $\gamma = 0$
- What is the optimal policy?

# Example: UCB on a Two Armed Bandit

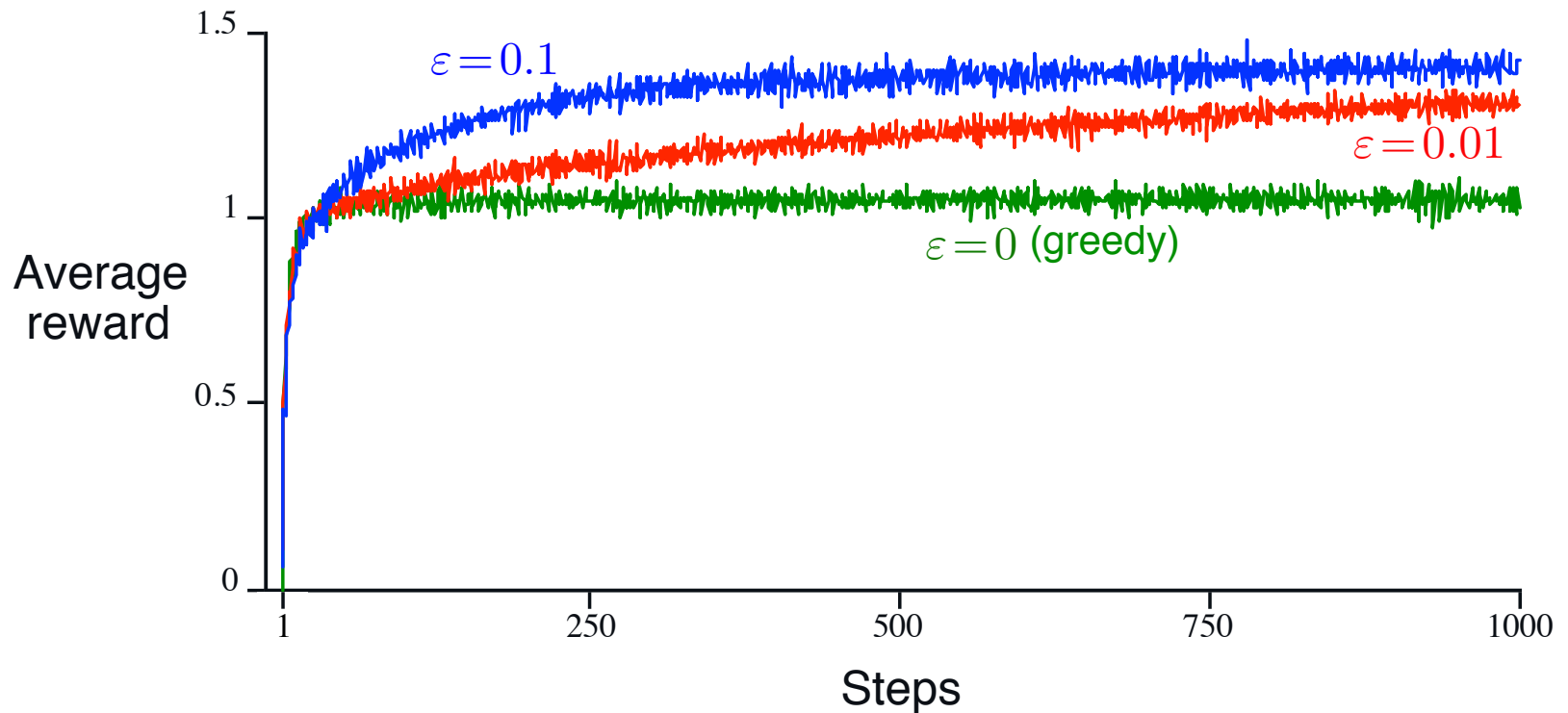
- A run of UCB algorithm with  $c = 2$ .
  1. Initialize  $Q_1(a) = \infty, \forall a \in \{1,2\}$ . UCB:  $\bar{Q}_1(a) = \infty$ .
  2. Pick action  $A_1 = 1$ . (break ties arbitrarily), receive a reward of 0.
    - a. Update  $N_2(a) \leftarrow N_1(a) + 1(A_1 = a)$
    - b. Update  $Q_2(1) = 0/1 = 0, Q_2(2) = \infty$
    - c. Update  $\bar{Q}_2(1) \leftarrow Q_2(1) + 2\sqrt{\frac{\log 2}{1}} \approx 0 + 1.67, \bar{Q}_2(2) = \infty$
  3. Pick action  $a = 2$  (because of what?), receive a reward of 1.
    - a. Update  $N_3(2) \leftarrow N_2(2) + 1, Q_3(2) \leftarrow 1$
    - b. Update  $\bar{Q}_3(2) \leftarrow Q_3(2) + 2\sqrt{\frac{\log 2}{1}} \approx 1 + 1.67$
    - c. Keep  $N_3(1), Q_3(1), \bar{Q}_3(1)$  unchanged.
  4. Which action to pick at next?
  5. ...

# A 10-armed bandits benchmark



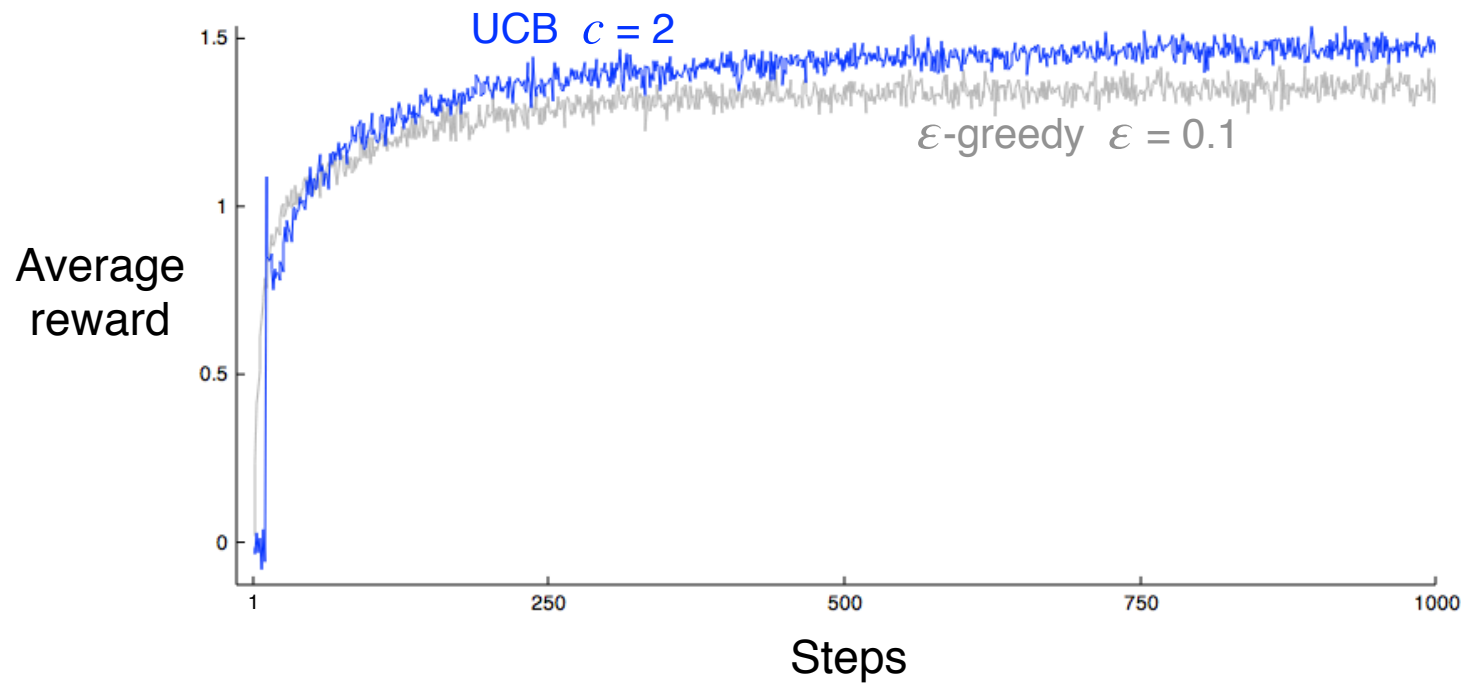
**Figure 2.1:** An example bandit problem from the 10-armed testbed. The true value  $q_*(a)$  of each of the ten actions was selected according to a normal distribution with mean zero and unit variance, and then the actual rewards were selected according to a mean  $q_*(a)$  unit variance normal distribution, as suggested by these gray distributions.

# Comparing the different algorithms



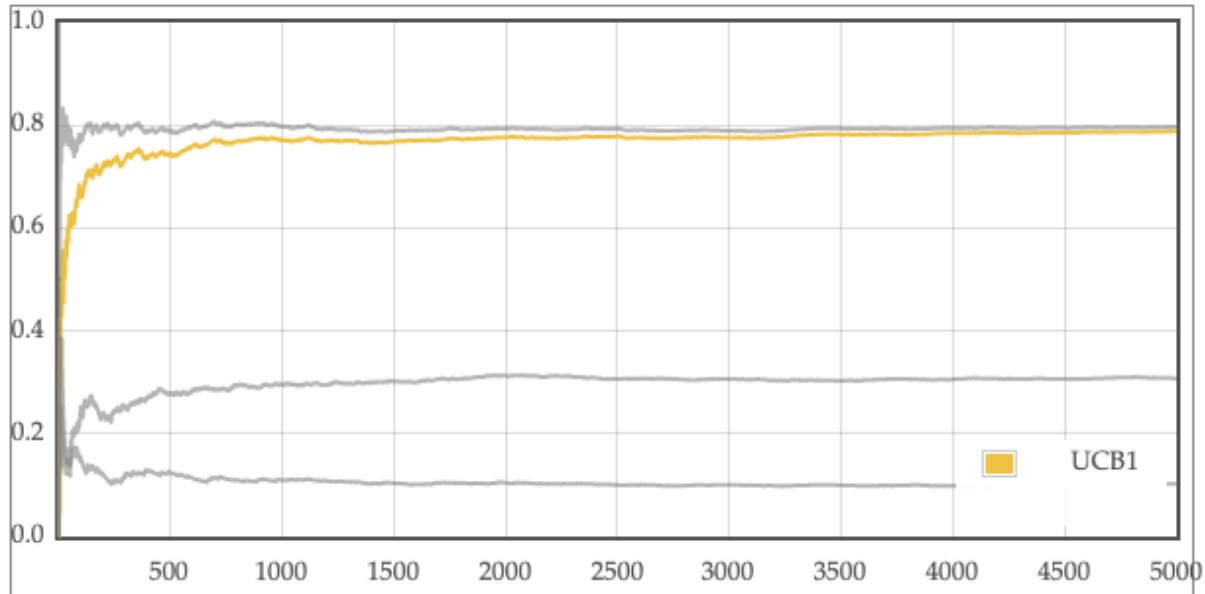


# UCB vs. $\epsilon$ -Greedy



# Live Demo of an Agent Solving Multi-Armed Bandits

Bandits		Agents			
1.	Bernoulli ▾ 0.1	1.	UCB1 ▾	Run game for	5000
2.	Bernoulli ▾ 0.3	+ create		steps.	
3.	Bernoulli ▾ 0.8			<input type="button" value="Go!"/>	
+ create				Average:	<input checked="" type="checkbox"/>



Credit Mark Reid: <http://mark.reid.name/code/bandits/>

# Variants of Bandits problems

- Online Learning from Expert Advice
  - Adversarial chooses the outcome
  - You observe outcome of other arms as well
  - Compare against the best arm in the hindsight
- Adversarial  $k$ -Armed Bandits
  - Same as above. But you observe only your arm.
- Nonstationary Bandits
  - Stochastic but the reward distribution changes over time.
  - Compare against the best arm for each time.
- Contextual bandits: you have a state in each time point.

# Variants of Bandits problems

- Online Learning from Expert Advice
  - Adversarial chooses the outcome
  - You observe outcome of other arms as well
  - Compare against the best arm in the hindsight
- Adversarial k-Armed Bandits
  - Same as above. But you observe only your arm.
- Nonstationary Bandits
  - Stochastic but the reward distribution changes over time.
  - Compare against the best arm for each time.
- Contextual bandits: you have a state in each time point.

**Remark:** In all these problems, there are algorithms with provably low-regret.

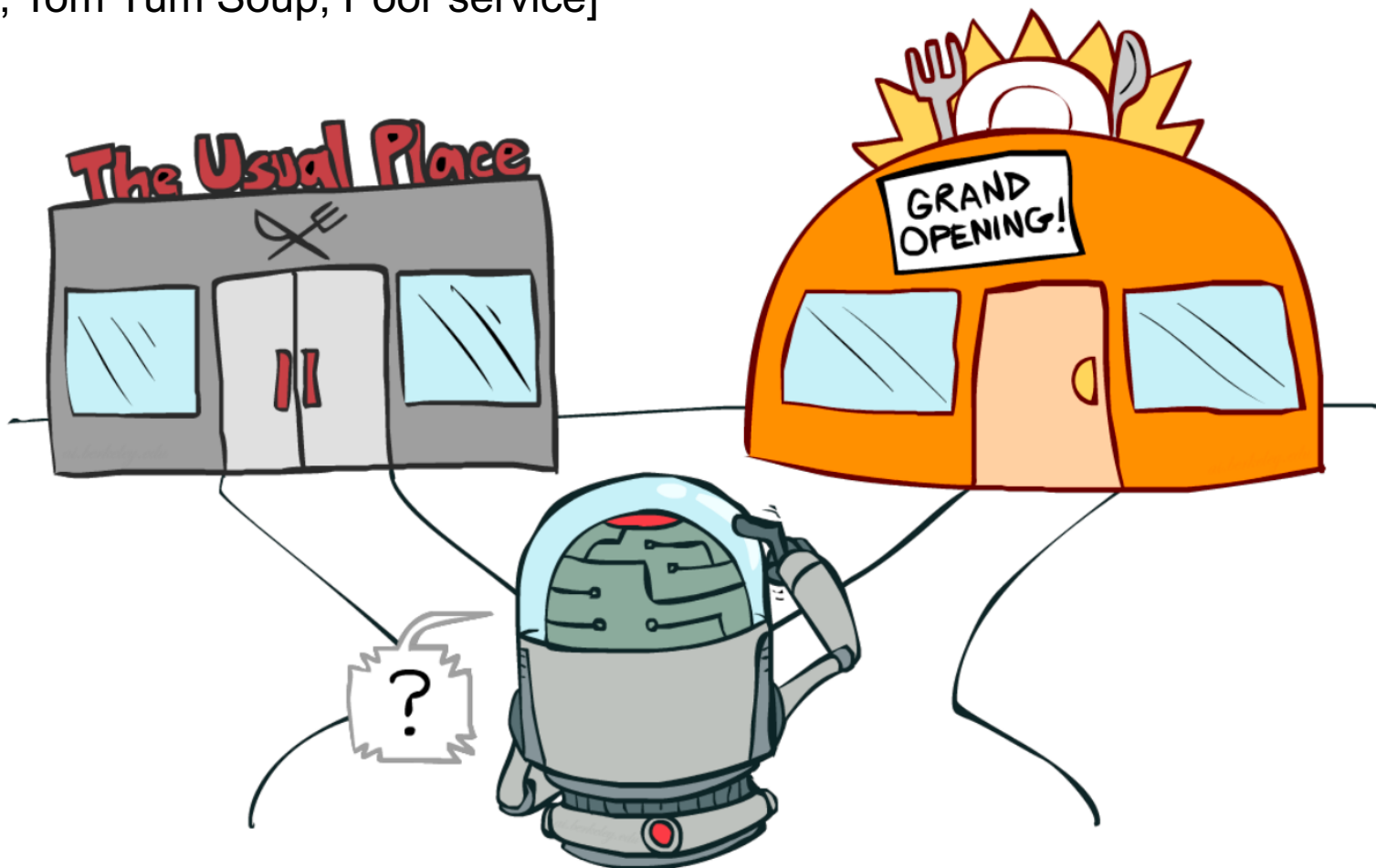
# Do I have to try, if I have features?

Features:

[Noodles, Tom Yum Soup, Poor service]

Features:

[Burger, Fries, Onion Ring, Fried Chicken]



(Illustration from Dan Klein and Pieter Abbeel's course in UC Berkeley)

# We know how to use with features, don't we?

- Classifier agent
  - Take features of a restaurant as input
  - Output a prediction of “will I like the food?”
- Train with supervised learning
  - Using the my previous visits to the restaurants
  - Using Yelp reviews

# We know how to use with features, don't we?

- Classifier agent
  - Take features of a restaurant as input
  - Output a prediction of “will I like the food?”
- Train with supervised learning
  - Using the my previous visits to the restaurants
  - Using Yelp reviews

Why can't we just use that?

# We know how to use with features, don't we?

- Classifier agent
  - Take features of a restaurant as input
  - Output a prediction of “will I like the food?”
- Train with supervised learning
  - Using the my previous visits to the restaurants
  - Using Yelp reviews

Why can't we just use that?

How to explore?



# Contextual Bandits: Problem Setup

- For each round  $t = 1, 2, 3, \dots, T$ :
  - A context  $x_t \sim$  unknown distribution i.i.d.
  - Agent picks an action  $a_t = 1, 2, 3, \dots, K$
  - Reward  $r_t \sim D(\cdot | x_t, a_t)$

- Agent's goals: A finite family of policies
  - Learn the best policy out of many policies  $\Pi$
  - Minimize the cumulative regret

$$T \cdot \max_{\pi \in \Pi} \mathbb{E}_{\pi} [r_t(x_t, a_t)] - \mathbb{E}_{\text{Agent's policy}} \left[ \sum_{t=1}^T r_t(x_t, a_t) \right]$$

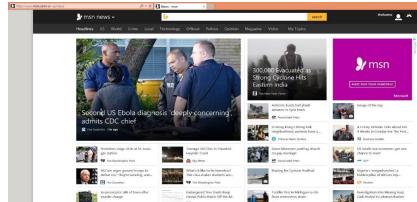
↑  
Reward from the best policy

↑  
Reward collected by the Agent

# Applications of Contextual Bandits

# Applications of Contextual Bandits

Personalized news?



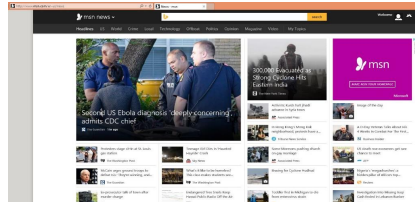
Repeatedly:

1. Observe features of user+articles
2. Choose a news article.
3. Observe click-or-not

**Goal: Maximize fraction of clicks**

# Applications of Contextual Bandits

Personalized news?

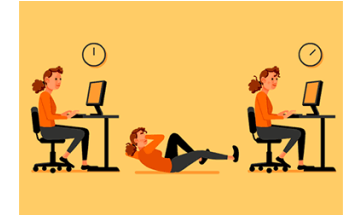


Repeatedly:

1. Observe features of user+articles
2. Choose a news article.
3. Observe click-or-not

**Goal: Maximize fraction of clicks**

Health advice?



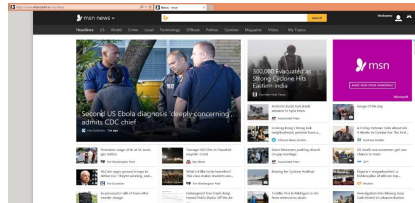
Repeatedly:

1. Observe features of user+advice
2. Choose an advice.
3. Observe steps walked

**Goal: Healthy behaviors (e.g. step count)**

# Applications of Contextual Bandits

Personalized news?

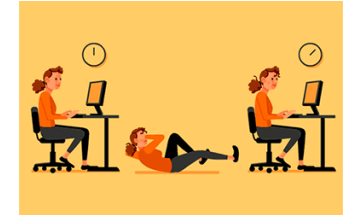


Repeatedly:

1. Observe features of user+articles
2. Choose a news article.
3. Observe click-or-not

**Goal: Maximize fraction of clicks**

Health advice?



Repeatedly:

1. Observe features of user+advice
2. Choose an advice.
3. Observe steps walked

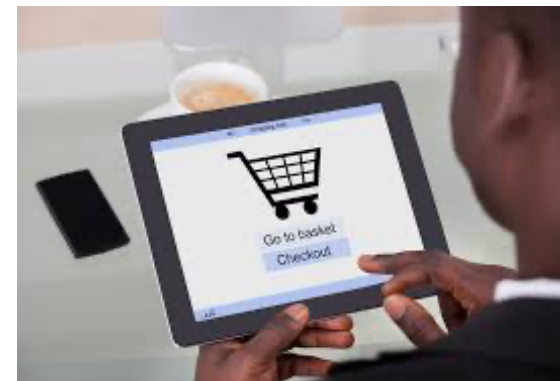
**Goal: Healthy behaviors (e.g. step count)**

The Amazon logo, consisting of the word "amazon" in a bold, black, sans-serif font with a curved orange arrow underneath it pointing from the letter 'a' to the letter 'z'.

Recommendations



buy or not buy



# Exploration vs. Exploitation in Contextual Bandits.

# Exploration vs. Exploitation in Contextual Bandits.

- Challenging because:
  - **Infinite state space**, never see the same context again.
  - **Exponentially large** policy space

# Exploration vs. Exploitation in Contextual Bandits.

- Challenging because:
  - **Infinite state space**, never see the same context again.
  - **Exponentially large** policy space
- Ideas:
  - ExploreFirst,  $\epsilon$ -Greedy  $O(T^{2/3})$



# Exploration vs. Exploitation in Contextual Bandits.

- Challenging because:
  - **Infinite state space**, never see the same context again.
  - **Exponentially large** policy space
- Ideas:
  - ExploreFirst,  $\epsilon$ -Greedy  $O(T^{2/3})$
  - UCB? But how do we construct Confidence Interval for an exponentially large set of policies?

# Exploration vs. Exploitation in Contextual Bandits.

- Challenging because:
  - **Infinite state space**, never see the same context again.
  - **Exponentially large** policy space
- Ideas:
  - ExploreFirst,  $\epsilon$ -Greedy  $O(T^{2/3})$
  - UCB? But how do we construct Confidence Interval for an exponentially large set of policies?
- Optimal regret:
$$O(\sqrt{KT \log |\Pi|})$$

# Next lecture

- Reinforcement learning for MDPs
  - Model-based vs model-free algorithms
  - Online policy iterations
  - Temporal difference learning
- Readings:
  - AIMA Ch. 21.1-21.3 (Ch 22.1- 22.3 in 4th Edition)
  - Sutton and Barto: Ch 4-6
  - Maybe: Sutton and Barto: Ch 6, Ch 13