11/9/2023 : Reinforcement Learning

Supervised Learning

Dataset = $\{(x^{(1)}, y^{(1)}), \dots\}$

input — desired output

Unsupervised Learning

Dataset = $\{x^{(1)}, x^{(2)}, \dots\}$

No supervision at all

Someone hands us a dataset
Algorithm takes dataset as input, doesn't influence what's in the dataset
"passive"

Reinforcement Learning

① Algorithm creates dataset as it runs

② Learning signal = Rewards for taking actions (i.e. did a good thing or bad thing happen)?

more than unsupervised learning

weaker than supervised learning

$\boxed{\text{Example}}$ Students selecting classes
- Action: Take some classes, not others
- State: What you know / what prereqs you satisfy
- Reward: Enjoyment, job

$\boxed{\text{Other examples}}$: Robotics, video games

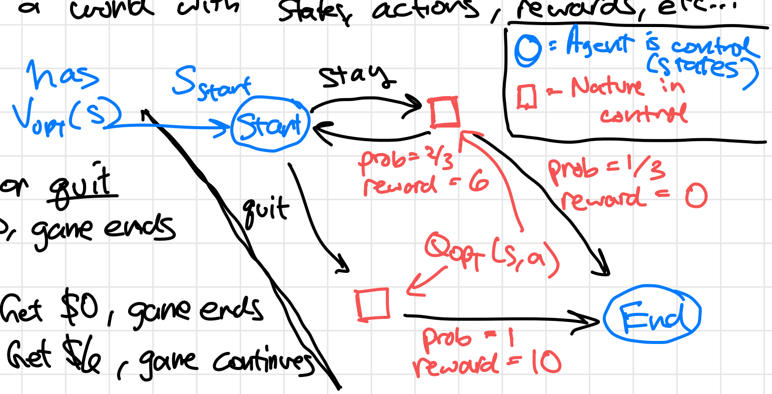
→ ~~①~~ Defining the world (No Learning)
→ ② Learning b/c the agent doesn't know how the world works

① Markov Decision Process (MDP)

Formal description of a world with states, actions, rewards, etc...

Example MDP:

At each timestep:
- Agent can stay or quit
- If quit: receive \$10, game ends
- If stay:
  - Probability 1/3: Get \$0, game ends
  - Probability 2/3: Get \$6, game continues



Formal ingredients of MDP:

- Set of states S (e.g. possible configurations / locations of robot)
- Starting state S_{start} (OR distribution over states)
- Actions (s): Possible actions at state s
- T(s, a, s'): Probability of transitioning to state s' starting at state s & taking action a
- Reward (s, a, s'): Reward received when transitioning from s to s' after taking action a

unknown in RL

In the example MDP:

T(Start, stay, End) = 1/3

Reward(Start, stay, End) = 0

- IsEnd(s): Is s an end state? Game ends when reaching an end state

---

Given a known MDP, what should the agent do?

$\boxed{\text{Policy}}$ Strategy used by an agent

Formally: mapping $\pi(s) \rightarrow \underbrace{a}_{\text{chosen action}} \in$ Actions (s)

current state

why? Visiting s multiple times has exact same transitions & rewards ⇒ best action is same

Value function: The value $V_\pi(s)$ for policy π and state s is: the expected (discounted) sum of rewards when starting at s, use policy π

Discounting: Future rewards are less valuable

- At each timestep, probability of survival < 1

we introduce a discount factor $\gamma \in [0, 1]$

= probability of survival at each timestep e.g. $\gamma = .99$

we care about discounted sum of rewards

For sequence r_1, r_2, r_3, discounted sum = $r_1 + \gamma r_2 + \gamma^2 r_3 + \cdots$

Optimal Policies

$\underbrace{V_{opt}(s)}_{\text{"optimal value"}}$ = maximum possible expected discounted sum of rewards starting at s for any policy

$\underbrace{Q_{opt}(s,a)}_{\text{"Q-value"}}$ = maximum possible expected discounted sum of rewards starting at s and forced to take action a

$$V_{opt}(s) = \begin{cases} 0 & \text{if } IsEnd(s) \\ \max_{a \in Actions(s)} Q_{opt}(s,a) & \text{else} \end{cases}$$

$$Q_{opt}(s,a) = \sum_{s' \in S} \underbrace{T(s,a,s')}_{\text{Prob of going to } s'} \Big[\underbrace{Reward(s,a,s')}_{\substack{\text{Reward now} \\ \text{(no discount)}}} + \underset{\text{discount}}{\gamma} \underbrace{V_{opt}(s')}_{\substack{\text{future} \\ \text{reward}}} \Big]$$

Optimal policy: $\pi^*(s) = \arg\max_{a \in Actions(s)} Q_{opt}(s,a)$

Lesson: If we can estimate $Q_{opt}(s,a)$ well, we can deduce the optimal policy ☆

Reinforcement Learning time

- Believe world is an MDP
- But $T(s,a,s')$, $Rewards(s,a,s')$ unknown to us

RL training pseudocode:

For episode = 1, 2, 3, ... :

$s_1 \leftarrow s_{start}$ // or sample distribution over start state

for $t = 1, \ldots$

- agent chooses action $a_t = \underbrace{\pi_{act}}_{\substack{\text{policy used to} \\ \text{act by agent}}}(s_t)$

- Agent receives:

Reward r_t

New state s_{t+1}

New data about the world

- Update agent's parameters

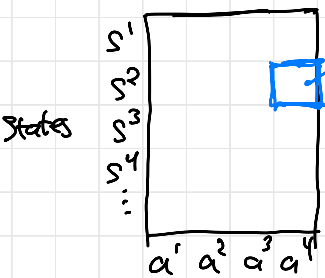
  AKA Learning

- If IsEnd(s_{t+1}): break

Learning algorithm: Q-Learning

Goal: Learn $Q_{opt}(s,a)$ for every (s,a)

We will maintain best guess $\hat{Q}(s,a)$ for each (s,a)



actions

$\hat{Q}(s^2, a^4)$ = our estimate of $Q_{opt}(s^2, a^4)$

each $\hat{Q}(s,a)$ is one parameter of model