4/6/2023

Supervised Learning

Dataset $= \{(x^{(i)}, y^{(i)})\}_{i=1}^{n}$

Unsupervised Learning

Dataset $\{x^{(i)}\}_{i=1}^{n}$

Someone gives us dataset
Learning algorithm has no influence on what the dataset is
"passive"

Bandits & Reinforcement Learning

<u>Learning algorithm takes actions</u>:

① Influence what info is observed
ie data
② Influence state of agent / world

Example: Students selecting classes
- Action: choose a class to take
- <u>Information</u>: You learn whether you like the class you take, don't learn about other classes ✓
- <u>State</u>: Taking intro class unlocks more advanced classes ✗

<u>Bandits</u>

- Taking action gives info only about that action
- No state changing between actions

Medicine: K treatment options for a condition w/ unknown effectiveness

Patients come in one at a time

<u>Action</u>: Prescribe one of K treatments

<u>Information</u>: Outcome of that patient

Early on, try all the treatments
Eventually learn which is best & mostly prescribe that

$\boxed{\text{News Headlines}}$ K headlines for article, want people to click

Action: For each visitor to site, show 1 of the K headlines

Info: Did they click on article

Stochastic Multi-Armed Bandit Problems

Rewards are random

one-armed bandit = slot machine

K different slot machines, learn which gives most $$

Set of actions $\{1, .., K\}$ ("arms")

Each action a has reward distribution $P_a(r)$

Unknown ahead of time

Fixed for all t

What payoff did you receive?

- Health of patient after treatment
- Did person click or not (1 or 0)

(In bandits/RL we maximize reward and not minimizing loss)

Player play this game for T rounds

At each time $t = 1, \ldots, T$:

- Player chooses action $A_t \in \{1, \ldots, K\}$
- Player receive reward $R_t \sim P_{A_t}(R)$

Depends on observed rewards $R_1, \ldots, R_{t-1}$, so A_t is also a random variable

Random variable

$\in \mathbb{R}$

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

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 better, lets try it more

// maybe action 1 isn't that good?

If $T=7$ then total reward $= 4$

Regret: How well did your strategy do compared to the optimal strategy

Define $N(a) =$ expected reward when choosing action a $= \mathbb{E}_{R \sim P_a(R)} [R]$

optimal action $a^* = \underset{a}{\operatorname{argmax}} \; N(a)$

Expected Regret: Expected difference between optimal strategy & player's strategy

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

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

Regret close to 0 good

large regret is bad

Exploration vs Exploitation

Want to try all the actions enough times to learn which is better ie gain knowledge that's useful later

"I want to try lots of subjects before specializing"

Use current knowledge to do what currently seems best

"I like math classes I will keep taking math"

Algorithm: Upper Confidence Bound (UCB) Algorithm

Idea:

- Player is estimating $N(a)$ for every a
- Estimates are uncertain
  - we will represent this as a confidence interval
  - "I think $N(a)$ is between 0.6 and 0.8"
  - lower bound, upper bound
- At each t, choose action with largest upper bound

Why?: Optimism in the face of uncertainty

If a is action with largest upper bound:

Either ① it's actually good = very good news

② It's not so good ⇒ can update our estimates & try something else

UCB Algorithm: Assume $0 \le R_t \le 1$

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
⋮		

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

Size of dataset we collected about a

$n_8(1) = 3$, $n_8(2) = 4$

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

$\hat{\mu}_8(1) = 1/3$, $\hat{\mu}_8(2) = 3/4$

How much uncertainty is in a sample mean?

If n examples

Variance of sample mean $= \dfrac{\sigma^2}{n}$ ← variance of one sample

$\Leftrightarrow$ Standard deviation of sample mean is $\sigma/\sqrt{n}$

order $1/\sqrt{n}$

For UCB:

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

So it's $O\left(\dfrac{1}{\sqrt{n_t(a)}}\right)$

ie $\mu(a) \in \left[\hat{\mu}_t(a) - \sqrt{\dfrac{2\log t}{n_t(a)}},\ \underbrace{\hat{\mu}_t(a) + \sqrt{\dfrac{2\log t}{n_t(a)}}}_{= UCB_t(a)}\right]$

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

Gets smaller as $n_t(a)$ gets larger

Exploitation term
"a is good if we think its reward is high"

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

Full algorithm:

1. For $t = 1, \dots, K$: try each action once
2. For $t = K+1, \dots, T$: choose $A_t = \arg\max_a UCB_t(a)$

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

Gets bigger over time very slowly
$\Rightarrow$ we never completely rule out an action.

If $n_t(a)$ constant eventually this UCB gets large

Gets bigger over time
$\Rightarrow$ UCB gets closer to $\hat{N}$
$\Rightarrow$ do exploitation

Can prove a bound on Regret of UCB

in particular, Regret is $O\left(\sqrt{KT \log T}\right)$

This is good because it's Sublinear
If we average across timesteps, average regret is $O\left(\frac{\sqrt{KT\log T}}{T}\right) \to 0$ as $T \to \infty$

After enough time, gap w/ optimal Strategy is negligible.