File No.

Reinforcement Learning:

- Reinforcement Learning (RL) is a type of ML where an agent learns to behave in a environment by performing actions and seeing the results.
- RL system is comprised ~~of two~~ main components: (1) Agent (2) Environment.

RL Definations:

Agent: The RL algorithm that learns from trial and error.

Environment: The world through which the agent moves

Action (A): All the possible steps that the agent can take

State (S): Current Condition returned by the environment.

Reward (R): An Instant return from the environment to appraise the last action.

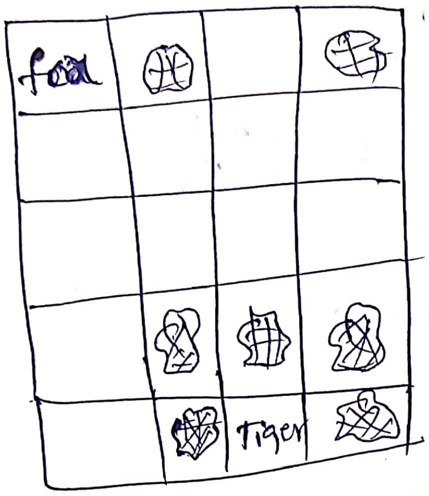
policy (π): The Approach that the agent uses to determine the next action based on the current state.

Value (V): The Expected long-term return with discount, as opposed to the Short-term Reward

Action-Value (Q): It is a value except it takes an extra parameter, the current action.

Reward Maximizations:

RL agent must be trained in such a way that he takes the best action. so that the reward is Maximum.



fox: Agent
Tiger: Opponent
▨ : Reward.

Goal: Maximum amount of meat taken by fox before killed by the tiger.

The basic ~~agent~~ function of the agent is maximize the reward So the agent must be trained in such a way that he takes the best action, so that the reward is maximum because the end goal of RL is to maximize the reward based on a set of actions.

Here fox is thinking cleverly it is eating that is closer to him rather than the meat which is closer to the tiger. Because closer is the tiger are more chances of getting killed.
So because of this the rewards which are near ~~tiger~~ the tiger, even they bigger meat chunks they will be discounted.
Agent is not going to ~~eat~~ eat the meat chunks which are closer to the tiger because of the risk. → It means that the agent is not going to explore ~~to eat meat~~
Discounting is represented by 'γ' (gamma).
'γ' is between '0' to '1'. smaller γ larger discount
larger γ smaller "
← which are closer to tiger.

- Exploitation ~~Exploration~~ is about using the already known exploited information to lighten the rewards
- Exploration is about exploring and capturing more information about an environment.

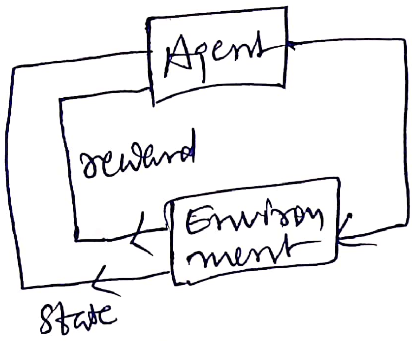
If foa uses exploitation it is safe
If foa user exploration it may be in danger.

Markov Decision Process:

The mathematical approach for mapping a solution in RL is called MDP.

The following parameters are used to attain a solution:

(1) Set of actions, A
(2) Set of states, S
(3) Reward, R (Collected defined value)
(4) policy, π (Series of actions to perform)
(5) Value, V
(Reward collected value is V.)



Note: maximize the rewards by choosing optimum policy.

Reinforcement Learning!

Bellman Equation!

→ It is central solving RL problems through methods like Value iteration, Policy Iteration, and Q-learning by iteratively applying the Bellman equation. agents can learn optimal policies that maximize cumulative rewards over time

The key elements used in:

- Action performed by the agent is 'a'
- State occurred by performing the action is 's'
- The Reward/feedback obtained for each good and bad action is 'R'
- The discount factor is 'γ' (Gamma)

then Bellman equation = $V(s) = \max[R(s,a) + \gamma V(s')]$

Where: $V(s)$ = Value calculated at a particular point

$R(s,a)$ = Reward at a particular state 's' by performing an action 'a'

γ = Discount factor

$V(s')$ = The value at the previous state.

Here we are taking the max of the complete values ∵ agent tries to find the optimal value

$\gamma = 0.9$ assume

find the values at each state start from S_3.

for S_3 block

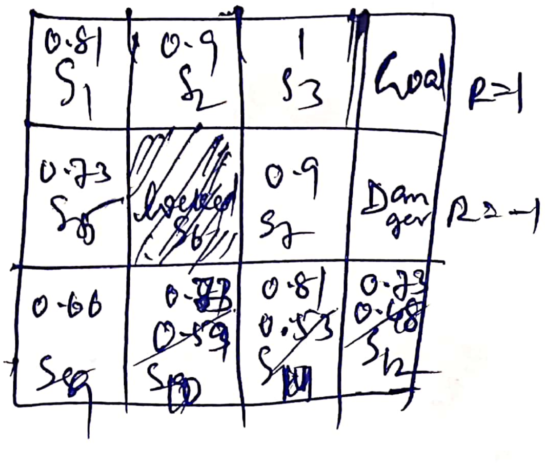
$V(s) = \max(R(s,a) + \gamma V(s))$

$= \max(1 + 0.9(0)) = 1$

$S_2 = \max(0 + 0.9 \times 1) = 0.9$

$S_1 = \max(0 + 0.9 \times 0.9) = 0.81$

$S_5 = \max(0 + 0.9 \times 0.81) = 0.73$



finally whenever Robot at any state it can find optimal path.

Q-Learning:

Through repeated iterations with the environment the Q-values converge to their optimal values, and the agent learns an optimal policy that maximizes the cumulative reward over time.

Policy is undertaken by agent (agent observes the given state and selects best possible action)

Policy: $Q^{\pi}(s_t, a_t)$ where s_t – state at time 't'
a_t – action at time 't'.

Reward: scalar quantity
correct action → reward

Penalty: commonly called as –ve reward.

Q-learning mainly depends on two factors:

1. Q-function: ①

$$Q^{\pi}(s_t, a_t) = E\left[R_{t+1} + \gamma R_{t+2} + \dots + \gamma^{n} R_{t+n} \middle| s_t, a_t\right]$$

⇓

Bellman Equation.

2. Q-Table: combination of actions and states.

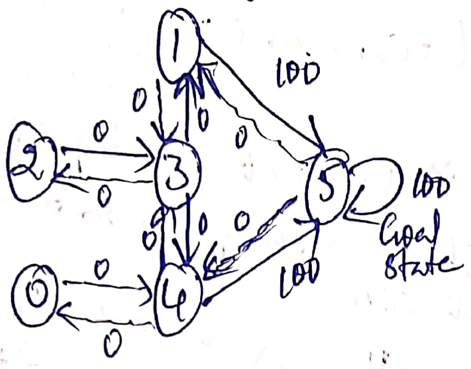
Ex: In a game ⇕
actions: up, down, left, right
state: start, end, reward, health, etc

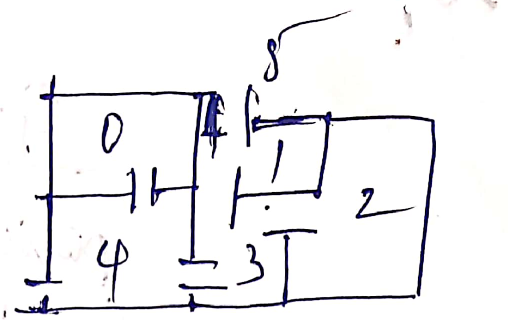
Steps followed:

(1) Exploration: explore all possible paths
(2). Exploitation: best possible path is identified
(3) Initialize a Q-table = 0
(4) Choose Action
(5) perform action
(6) measure reward
(7) update a Q-table

Q-learning

Eg:





We can put the state diagram and the instant reward values into the following reward table "matrix R"
the -1's represents null values

$$R = \begin{array}{c|cccccc} & 0 & 1 & 2 & 3 & 4 & 5 \\ \hline 0 & -1 & -1 & -1 & -1 & 0 & -1 \\ 1 & -1 & -1 & -1 & 0 & -1 & 100 \\ 2 & -1 & -1 & -1 & 0 & -1 & -1 \\ 3 & -1 & 0 & 0 & -1 & 0 & -1 \\ 4 & 0 & -1 & -1 & 0 & -1 & 100 \\ 5 & -1 & 0 & -1 & -1 & 0 & 100 \end{array}$$

States   actions

learning rate $= \gamma = \underline{0.8}$

$$Q = \begin{array}{c|cccccc} & 0 & 1 & 2 & 3 & 4 & 5 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 & 0 & 0 \end{array} \Rightarrow \begin{array}{c|cccccc} & 0 & 1 & 2 & 3 & 4 & 5 \\ \hline 0 & 0 & 0 & 0 & 0 & 80 & 0 \\ 1 & 0 & 0 & 0 & 64 & 0 & 100 \\ 2 & 0 & 0 & 0 & 64 & 0 & 0 \\ 3 & 0 & 80 & 51 & 0 & 80 & 0 \\ 4 & 64 & 0 & 0 & 64 & 0 & 100 \\ 5 & 0 & 80 & 0 & 0 & 80 & 100 \end{array}$$

$Q(\text{state}, \text{action}) = R(\text{state}, \text{action}) + \gamma * \text{Max}(\text{next state}, \text{all actions})]$

$Q(1,5) = R(1,5) + 0.8 * \max(Q(5,1), Q(5,4), Q(5,5))$
$= 100 + 0.8 * 0 = 100$

$Q(3,1) = R(3,1) + 0.8 * \max(Q(1,3), Q(1,5)) = 0 + 0.8 \times 100 = 80$

⋮

like tracing the best sequence