Date (M|T|W|T|F|S)

RL Final

(Epsilon soft policy, On policy / Off policy)

Onpolicy - behavioral policy = target policy
(agent)
(take action) (learn)

.. Epsilon soft, Epsilon greedy

non-optimal

Epsilon soft is good in case of behavioral policy because it gives us exploration

but not in case of target because epsilon soft stops us from taking greedy action everytime

Epsilon soft is not bad because exploration is necessary too but it is also not optimal

---

Off policy

behavioral policy ≠ target policy

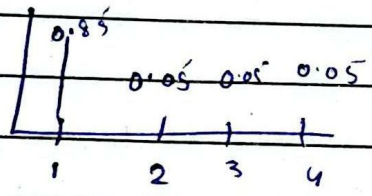
policy
probability of taking action

behavioral policy — probability distribution
Target policy —

e.g

$X \rightarrow$ random variable

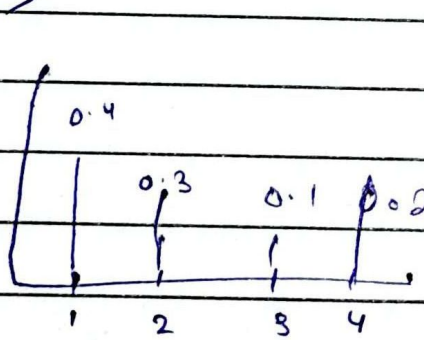
$b(x)$



$$E(X) = \sum_{x \in X} x \cdot b(x)$$

$$= 1(0.85) + 2(0.05) + 3(0.05) + 4(0.05)$$

$\pi(x)$



$$= 1(0.4) + 2(0.3) + 3(0.1) + 4(0.2)$$

if probably is not given

we repeat experiment for a large no of time then Calculate mean

Date _____ M T W T F S

Castelli

Now we want to determine the Estimated value of v under π by using samples from B.

~~No~~ ~~We~~ Now the courses of b and π is different

We modify the values of B such a way that we get expected value of x

Important Sampling (sample generate from behavior policy but we have to modify their return so it sets equal to the expected value of target policy)

$$E_\pi(X) = \sum_{x \in X} x \cdot \pi(x)$$

$$= \sum_{x \in X} x \frac{b(x)}{b(x)} \pi(x)$$

$$= \left(\sum_{x \in X} x\, b(x) \right) \left(\frac{\pi(x)}{b(x)} \right)$$

Expected value — outcome i.e. Importance sampling

How much importance is given to it

$$= E_b(X) \cdot \rho(x)$$

$$= \frac{1}{n} \sum_{i=1}^{n} x_i \; \rho(x)$$

average weight

Date _____ M T W T F S

for above example

$$\frac{1 \times \frac{0.4}{0.85} + 1 \times \frac{0.4}{0.85} + 1 \times \frac{0.4}{0.85} + 2 \times \frac{0.3}{0.05} + 3 \times \frac{0.1}{0.05}}{\text{no of samples}}$$

How to find $\rho(x)$ in Monte Carlo

$$\rho = \frac{Pr(\text{trajectory under } \pi)}{Pr(\text{trajectory under } b)} \rightarrow \text{target} \rightarrow \text{behaviour}$$

Importance sampling estimates the value functions for a policy π with samples collected previously from an older policy $\bar{\pi}$

Date [M][T][W][T][F][S]

How to find Important sampling in Monte Carlo

$$\rho = \frac{Pr(\text{trajectory under } \pi)}{Pr(\text{trajectory under } b)} \rightarrow \text{behaviour} \quad \rightarrow \text{target}$$

$$Pr(\text{trajectory under } \pi) = Pr(S_0, A_0, S_1, A_1, S_2, A_2, S_3 \cdots A_T, S_T)$$

↓ trajectory under π

$$= \pi(A_0|S_0)\, P(S_1|S_0, A_0)$$

$$= Pr(A_t, S_{t+1}, A_{t+1}, S_{t+2}, A_{t+2} \cdots A_T, S_T | S_t, A_t)$$

$$= \pi(A_t|S_t)\, p(S_{t+1}|S_t, A_t)$$

$$= \pi(A_{t+1}|S_{t+1})\, p(S_{t+2}|S_{t+1}, A_{t+1})$$

$$\vdots$$

terminal state

$$Pr(\text{trajectory under } \pi) = \prod_{k=t}^{T-1} \pi(A_k|S_k)\, p(S_{k+1}|S_k, A_k)$$

$$Pr(\text{trajectory under } b) \rightarrow \prod_{k=t}^{T-1} b(A_k|S_k)\, p(S_{k+1}|S_k, A_k)$$

So

$$\rho_{t:T-1} = \prod_{k=t}^{T-1} \frac{\pi(A_k|S_k)}{b(A_k|S_k)}$$

Page 19

Date 20, 21, 22, 23 (M T W T F S)

$$Q_{t-1} = R_t + G_t$$

$$Q_{T-1} = \frac{\pi(A_{T-1} | S_{T-1})}{b(A_{T-1} | S_{T-1})}$$

$$G_{T-1} = R_{t+1}$$

$$G_{T-1} * Q_{T-1}$$

$$G_{T-2} = R_{T-1} + R_{T-2}$$

$$= G_{T-1} + R_{T-2}$$

$$G_{T-2} * Q_{T-2}$$

You have to multiply your samples with a value so that you modify your behavior policy towards target policy

W → Importance sampling ratio

Onpolicy → Off policy

Multiply sample by an Importance sampling ratio

$$G \leftarrow \gamma W G + R_{t+1}$$

Date

M T W T F S

Temporal Difference (Sample based)

$V(S_t)$ = Average (Returns)

= Average {Returns[0] + Returns[1] + ...}

$$V(S_t) = \frac{1}{n-1} \sum_{i=1}^{n-1} G_i$$

MAB $$Q_{n+1} = \frac{1}{n} \sum_{i=1}^{n} R_i$$

$$= \frac{1}{n} \left[\sum_{i=1}^{n-1} R_i + R_n \right]$$

$$= \frac{1}{n} \left[\frac{n-1}{n-1} \sum_{i=1}^{n-1} R_i + R_n \right]$$

$$= \frac{1}{n} \left[(n-1) Q_n + R_n \right]$$

$$= \frac{1}{n} \left[n Q_n - Q_n + R_n \right]$$

$$Q_{n+1} = Q_n + \frac{1}{n} \left[R_n - Q_n \right]$$

New estimate = old + step size [Target - old estimate]

Incremental update in Monte Carlo

$$V(S_t) = \frac{1}{n} \sum_{i=1}^{n} G_n = V(S_t) + \alpha \left[G_t - V(S_t) \right]$$

Problem: We can't find V(s) until the episode is finished

(30) —— 5 —— (35) —— (15) —— (10) —— (3)

V(S.)

Date _____ M T W T F S

$$V(S_t) = V(S_t) + \alpha [G_t - V(S_t)]$$

In bellman equation we can estimate $V(s)$ by from the next $(V(s'))$

DP (Bootstrapping) based on some values you can estimate future values.

Temporal Difference { Monte Carlo — Sampling
DP — Bootstrapping

Bellman

$$V(S_t) = \mathbb{E}(G_t | S_t = s)$$

$$G_t = R_{t+1} + \gamma G_{t+1}$$

$$V_\pi(S_t) = \mathbb{E}(R_{t+1} + \gamma G_{t+1} | S_t = s)$$

$$V_\pi(S_t) = \mathbb{E}(R_{t+1} + \gamma V(S_{t+1} | S_t = s))$$

$$G_t = R_{t+1} + \gamma V(S_{t+1})$$

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots$$

Date M T W T F S

$$G_t = R_{t+1} + \gamma G_{t+1}$$

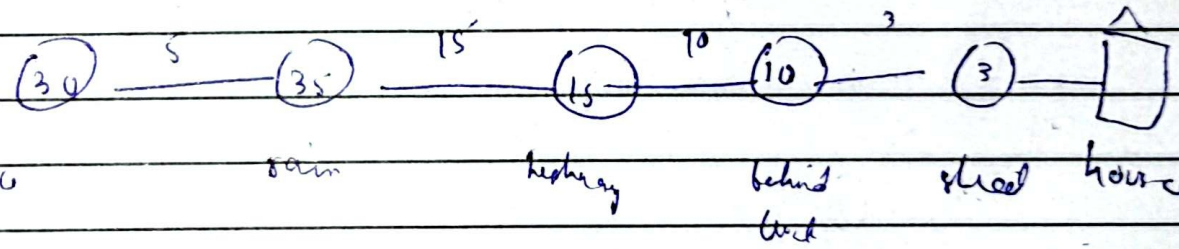
$$G_t \approx R_{t+1} + \gamma (V(S_{t+1}))$$

$$G_t \approx R_{t+1} + \gamma V(S_{t+1})$$

$$V(S_t) = V(S_t) + \alpha [G_t - V(S_{t+1})]$$

Temporal Difference

$$V(S_t) = \alpha [R_{t+1} + \gamma V(S_{t+1}) - V(S_t)]$$



$$V(S_1) = 30 + 1[5 + 1(35) - 30] \Rightarrow 30 + 5 = 35$$

$$V(S_2) = 35 + 1[15 + 1(15) - 35] \Rightarrow 35 - 5 = 30$$

Tabula — TO (0)

looking one step ahead.

Tabular

number of states is finite

Date M T W T F S

Intro to Q planning and Dyna Q

Multi-armed bandits

learning — Estimate $\rightarrow Q_t(A_t) = \sum_{i=1}^{t-1} R_i$

MDP:

Action states

Model based is planning

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r|s, a)[r + \gamma V(s')]$$

$$q(s,a) = \sum_{s'} \sum_r p(s', r|s, a)\left[r + \gamma \sum_{a'} \pi(a'|s') q(s', a')\right]$$

Control:

policy iteration
value iteration

Model free learning

Monte Carlo — prediction
→ control — on policy
— off policy

TD — prediction
→ control → SARSA

Date (M T W T F S)

$$Q(S_t, A_t) = Q(S_t, A_t) + \alpha[R_t + \gamma Q(S', A') - Q(S_t, A_t)]$$

(1) learning

$$Q(S_t, A_t) = Q(S_t, A_t) + \alpha[R_t + \gamma \max_a Q(S', a) - Q(S_t, A_t)]$$

off policy

* If we dont have probabilities we have to do a large number of experiment to find the ultimate

time and memory

Distribution model $\xrightarrow{\text{Experience}}$ Sample Model $\xrightarrow{\text{planning}}$ value

Distribution model $\xrightarrow{\text{planning}}$ values

Sample $\xrightarrow{\text{learning}}$ value, policy

Date M T W T F S

Castelli

Q-planning

~~loop forever:~~
~~select a state S~~

Loop forever:
1. select S, A at random
2. send S, A to a sample model and obtain a sample next reward, R, and a sample next state, S'
3. Apply one-step tabular Q-learning to S, A, R, S'.

$$Q(S,A) \leftarrow Q(S,A) + \alpha [R + \gamma \max_a Q(S',a) - Q(S,A)]$$

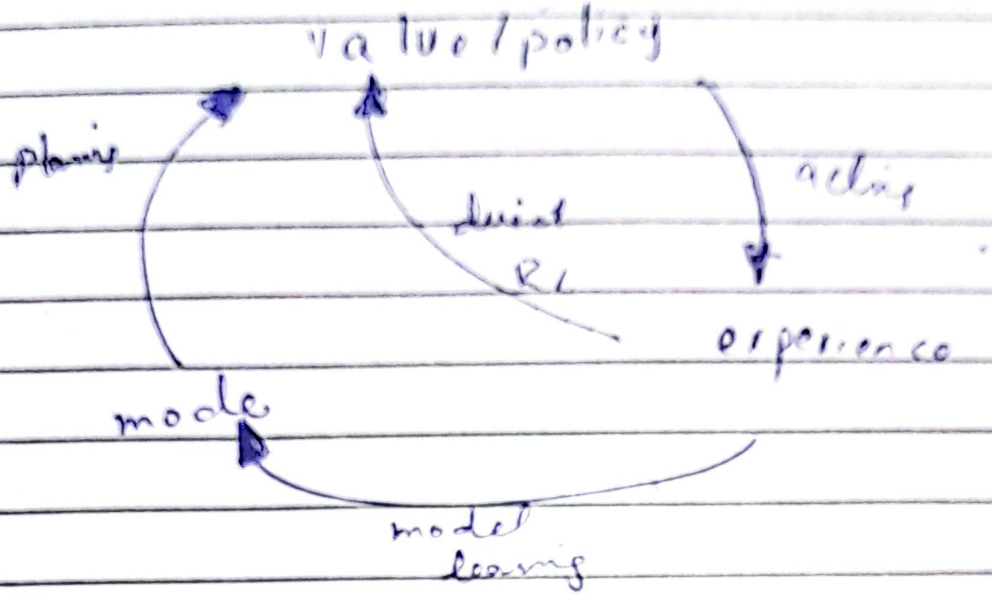
Sample Model
↓
Simulated Experience

Dyna Architecture

Dynamic ~~Sample model~~

→ learning → planning → Acting

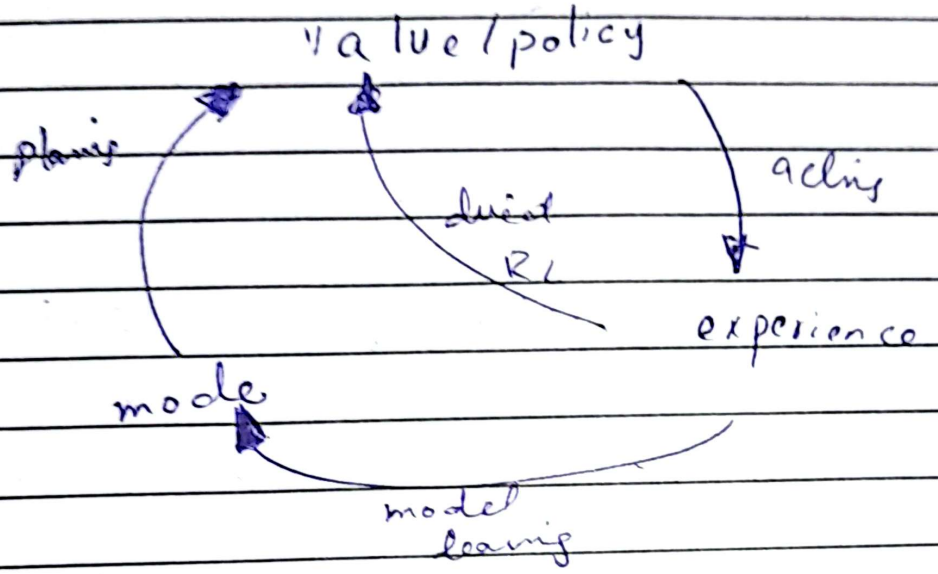
learning → Planning → Acting



Dyna Q → for learning and planning using Q-learning algorithm

robot standing on a cliff

Date _____ M T W T F S



Dyna Q → for learning and planning using Q algorithm

robot standing on a cliff

Date _____ M T W T F S

Castelli

Dyna Q

Dyna Q

~~action must be deterministic~~

Why the actions are deterministic in Dyna Q

~~if~~ the action is ~~deter~~ stochastic

then the model will not be useful

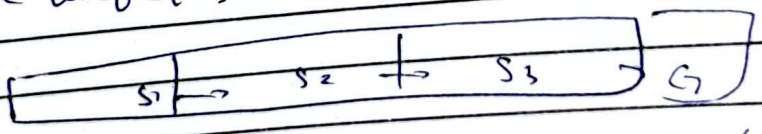
For planning

first the action

Stochastic Model fail not be useful.

Model $(S_1, A_1) = S_2, 0$

$S_1, A_1 = S_2, 1$



1st episode planning doesn't help you much

S, A	R, S'
S_1, R_1	$0, S_2$
S_2, R_2	$0, S_3$
S_3, R_3	$1, G$

learns $Q(S_1, R_1) = 0 + 1[0 + 0.9(0) - 0]$

$Q(S_1, R_1) = 0$

as initially all Q values are 0

planning Repeat n times S_1, S_2

still 0

Date

M T W T F S

Castelli

In next step

$$Q(S_3, R) = 0 + 1[1 + 0.9(0) - 0]$$
$$= 1$$

$$Q(S_3, R) = 1 ; G$$

Planning

Planning random choose S_2 S_2 update because of S_3

$$Q(S_2, R) = 0 + 1[0 + 0.9(1) - 0]$$

max

$$= 0.9$$

Next time S_1

Problems

Model Incomplete

Model Inaccurate

↳ If an action is inaccurate

So we update those action time reward larger so that agent chooses them next time and update their values.

Dyna Q+

we dynamically update reward based on how less frequent that is visited

Those state action who are updated after long time and not done, we make their reward larger so that agent chooses them

$$R(A) = R + \kappa \sqrt{\tau}$$

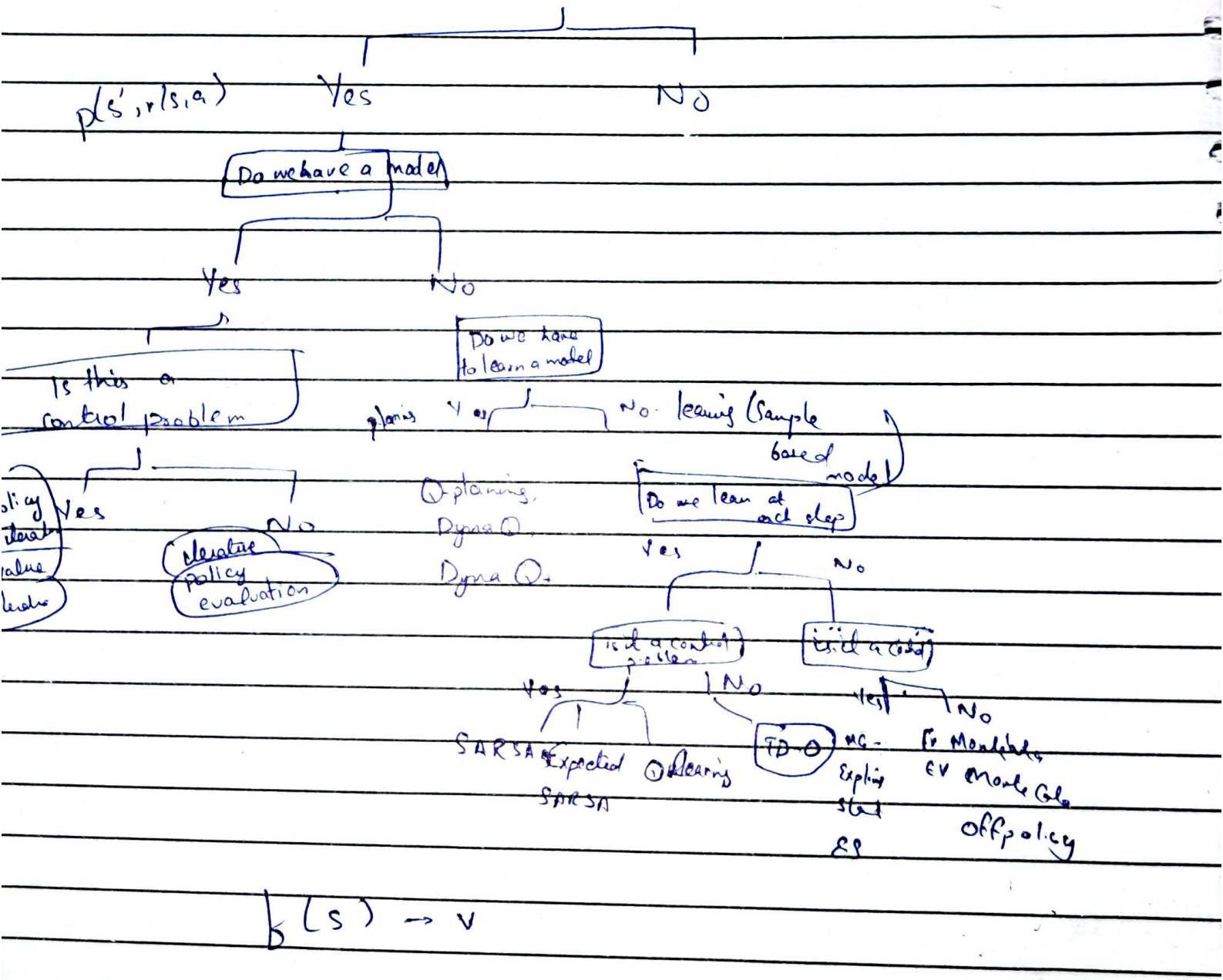
κ — hyperparameter

τ — number of steps elapsed since the action A has been taken

Date _____ M T W T F S

Castelli

Can we use tables to solve values



$f(s) \rightarrow v$

Date

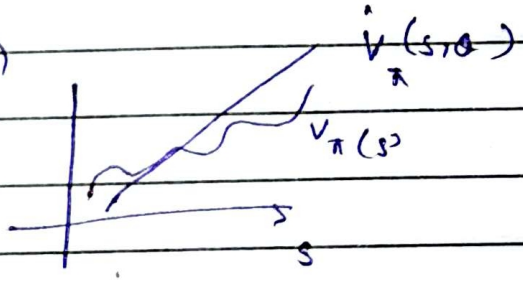
Dont know the $v(s)$

true value $V_\pi(s) = y$

approximate value $\hat{V}(s, w) = \hat{y}$

Need a function

$$J(\theta) = \frac{1}{2} \sum_s \left(V_\pi(s) - \hat{V}(s, w)\right)^2$$



In supervised
true values are given
but here we dont
know true value

when we change θ
error gets large for few state
but gets small for few state

'it change / reduce overe error

We set α to appropriate
change accordingly

$$= \frac{1}{2} \sum_s \mu(s) \left(V_\pi(s) - \hat{V}(s, w)\right)^2$$

Date _____ M T W T F S

Castelli

state which visit more

↓

more important

↓

greater $U(s)$ value

Frequently visit states →

$$\theta = \theta_i - \frac{\partial J}{\partial \theta_i}$$

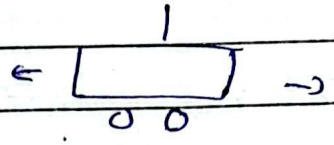
~~$V(s)$~~

Date M T W T F S

SARSA → policy iteration

Qlearning → value iteration

Cartpole



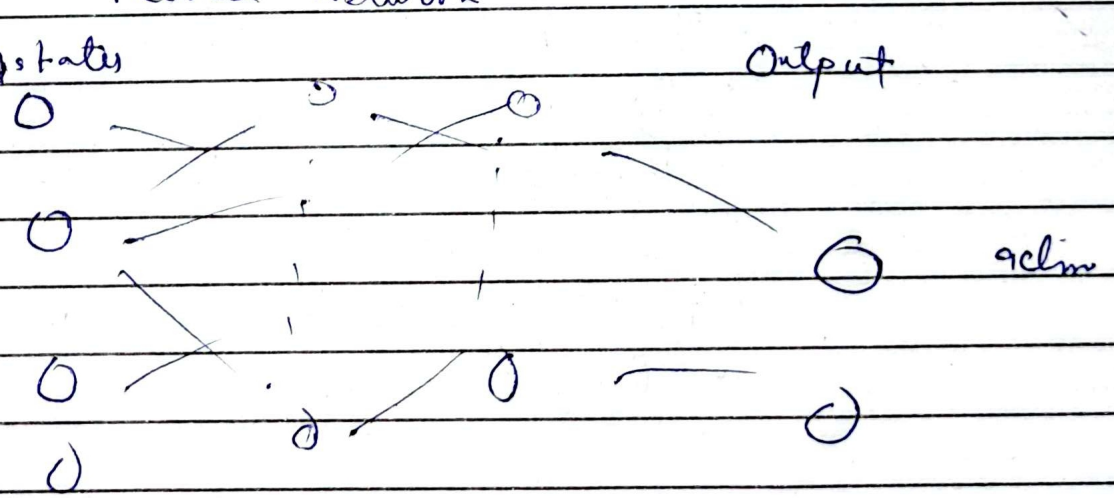
Actions 2 left, right

State → location, speed, angle, angular velocity

~~State space~~ → infinite

$q(s,a)$ → can't store in a tabular data structure

Neural Network



forward pass

we have to change weights

Date _____ M T W T F S

Castelli

Loss function $J(\theta) = \frac{1}{2} \sum (V(s) - V(s,w))^2$

need Actual and Predicted value

$$Q(S,A) \leftarrow Q(S,A) + \alpha[R + \gamma \max_a Q(S'A) - Q(S,A)]$$

predicted values

Target values

$$\text{target} = R + \gamma \max_a Q(s,a')$$

from environment

From neural network

Algorithm:

1- Initialize:

- Neural Network (Q-network to estimate $Q(s,a)$)
- Replay buffer memory)
- Hyperparameters like learning rate, epsilon, gamma

2- For each episode:

- Reset the environment to get an initial state
- For each step:

1- Choose an action:

- Use ε-greedy strategy (explore or exploit)

2- Take the action

- Observe next state and reward

Date [M|T|W|T|F|S]

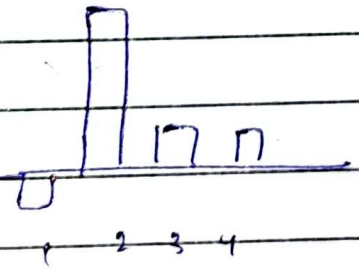
3- Store the experience.
- Save (state, action, reward, next state) to memory

4- Sample a batch from memory and:
- Compute Q-value estimates from the network.
- Compute the Q-learning target:
  target = $r_t + \gamma \max_a Q(s,a)$
- Compute loss and update network weights via backpropagation.

3- Repeat until the agent learns to balance the pole.

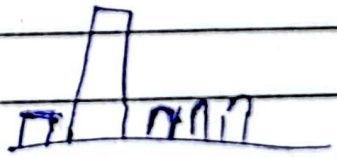
Policy Gradients

$Q(s,a)$



ε-greedy →

$\pi(a|s)$



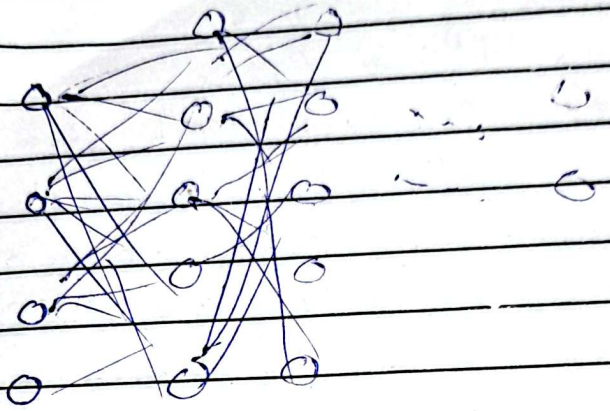
Learns not to greedy action as one

Instead of action values

→ get action probabilities

We use softmax

Date M T W T F S

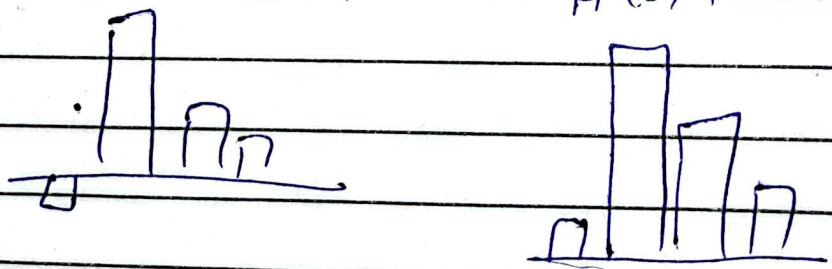


Softmax

Preference for actions
$H(s,a)$ (relative)

Actn value $Q(s,a)$ (actual values)

$Q(s,a)$ $\pi(s,a)$

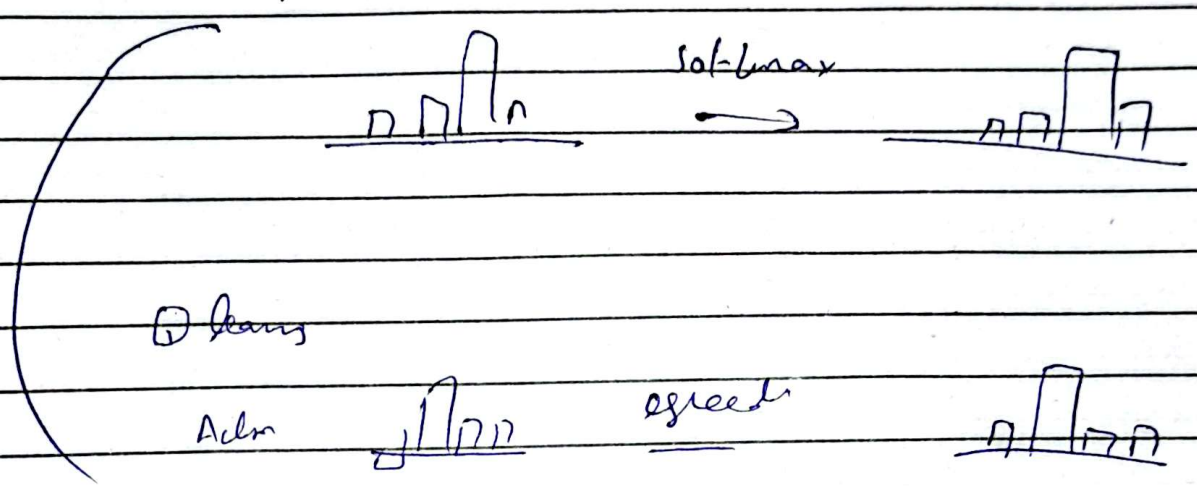


Reference not treating equally

Date _____ M T W T F S

In policy gradient we have action preferences

$\pi(a|s)$



It consider the preferences.

Policy gradient use

Gradient Ascend

You maximize the objective function

$J(\theta) = r(\pi)$

Average reward / action maximize karna hota such hota

Average Return

$\sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a) \cdot r$ $= \sum_s \mu(s) \sum_a \pi(a|s,\theta) \sum_a \pi(a|s,\theta) \cdot r$

$\mathbb{E}[R|s,a)$

$\mathbb{E}[R|s)$

Date MTWTFS Castelli

$$\nabla \bar{r} = \sum_s \mu(s) \sum_a \pi(a|s,\theta)\, q_\pi(s,a)$$

Law

$$\theta = \theta + \nabla \bar{r}$$

$$\nabla \bar{r} = \sum \ln \pi(a|s,\theta) \cdot q_\pi(s,a)$$

$\leftarrow G_t$

policy gradient using Monte Carlo

called

REINFORCE

$\leftarrow TD$

Actor

Critic