- Bandits
- Reinforcement learning

Learning algorithm takes actions:

① influence what info is observed

② influence state of agent / world

Example: student selects classes

- action: choose a class to take
- information: you learn whether you like the class
  don't know about other classes
- state: take intro classes → more advanced class

Bandits:

- taking action gives info only about that action
- no state changes between actions

e.g. medicine

k treatment options for a condition w/ unknown effects.

Patients come in one at a time.

Action: prescribe one of k treatment.

Information: outcome of that patient

Early on, try all treatments. Eventually, learn which is best.

e.g. news headline

k headlines for article, want people to click

Action: for each visitor to website, show 1 of k headlines.

Info: did they click on website

Stochastic multi-armed bandits problems

rewards are random

Set of actions $\{1, \cdots, k\}$ "arms"

Each action a has reward dist $P_a(r)$ ← unknown ahead of time

what payoff you get

↑ fixed for all t

In bandit/RL, we maximize reward, not minimize loss

Players play this game for T rounds,

at each time $t = 1, \cdots, T$:

- player chooses action $A_t \in \{1, \cdots, k\}$ ← depend on $R_1, \cdots, R_{t-1}$, so A_t is a r.v.
- player receives rewards from $R_t \sim P_{A_t}(R)$

R_t: random variable

Goal: maximize total rewards $\sum_{t=1}^{T} R_t$

Example:

$k=2$

t	A_t	R_t
1	1	1
2	2	0
3	1	0
4	1	0
5	2	1
6	2	1
7	2	1

Action 1 is better, try more?

if $T=7$, then reward $= 4$

Regret: how well did your strategy do compared to the optimal strategy.

Define $\mu(a)$ = expected reward when choosing action a

$$= \mathbb{E}_{R \sim P_a(R)}[R]$$

optimal action $a^* = \arg\max_a \mu(a)$

Expected regret: expected difference between optimal strategy and player's strategy.

$$\underbrace{\mu(a^*) \cdot T}_{\text{expected reward optimal}} - \underbrace{\mathbb{E}\left[\sum_{t=1}^{T} R_t\right]}_{\text{expected reward for player}}$$

$$= \mu(a^*) \cdot T - \sum_{t=1}^{T} \mu(A_t)$$

Exploration v.s. Exploitation

exploration: try all the actions enough time to learn which is better
gain knowledge that's useful later

exploitation: use current knowledge to do what currently seems best

Algorithm: Upper Confidence Boundary (UCB)

Idea:
- player is estimating $\mu(a)$ for all a
- estimates are uncertain
  - ↳ represent this as a confidence interval
- At each t, choose action with largest upper bound

Why? Optimism in face of uncertainty

If a is action with largest upper bound:

either ① it's very good

② not good ⇒ update estimates and try sth else

UCB Algorithm: assume $0 \leq R_t \leq 1$

At time t, let $n_t(a)$ denote # of times we tried a up till time t

size of dataset collected about a

Let $\hat{\mu}_t(a)$ be sample mean rewards
when taking action a in the data before time t

How much uncertainty is in a sample mean?

In n examples, variance of sample mean $\frac{\sigma^2}{n}$

std of sample mean $\frac{\sigma}{\sqrt{n}}$

For UCB:

For action a, use CI of $\pm \sqrt{\frac{2 \log t}{n_t(a)}}$ at time t

$$\mu(a) \in \left[\hat{\mu}(a) \pm \sqrt{\frac{2 \log t}{n_t(a)}}\right]$$

$$UCB_t(a) = \hat{\mu}(a) + \sqrt{\frac{2 \log t}{n_t(a)}}$$

exploitation term

"a is good if reward is high"

exploration

"a is useful if we haven't tried it much yet"

Algorithm:

1. For $t = 1, \dots, k$: try each action once
2. For $t = k+1, \dots, T$: choose $A_t = \underset{a}{\text{argmax}}\ UCB_t(a)$

What happens to $\sqrt{\frac{2 \log t}{n_t(a)}}$ over time?

gets bigger slowly ↑ gets bigger over time

⇒ never rule over an action

⇒ UCB closer to $\hat{\mu}(a)$

⇒ do exploitation

⇒ if $n_t(a)$ constant

eventually this UCB gets larger

Can prove a bound on regret of UCB

$O(\sqrt{kT\log T})$

This is good b/c it's sublinear.

If average across timesteps,

average regret is $O\left(\frac{\sqrt{kT\log T}}{T}\right) \to 0$

as $T \to \infty$

After enough time, close to optimal strategy.