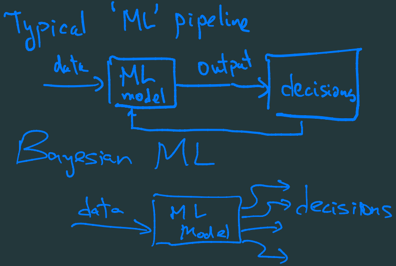
ORIE 4742 - Info Theory and Bayesian ML

Bayesian Decision Making

May 7, 2020

Sid Banerjee, ORIE, Cornell



decision theory in a nutshell

Bayesian decision theory in learning

drawn from posterior

given prior F on θ, choose 'action' $\hat{\theta}$ to minimize loss function $\mathbb{E}_F[L(\theta, \hat{\theta})]$

examples

- L_0 loss: $L(\theta, \hat{\theta}) = \mathbb{1}_{\{\theta \neq \hat{\theta}\}} \Rightarrow \hat{\theta_{L_0}} =$ mode of F (Eg - spam filtering COVID tests)
- L_1 loss: $L(\theta, \hat{\theta}) = ||\theta - \hat{\theta}||_1 \Rightarrow \hat{\theta_{L_1}} =$ median of θ under F
- L_2 loss: $L(\theta, \hat{\theta}) = ||\theta - \hat{\theta}||_2 \Rightarrow \hat{\theta_{L_2}} = \mathbb{E}_F[\theta]$

decision theory in 'decision-making'

given prior F on X, choose 'action' $a \in \mathcal{A}$ to minimize loss, i.e.

'test data'

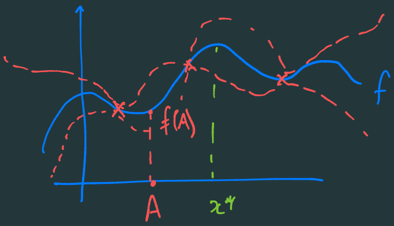
$$a^* = \arg\min_{a \in \mathcal{A}} \mathbb{E}_{\underbrace{X \sim F}_{\text{posterior for } X \text{ given data}}}[L(a, X)]$$

Eg - X ~ stock price in t days
p = price of stock today
$a \in \{0,1\}$, $1 =$ 'buy'
$L(0,X) = 0$, $L(1,X) = (X-p)\gamma^t$

example: Bayesian optimization

<u>Aim</u> - $\max \mathbb{E}[f(A)]$, f unknown

final choice of x



- Choose points $\underbrace{X_1, X_2, \ldots, X_s}_{\text{Samples}}$

Pick $A \in \mathbb{R}$ s.t. max $f(A)$

- decision problem - choice of $X_1, X_2, \ldots, X_s$, A

(easier problem - pick X_s, A given $X_1, \ldots, X_{s-1}$)

- 'Heuristic' - Pick X_s to maximize { Expected improvement / Knowledge gradient }

- pick A to max $\mathbb{E}[f(A) \mid X_1, \ldots, X_s]$

As an MDP: $X_1 \to f(X_1) \to X_2 = \Phi(X_1, f(X_1)) \to f(X_2) \to \ldots \to f(X_s) \to A \to f(A)$

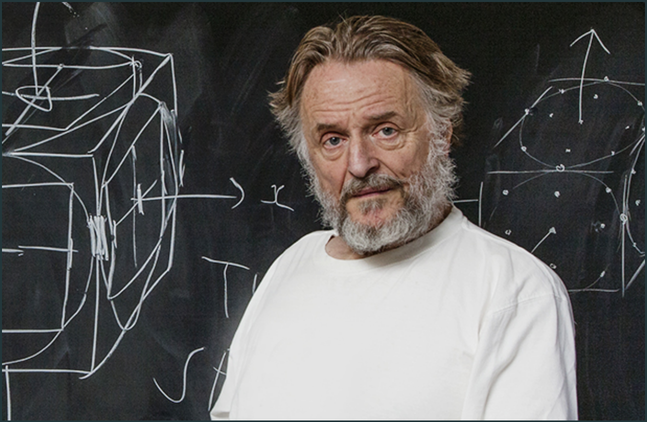
next, we play a game [stochastic variant of Nim]

- Setup: A pile of 10 toothpicks
- You will be playing against an oblivious random adversary (called Computer).
- A Sequence of Events in Each Iteration:
  - You start first. You can take <u>either 1 or 2</u> toothpicks from the pile.
  - After you make the decision, ~~I~~ the computer will flip a random fair coin. If the coin lands HEAD, the Computer will remove 1 toothpick from the pile. Otherwise, the Computer will remove 2 toothpicks.
- The game proceeds until all toothpicks are removed from the pile.
- If you end up holding the last toothpick, you win $20. Otherwise, you get nothing.

Courtesy: Paat Rusmevichientong

(note: this is a variant of a game called Nim; see Youtube video)

talking of playing games (in memorium)



Combinatorial game theory

for more on such games, see winning ways for mathematical plays

Conway, Berlekamp, Guy

analyzing the game (sequential decision making)

divide game into rounds:

– in each round, you go first followed by COMPUTER

– In k^{th} round, computer picks $X_k \sim Unif\{1, 2\}$ toothpicks

analyzing the game

divide game into rounds:

– in each round, you go first followed by COMPUTER

– In k^{th} round, computer picks $X_k \sim Unif\{1, 2\}$ toothpicks

observations

- if the game starts with 1 or 2 toothpicks, then we win!
  (if game starts with 0 toothpicks, assume we lose.)

analyzing the game

divide game into rounds:

– in each round, you go first followed by COMPUTER

– In k^{th} round, computer picks $X_k \sim Unif\{1, 2\}$ toothpicks

observations

- if the game starts with 1 or 2 toothpicks, then we win!
  (if game starts with $\leq$ 0 toothpicks, assume we lose.) (ie, if $S_k \leq 0$, then loss)
- suppose after $k - 1$ rounds, game has $S_k \geq 3$ toothpicks left, and let S_{k+1} be number of toothpicks left when we play next:
  - if we pick 1 match, then $S_{k+1} = S_k - 1 - X_k$
  - if we pick 2 match, then $S_{k+1} = S_k - 2 - X_k$

  decision

$$S_k \longrightarrow \underset{\in\{1,2\}}{\text{player picks}} \longrightarrow X_k \longrightarrow S_{k+1}$$

analyzing the game

divide game into rounds:

– in each round, you go first followed by COMPUTER

– In k^{th} round, computer picks $X_k \sim Unif\{1,2\}$ toothpicks

observations

- if the game starts with 1 or 2 toothpicks, then we win!
  (if game starts with 0 toothpicks, assume we lose.)
- suppose after $k-1$ rounds, game has $S_k \geq 3$ toothpicks left, and let S_{k+1} be number of toothpicks left when we play next:
  - if we pick 1 match, then $S_{k+1} = S_k - 1 - X_k$
  - if we pick 2 match, then $S_{k+1} = S_k - 2 - X_k$

to 'solve' this game, we use dynamic programming.

analyzing the game

- if after $k-1$ rounds, game has $S_k \geq 3$ toothpicks, and S_{k+1} is number of toothpicks when we play next:
  - If we pick 1 match, then $S_{k+1} = S_k - 1 - X_k$
  - If we pick 2 match, then $S_{k+1} = S_k - 2 - X_k$

  (where $X_k \sim Unif\{1, 2\}$)

let $V(x) = \max \mathbb{E}[\text{Reward}]$ if round starts with x toothpicks (Value fn)

analyzing the game

- if after $k-1$ rounds, game has $S_k \geq 3$ toothpicks, and S_{k+1} is number of toothpicks when we play next:
  - If we pick 1 match, then $S_{k+1} = S_k - 1 - X_k$
  - If we pick 2 match, then $S_{k+1} = S_k - 2 - X_k$

  (where $X_k \sim Unif\{1,2\}$)

let $V(x) = \max \mathbb{E}[\text{Reward}]$ if round starts with x toothpicks

- $V(-1) = V(0) = 0$, $V(1) = V(2) = 20$. Want to find $V(10)$

analyzing the game

- if after $k-1$ rounds, game has $S_k \geq 3$ toothpicks, and S_{k+1} is number of toothpicks when we play next:
  - If we pick 1 match, then $S_{k+1} = S_k - 1 - X_k$
  - If we pick 2 match, then $S_{k+1} = S_k - 2 - X_k$

  (where $X_k \sim \mathit{Unif}\{1, 2\}$)

let $V(x) = \max \mathbb{E}[\text{Reward}]$ if round starts with x toothpicks

- $V(-1) = V(0) = 0$, $V(1) = V(2) = 20$. Want to find $V(10)$
- $V(3) = \max \mathbb{E}[\text{Reward}]$ if round starts with 3 toothpicks

analyzing the game

- if after $k-1$ rounds, game has $S_k \geq 3$ toothpicks, and S_{k+1} is number of toothpicks when we play next:
  - If we pick 1 match, then $S_{k+1} = S_k - 1 - X_k$
  - If we pick 2 match, then $S_{k+1} = S_k - 2 - X_k$

  (where $X_k \sim Unif\{1, 2\}$)

let $V(x) = \max \mathbb{E}[\text{Reward}]$ if round starts with x toothpicks

- $V(-1) = V(0) = 0,\ V(1) = V(2) = 20.$ Want to find $V(10)$
- $V(3) = \max \mathbb{E}[\text{Reward}]$ if round starts with 3 toothpicks

$$= \max \left\{ \mathbb{E}[R \text{ if we pick 1 of 3}], \mathbb{E}[R \text{ if we pick 2 of 3}] \right\}$$

analyzing the game

- if after $k-1$ rounds, game has $S_k \geq 3$ toothpicks, and S_{k+1} is number of toothpicks when we play next:
  - If we pick 1 match, then $S_{k+1} = S_k - 1 - X_k$
  - If we pick 2 match, then $S_{k+1} = S_k - 2 - X_k$

  (where $X_k \sim Unif\{1,2\}$)

let $V(x) = \max \mathbb{E}[\text{Reward}]$ if round starts with x toothpicks

- $V(-1) = V(0) = 0$, $V(1) = V(2) = 20$. Want to find $V(10)$
- $V(3) = \max \mathbb{E}[\text{Reward}]$ if round starts with 3 toothpicks

$$= \max \left\{ \mathbb{E}[R \text{ if we pick 1 of 3}], \mathbb{E}[R \text{ if we pick 2 of 3}] \right\}$$

$$= \max \left\{ \mathbb{E}[V(3-1-X)], \mathbb{E}[V(3-2-X)] \right\}$$

analyzing the game

- if after $k-1$ rounds, game has $S_k \geq 3$ toothpicks, and S_{k+1} is number of toothpicks when we play next:
  - If we pick 1 match, then $S_{k+1} = S_k - 1 - X_k$
  - If we pick 2 match, then $S_{k+1} = S_k - 2 - X_k$

  (where $X_k \sim \mathit{Unif}\{1, 2\}$)

let $V(x) = \max \mathbb{E}[\text{Reward}]$ if round starts with x toothpicks

- $V(-1) = V(0) = 0$, $V(1) = V(2) = 20$. Want to find $V(10)$
- $V(3) = \max \mathbb{E}[\text{Reward}]$ if round starts with 3 toothpicks

$$= \max \left\{ \mathbb{E}[R \text{ if we pick 1 of 3}], \mathbb{E}[R \text{ if we pick 2 of 3}] \right\}$$

$$= \max \left\{ \mathbb{E}[V(3-1-X)], \mathbb{E}[V(3-2-X)] \right\} \quad X = \begin{cases} 1 & \text{wp } 1/2 \\ 2 & \text{wp } 1/2 \end{cases}$$

$$= \max \left\{ \underbrace{\left(\frac{V(1)+V(0)}{2} \right)}_{\frac{20+0}{2}=10}, \underbrace{\left(\frac{V(0)+V(-1)}{2} \right)}_{\frac{0+0}{2}=0} \right\} = \underline{10}$$

analyzing the game

$V(x) = \max \mathbb{E}[\text{Reward}]$ if round starts with x toothpicks

- $V(-1) = V(0) = 0$, $V(1) = V(2) = 20$. Want to find $V(10)$
- $V(3) = \max\left\{0.5\big(V(1) + V(0)\big), 0.5\big(V(0) + V(-1)\big)\right\} = 10$
- $V(4) = \max\left\{0.5\big(V(2) + V(1)\big), 0.5\big(V(1) + V(0)\big)\right\} = 20$
- $V(5) = \max\left\{0.5\big(V(3) + V(2)\big), 0.5\big(V(2) + V(1)\big)\right\} = 20$
- $V(6) = \max\left\{0.5\big(V(4) + V(3)\big), 0.5\big(V(3) + V(2)\big)\right\} = 15$
- $V(7) = \max\left\{0.5\big(V(5) + V(4)\big), 0.5\big(V(4) + V(3)\big)\right\} = 20$
- $V(8) = \max\left\{0.5\big(V(6) + V(5)\big), 0.5\big(V(5) + V(4)\big)\right\} = 20$
- $V(9) = \max\left\{0.5\big(V(7) + V(6)\big), 0.5\big(V(6) + V(5)\big)\right\} = 17.5$
- $V(10) = \max\left\{0.5\big(V(8) + V(7)\big), 0.5\big(V(7) + V(6)\big)\right\} = 20$

analyzing the game

$V(x) = \max \mathbb{E}[\text{Reward}]$ if round starts with x toothpicks

- $V(-1) = V(0) = 0$, $V(1) = V(2) = 20$. Want to find $V(10)$
- $V(3) = \max\left\{0.5\big(V(1) + V(0)\big), 0.5\big(V(0) + V(-1)\big)\right\} = 10$
- $V(4) = \max\left\{0.5\big(V(2) + V(1)\big), 0.5\big(V(1) + V(0)\big)\right\} = 20$
- $V(5) = \max\left\{0.5\big(V(3) + V(2)\big), 0.5\big(V(2) + V(1)\big)\right\} = 20$
- $V(6) = \max\left\{0.5\big(V(4) + V(3)\big), 0.5\big(V(3) + V(2)\big)\right\} = 15$
- $V(7) = \max\left\{0.5\big(V(5) + V(4)\big), 0.5\big(V(4) + V(3)\big)\right\} = 20$
- $V(8) = \max\left\{0.5\big(V(6) + V(5)\big), 0.5\big(V(5) + V(4)\big)\right\} = 20$
- $V(9) = \max\left\{0.5\big(V(7) + V(6)\big), 0.5\big(V(6) + V(5)\big)\right\} = 17.5$
- $V(10) = \max\left\{0.5\big(V(8) + V(7)\big), 0.5\big(V(7) + V(6)\big)\right\} = 20$

optimal policy: move to nearest multiple of 3

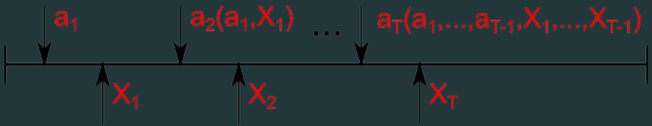
we always win if $x \neq 0 \mod (3)$

sequential decision making

Markov decision process (MDP)

general paradigm for sequential decision making

problem: $\max_{a:\text{“Actions”}} \mathbb{E}_X[f(X_1, a_1, X_2, a_2, \ldots, X_T, a_T)]$



main concepts

- state: S - summary of history $S_t = (a_1, X_1, a_2, X_2, \ldots, a_{t-1}, X_{t-1})$
- value function: $V(\cdot)$ - ‘value-to-go’ for given state (ie, Expected value earned by an ‘optimal’ policy)
- Bellman equation (or dynamic program equation):

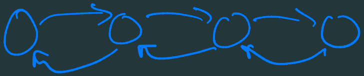
$$V(S_t) = \max_{a_t:\text{actions}} \mathbb{E}\Big[R_t(S_t, a_t) + V\Big(S_{t+1}(S_t, a_t)\Big)\Big]$$

Transition ‘kernel’ = distn over S_{t+1} given S_t, a_t

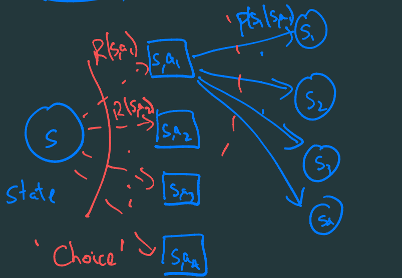
optimal policy: pick any a_t that is a maximizer of above eqn

Markov chain vs. Markov decision process

Markov chain

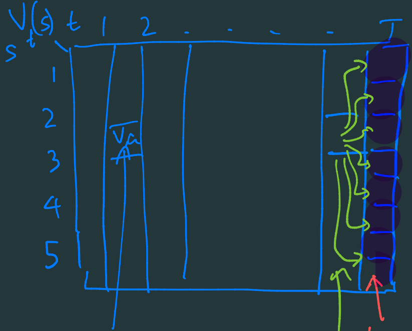


MDP



'Solution' to an MDP

$\mathcal{T} = \{1, 2, \ldots, T\}$, $S_t \in \{1, 2, \ldots, 5\}$



compute via Bellman Eqn

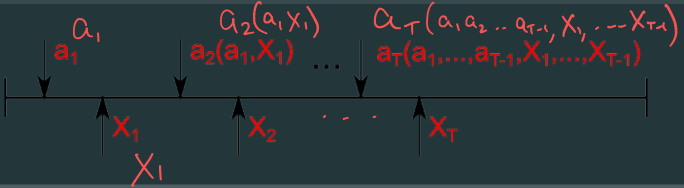
terminal conditions

for state s at time t store

- $V_t(s) = \max$
- $a^*_t(s) = \arg\max_a \left(\mathbb{E}\left[R_t(s_t,a) + V(\ldots)\right]\right)$

(finite horizon) MDP

sequential decision making: $\max\limits_{a:\text{"Actions"}} \mathbb{E}_X[f(a,X)]$

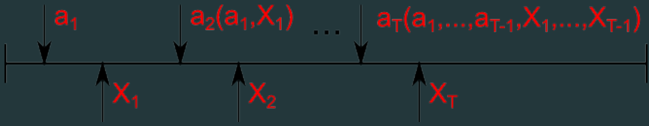


main concepts

- horizon: T - discrete 'decision periods' $t = \{1, 2, \ldots, T\}$

(finite horizon) MDP

sequential decision making: $\max_{a:\text{"Actions"}} \mathbb{E}_X[f(a, X)]$

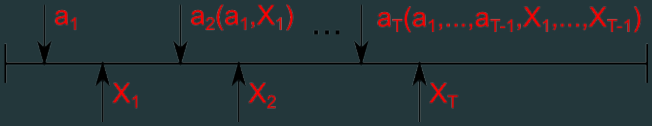


main concepts

- horizon: T - discrete 'decision periods' $t = \{1, 2, \ldots, T\}$
- state: $s_t \in \mathcal{S}_t$ - concise summary of history

(finite horizon) MDP

sequential decision making: $\max_{a:\text{“Actions”}} \mathbb{E}_X[f(a, X)]$

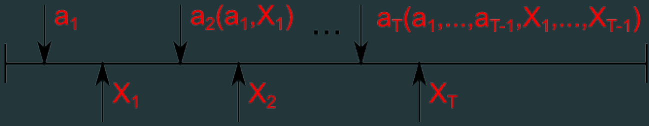


main concepts

- horizon: T - discrete ‘decision periods’ $t = \{1, 2, \ldots, T\}$
- state: $s_t \in \mathcal{S}_t$ - concise summary of history
- action: $a_t \in \mathcal{A}(s_t)$ - allowed set actions in each period

(finite horizon) MDP

sequential decision making: $\max_{a:\text{“Actions”}} \mathbb{E}_X[f(a, X)]$

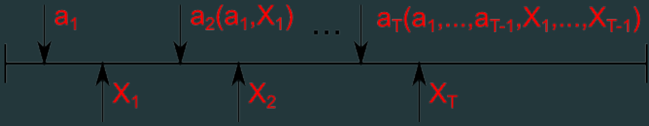


main concepts

- horizon: T - discrete ‘decision periods’ $t = \{1, 2, \ldots, T\}$
- state: $s_t \in \mathcal{S}_t$ - concise summary of history
- action: $a_t \in \mathcal{A}(s_t)$ - allowed set actions in each period
- randomness/disturbance: X_t - determines state transition probability $p(s_{t+1}|s_t, a_t)$ (or $s_{t+1} = f(s_t, a_t, X_t)$)

(finite horizon) MDP

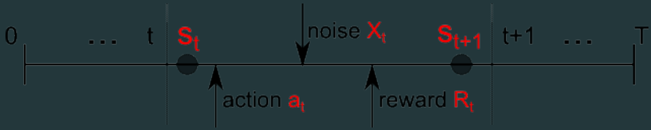
sequential decision making: $\max_{a: \text{"Actions"}} \mathbb{E}_X[f(a, X)]$



main concepts

- horizon: T - discrete 'decision periods' $t = \{1, 2, \ldots, T\}$
- state: $s_t \in \mathcal{S}_t$ - concise summary of history
- action: $a_t \in \mathcal{A}(s_t)$ - allowed set actions in each period
- randomness/disturbance: X_t - determines state transition probability $p(s_{t+1}|s_t, a_t)$ (or $s_{t+1} = f(s_t, a_t, X_t)$)
- Reward: $R_t(s_t, a_t, X_t)$ (or $R_t(s_{t+1}|s_t, a_t)$)

'solving' an MDP



dynamic programming

- value function: $V_t(s) \triangleq$ maximum expected expected reward over periods $\{t, t+1, \ldots, T\}$ starting from state s
- terminal conditions $V_T(s)$ for all s
- Bellman equation (or dynamic program equation):
  $V_t(S_t) = \max_{a_t:\text{actions}} \mathbb{E}\Big[R_t(S_t, a_t) + V_{t+1}\Big(S_{t+1}(S_t, a_t)\Big)\Big]$
  optimal policy: pick any a_t that is a maximizer of above eqn

example: distributing food to soup kitchens

- mobile food pantry has C meals to distribute between H soup kitchens
- kitchen i has demand $D_i \sim F_i$ (F_i is known)
- can choose to give $X_i \geq 0$ units of food (action)
- objective: maximize sum of log fill ratios $\underbrace{\sum_{i=1}^{H} \log\left(\frac{X_i}{D_i}\right)}_{\text{'proportional fair objective'}}$ 'filling ratio' $\min\left(\frac{X_i}{D_i}, 1\right)$ / Nash social welfare

- Check: If $D_1 = D_2 = \ldots = D_H > C/H$

optimal $X_i = C/H$

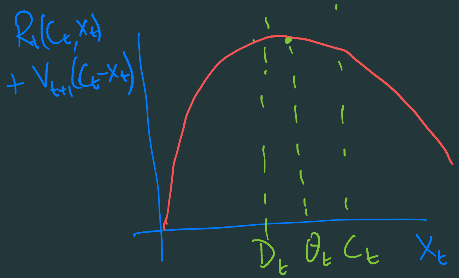
State: $S_t = C_t \equiv$ Amount of food left for $\{t, t+1, \ldots, H\}$

$A_t = X_t \equiv$ Amount " " given to location t

$$\boxed{V_t(C_t) = \max_{X_t : X_t \in [0, C_t]} \mathbb{E}\left[\log\left(\min\left(\frac{X_t}{D_t}, 1\right)\right) + V_{t+1}(C_t - X_t)\right]}$$

example: distributing food to soup kitchens

'Solution' - Threshold θ_t s.t. $X_t = \min(D_t, C_t, \theta_t)$



example: distributing food to soup kitchens

'Curse of dimensionality'

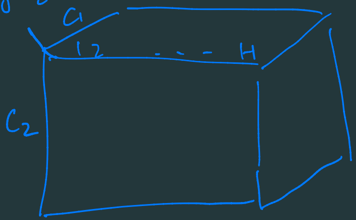
- mobile food pantry has C_j cans of item $j \in \{1, 2, \ldots, d\}$ to distribute between H soup kitchens
- kitchen i has demand $D_{ij} \sim F_i$ for item j
- can choose to give $X_{ij} \geq 0$ units of each item
- objective: maximize product of utilities $\prod_{i=1}^{H} \left(U_i(\sum_j v_{ij} \frac{X_{ij}}{D_{ij}}) \right)$

$$\approx \sum_{i=1}^{H} \log \left(U_i(\{X_{ij}, D_{ij}\}) \right)$$

If $j = 1$

'Complexity' $\approx CH$

If $j = 2$



'Complexity' $= C_1 . C_2 . H$

For general j:

complexity $= C_1 C_2 \ldots C_j H$

$\approx C^j H$

'solving' real MDPs

- exact solution via DP
  - newsvendor problem, selling single item ('convexity')
  - 'index' policies (greedy policies) - Gittin's index
- approximate methods (Thompson sampling)
  - Expected improvement / KG for BayesOpt
- iterative methods (value/policy iteration, Q learning)
  - approximate $V_t(s)$ (or $a_t^*(s)$) via some iteration
  - Q-learning (more generally, RL) - Solve the MDP approx^ly 'without knowing R, transitions'

example: the multi-armed bandit problem

- K actions, H horizon
- action $a \in [K]$ has reward $R(a) = Ber(\theta_a)$, with unknown θ
- aim: maximize $\sum_{t=1}^{H} R(A_t)$

Q: If you know $\{\theta_a\}$, what is your policy?

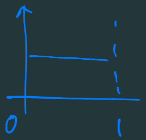
A: pick highest θ_a

$\theta_1 > \theta_2 > \theta_3$ $K=3$

- Exploration vs. Exploitation
- Examples of 'bad' policies - Equal play, fix arm
  - play each arm n times, for remaining $H-3n$, pick arm with highest MLE for θ_a
  - These perform badly ($\theta_1 H - \mathbb{E}[\text{Reward}] = \text{Regret} \approx cH$)

example: the multi-armed bandit problem (Bayesian model)

- <u>Idea</u> – Assume $\theta_a \sim \text{Beta}(1,1)$



  – Choose A_t via some rule

  – Update posterior $\theta_a \sim \text{Beta}(1+\underbrace{S_a}_{\text{success of a}},\ 1+\underbrace{F_a}_{\text{failures of a}})$

<u>Fact 1</u> – If $H \sim \text{Geom}(\gamma)$ then optimal solution for the MDP is known (Gittin's index)

<u>Fact 2</u> – For fixed H, if we sample $\theta_{at} \sim \text{Beta}(1+S_a[t],\ 1+F_a[t])$ and pick $A_t = \arg\max\{\theta_t\} \Rightarrow \mathbb{E}[\text{Regret}] = cK\log H$

Thompson sampling