* Fuzzy Logic :-
↳ description of statement

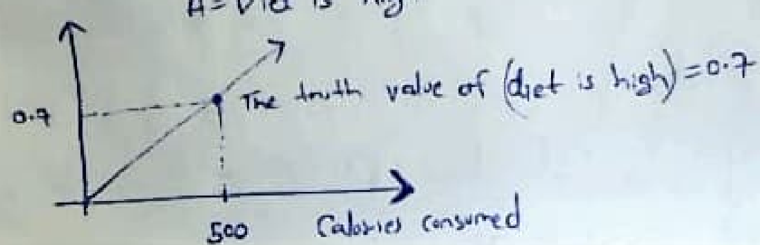
eg- The person is tall.

- Membership function
- Defuzzification

What is the measurement / % of a statement to be T/F.

0 − 1 → fuzzy mode → 0*, 0.8, 1*

A = Diet is high



And → weaker condition → $T(A \wedge B) = \min\{T(A), T(B)\}$

OR → Stronger condition. → $T(A \vee B) = \max\{T(A), T(B)\}$

eg :- (Sl. do)

a) $f_{\text{diet high}} = \frac{1}{5000} x = \frac{3000}{5000} = 0.6$ → Membership for diet high for this patient = 0.6

b) $f_{\text{exercise high}} = \frac{1}{2000} x = \frac{1000}{2000} = 0.5$

Fuzzy rules

1) T_{truth}(diet low) ∧ Truth (ex high) = min (0.4, 0.5) = 0.4

2) T(diet high) ∨ T(Ex low) = max (0.6, 0.5) = 0.6

3) Balanced ⇒ Risk low = 0.4

4) Unbalanced ⇒ Risk high = 0.6

$T(\text{high risk}) = \frac{1}{125} x$ (x = likelihood of heart disease)

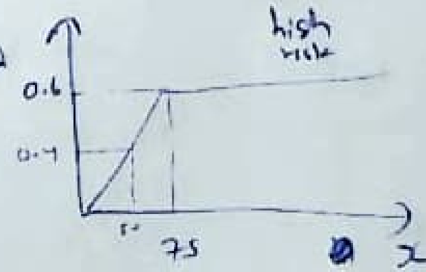
$0.6 = \frac{x}{125}$ ⇒ $x = 75$

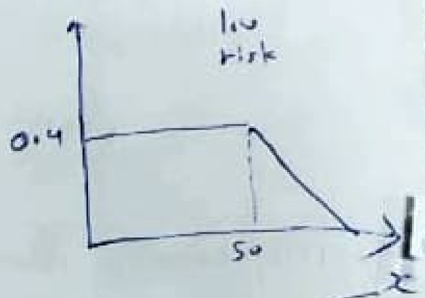
$f(\text{risk low}) = 0.8 - \frac{x}{125}$

$x = 50$

intersection / And

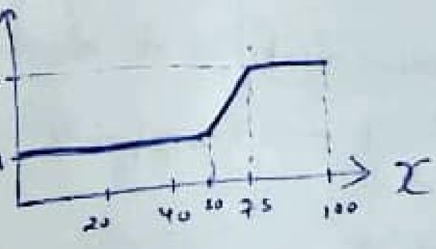
Patient specific





Aggregated Risk function

Truth value (Risk high) $\vee$ Truth value (Risk low)



likelihood of heart disease

→ Defuzzification :-

likelihood of heart disease of this person ?

$$\int_0^{100} f(\text{aggregated risk})\,dx = \int_0^{50} 0.4\,dx + \int_{50}^{75} \frac{x}{125}\,dx + \int_{75}^{100} 0.6\,dx$$

$$= 20 + \left[\frac{x^2}{2}\right]_{50}^{75} + 15$$

$$= \boxed{47.5\%}$$

→ Evaluate the risk

→ likelihood → defuzzyfication

* State machine.

→ Bayesian classifier → Confidence / Belief score

Softmax → entropy → uncertainity

* MDP (Markov Decision Process) :-

→ Planning in Uncertain Environment

→ Learning ⇝ interaction

⇒ Reinforcement learning vs supervised learning

⇓

$\pi : S \mapsto A$

Policy — state — Action

→ Take action A to affect states.

→ Value function

Predicted value

↳ Can't figured out immediately.

→ State, Actions

Transition Probability :- $P_r(S_{t+1} \mid S'_t, A_t)$

* Markov Property $a \to b \to c \to d \to \overset{t}{e} \rightsquigarrow$

↓

Given the current S & A, the next state is independent of all privious state & actions.

↓

So, memory-less

Reward : $R(S) \to$ real value

Find a policy : $\pi : S \longrightarrow A$ to maximize reward

maximize expected reward $E[r'_t \mid \pi, S_t]$ for all state S ~~states~~

Constraint: The agent has "t" timestep to complete the goal.

$$E\left[\sum_{k=0}^{t} r_k' \,\middle|\, \pi, s_0\right] \rightarrow \text{maximize}$$

$\hookrightarrow$ t timestep

→ if $t = \infty$ → infinite time horizon.

Sooner ↓, more reward ↑

$$E\left[\sum_{k=0}^{\infty} \gamma\, r_k' \,\middle|\, \pi, s_0\right] \rightarrow \text{discount factor}$$

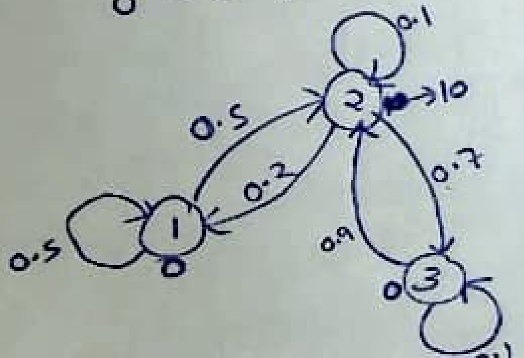
$\gamma = 0.9$

Immediate reward + future reward

Value of state

$$v(s) = R(s) + \gamma \sum_{s'} P(s' \mid s)\, v(s')$$

eg:-



*RL

Markov Chain MDP

→ S
→ A
→ TP
→ R
→ V
→ γ discount factor

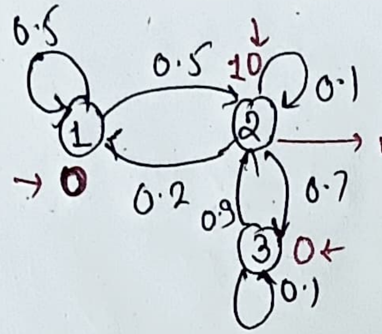
Reward : $R(S)$

Value of a State

$V(s)$: How much total reward do we expect to get if we start from state S

$$V(S) = \underbrace{R(s)}_{\text{immediate reward}} + \underbrace{\gamma \times \sum_{s'} P(s'|s) \cdot V(s')}_{\text{Expected long term reward}}$$

**



→ most rewarded state

$\gamma = 0.9$

V.

Assume $\gamma = 0.9$

$$V(1) = 0 + 0.9\left(\underbrace{0.5}_{P(S_1)} \times V(1) + 0.5 \times V(2)\right)$$

$$V(2) = 10 + 0.9\left(0.1 \times V(2) + 0.2 \times V(1) + 0.7 \times V(3)\right)$$

$$V(3) = 0 + 0.9\left(0.1 \times V(3) + 0.9 \times V(2)\right)$$

$V(1) = 40.5$ $\quad \boxed{V(2) = 49.5}$ $\quad V(3) = 49.1$

Discount factor γ determines :
the weight given to future rewards compared to current reward
(immediate)

SUPERVISED LEARNING

Task : $\underbrace{\pi}_{\text{Policy}} : S \longrightarrow A$ learn from experience

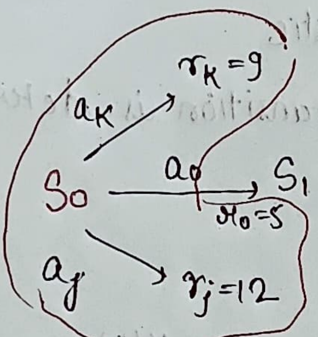
State space
↑
explore ⟶ RL algo

Reinforcement learning is ~~lega~~ learning from interaction / experience

what the agent tries to ~~mod~~ optimize ?

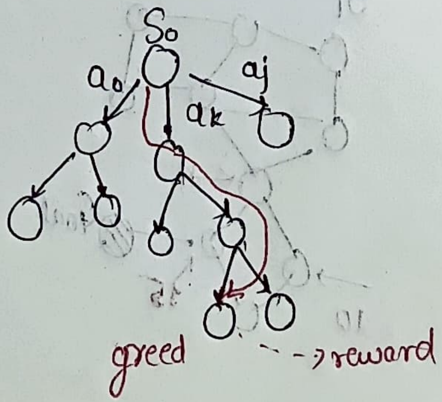
$$S_0 \xrightarrow[r_0]{a_0} S_1 \xrightarrow[r_1]{a_1} S_2 \longrightarrow \cdots \longrightarrow$$

⇒ The total future discounted reward



greedy choice

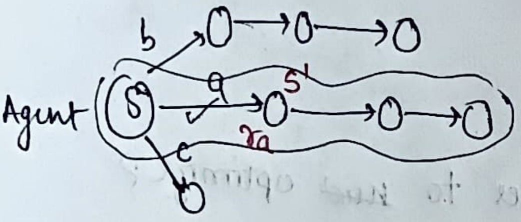
$$V^{\pi}(S_t) = r_t + \gamma r_{t+1} + \gamma\gamma_{t+2} \cdots$$



** What would be the optimal policy

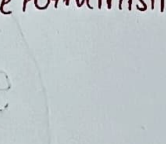
$$\pi^*(S) = \underset{a}{\operatorname{argmax}} \Big\{ \underbrace{r(S,a)}_{\text{immediate value}} + \gamma . \underbrace{V^*[\delta(S,a)]}_{\text{future reward}} \Big\}$$

$$\pi^*(S) = \underset{a}{\operatorname{argmax}} \{ \quad - \quad - \quad . \quad \}$$



maximum reward if action a is taken

reward is delayed

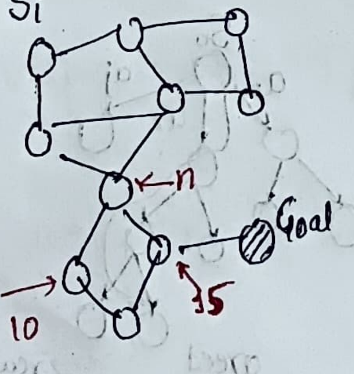


Assumption

1. We know V^*
2. reward is deterministic
3. $S_{t+1} = \delta(S_t, a)$, transition is deterministic

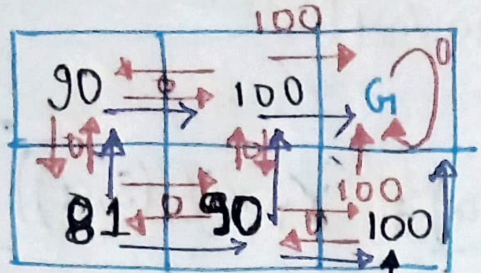
Shortest path

Si



$$V(S) = \frac{1}{\text{distance}}$$

⇒ If you know $V(S)$, The problem is trivial



v*(S)

immediate

$r + \gamma$ Future

$\gamma = 0.9$

v* (s) value

optimal: Take highest v* of neighbouring state.

▷ Agent cannot see the complete state space

Solving a RL Problem

→ Dynamic Programming
→ Monte carlo
→ Temporal

Compute V^π from π

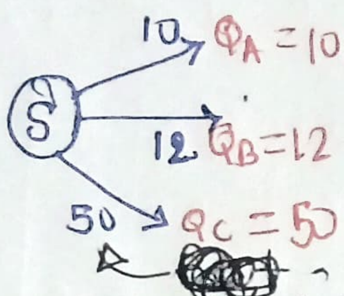
Improve π based on V^π

Bellman Equation

$$v^*(s) \leftarrow \max_a \Big[\underbrace{r(s,a)}_{\text{Immediate}} + \underbrace{\gamma\, v^*(\delta(s,a))}_{\text{Future}}\Big]$$

Learn $(r(s,a), \delta(s,a)]$

Q-Function: Learns good state-action pair



Good state action pair depend on max value of Q-function

Q-Learning

$$\pi^*(s) = \arg\max_a Q(s,a) \qquad \boxed{Q(s,a)}$$

$$v^*(s) = \max_a Q(s,a)$$