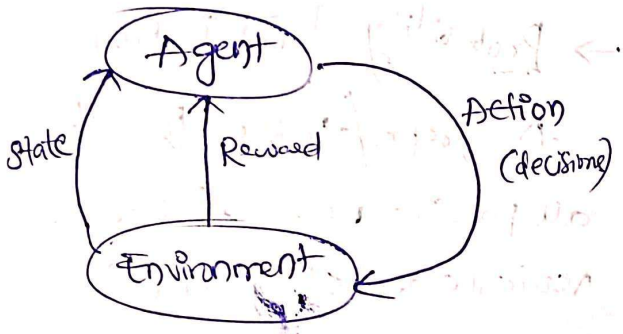
Reinforcement Learning :-

It is a type of ML, where an agent takes action based on the changes occuring in the environment & stores the reward received by performing the action in state stage. (feedback)

Using stored data it takes next step & continues for further actions.

- Here the model learns by it self.



- The Goal is to achieve the best result.
- RL is totally different from supervised & unsupervised learning.
- RL is an area of ML. It is abt taking suitable action to maximize the reward in a particular situation.

→ In RL, RL agent decides what to perform according to the situations for a given task. RL agent learns by itself from its experience.

→ Key components

1) Agent :- The learner & decision maker tries to learn the best actions to take in an environment.

2) Environment :-
This is the place where an agent will interacts with. It can be any thing like game, real world task etc.

3) Action :-
These are the moves taken by an agent in the environment.

4) State :-
State is a situation returned by the environment after each action is taken by the agent.

5) Reward :-
The feedback returned to agent from the environment for its action.

6) Policy :- The strategy or rule that the agent follows to decide the actions to take in different states.

→ There are mainly two types of RL.
1) Model based RL
2) Model free based RL.

9492 prapadil

→ In model based RL the agent tries to build the model and has access to a model of the environment.

→ Once the agent has the model, agent can plan ahead and choose actions.

→ In model free RL the agent doesn't build the model of environment. The agent learns without a model of the environment.

⇒ Basics of probability & linear algebra :- to understand RL one should have basic of

Probability Basics in RL :-

Probability :- It is the measure of the likelihood of an event occuring.

It lies blw 0 & 1.

In RL probabilities are used to represent uncertainity such as likelihood of transitioning from one state to another state. etc.

$P(E) = \frac{\text{no. of favourable outcomes}}{\text{Total no. of outcomes.}}$

E → Event

Ex:- Coin a toss.

$P(H) = \frac{1 \text{ (favourable)}}{2 \text{ (total)}}$

H → Head

Random Variable :-

Variable that take different values randomly.

In RL, RV represent states, actions, reward and other quantities that are subjected to uncertainity (no perfect information).

→ Probability Distribution :-

A graph/table that show all possible values of a random variable & their probabilities.

→ Expected Value :-

The avg value of a RV.

Probability distribution Ex:-

Coin a Toss.

$P(H) = 1/2$

$P(T) = 1/2$ (∵ All values are shown)

→ Conditional Probability :-

The prob. of an event occuring given that another event has occured.

$P(B/A) = \frac{P(A \cap B)}{P(A)}$.

Linear Algebra in RL :-

LA is the branch of mathematics concerning linear Equation such as

$$a_1x_1 + a_2x_2 + \cdots + a_nx_n = b$$

- Linear Algebra helps in managing & processing data when the environment is complex & when we have lot of information i.e (large state space & actions.
- In RL LA is used to represent the rel-ship blw variables to make it easier to analyse & minimize the learning process.

Linear algebra includes :-

Vectors :- These are mathematical objs that represent both magnitude & direction.

In RL things like states, actions & rewards can be represented as vectors.

Matrices :-

These are used to represent transition Probabilities, rewards & value function.

Matrix operations :-

Such as Add, sub, mul, & inversion are used in RL to update value functions, calculate state-action values etc

→ Eigen values & Vectors :

These are used in algorithms like PCA (principal component analysis) to extract important features from states.

→ Both probability & linear algebra are important for RL bcoz probability helps us model the uncertainity in RL environment whereas LA helps us making decisions by representing value fun. & policy.

→ Even Both help us optimize the performance of RL agents.

⟹ Definition of Stochastic Multi armed bandit :-

A stochastic multi armed bandit (MAB) is a classic problem in RL that involves making decisions in an uncertain environment.

MAB is a simplified decision making problem where we have multiple slot machines each with different unknown reward probabilities.

Arm refers to an option or choice that an agent can select.

→ In simple SMAB is choosing the best machine with the given multiple slot machines (arms).

→ The aim of SMAB is to find the optimal arm to maximize the cumulative reward (total reward) or sum of all the rewards.

→ Imagine you are in a casino with several slots machines (arms). Each machine has a different prob. of paying out money when you will pull the lever.

Initially we don't know the probabilities of machines in advance.

→ Our goal is to figure out which machine is the best and gives higher reward by pulling the lever.

→ Working:-

Initially there are multiple machines each provides different rewards. We need to find the best one by Exploration and Exploitation.

→ Exploration defines trying different machines to learn which is best.

→ Exploitation focus on the machine that already known to give the best rewards.

⇒ Definition of Regret :-

In RL regret is a measure of how much worse (worst) your agent's performance is compared to the best possible performance it could have achieved.

Regret is the difference b/w the optimal reward an agent could have received and the actual reward it received.

Regret = optimal Reward − Actual Reward

- Regret is a crucial concept in RL bcoz it helps agents evaluate their decisions and learns from their mistakes.

By minimizing regret, agent can improve their performance & make better decisions over time.

Ex:-

Suppose an agent is playing a game where it receives

rewards for completing levels. The reward for completing 1 level is 100 points. However, the agent only receives 80 points.

The regret would be

Regret = Optimal Reward − Actual Reward

100 − 80

$R = 20.$

Here agent missed out 20 points.

→ 2 Types of regret

1) Internal 2) External

It measures the difference blw agent's actual reward & the reward it would have received, is called External regret if it followed best fixed action.

Internal regret :-

It measures the difference blw the agent actual reward & the reward it would have received if it had followed the best sequence of action

⇒ Achieving Sublinear regret (ASR) :-

ASR in RL refers that the agent interact with the environment over time, the amount of regret it experiences grows more slowly than the total no. of actions it takes.

Sublinear regret means the agents regret mistakes decreases over time.

→ ASR is important bcoz it ensures that the agent learn efficiently & make good decisions most of the time.

→ ASR minimizes the difference blw total reward obtained by an agent & the total reward of an optimal strategy.

→ Strategies / Types of ASR :-

1) Explore - Exploit trade off / Strategy

The agent needs to balance trying out new actions by exploring and sticking with the actions that have already worked well by exploiting.

2) Upper Confidence Bound (UCB) :-

Assigning confidence bounds to each actions estimated value.

Ex: consider 2 actions, A & B, If agent tried many time action A, & B tried only few times then UCB choose action B to reduce uncertainity.

3) Thompson Sampling :-

This strategy uses probability distribution to choose actions.

It aims to maximize rewards by balancing exploration & exploitation.

Ex: If action A is mostly likely to give the highest reward, the agent will pick action A, but may still explore actions based on their probabilities.

4) Optimism in the face of uncertainty :-

using optimistic initial values to encourage exploration. i,e be optimistic abt unknown actions & try them.

In RL, the idea of optimism is often used when agent isn't sure with which actions will lead to the best reward.

→ ASR achieves good decision making & improved performance.

⇒ UCB Algorithm :-

upper confidence bound.

- It's an algo. used in RL to balance exploration & exploitation.
- It is a way to make decisions in situations where you don't know the best option
- Ex: choosing best machine among the multiple machines.

UCB working

UCB focuses on maximizing reward over time by considering both how good an option seems and how much you need to explore that option.

1) Initialize the estimated reward and confidence bounds for each action.
2) Choose the actions with the highest UCB.
3) Update the estimated reward & CB based on the observed reward. & repeat step 2, 3

→ UCB gives higher priority to options that have been tried less

Types of UCB

1) UCB1 (classic UCB) :-

This is the basic version of UCB. It is the original UCB algo. that balances explore & exploitation using UCB formula.

formula

$$UCB(A) = \text{avg reward of actions } A + C \times \sqrt{\frac{\ln(t)}{N(a)}}$$

where

UCB(a) = Upper confidence bound for action 'a'.

Avg reward ← taking actions.

C = Parameter controlling b/w explore & exploit.

ln(t) = natural algorithm.

N(a) = no. of times action has been choosen

2) UCB Tuned :-

This is a refined version of UCB1.

It adjusts the constant factor in the confidence bound to handle reward distribution for action.

It outperforms UCB1 in many environments.

3) KL-UCB

Kullback-Leibler UCB.

It focuses on minimizing the KL divergence b/w the estimated & true reward distribution.

4) MOSS : (Minimax optimistic sampling strategy) :-

It is the extension of UCB.

It provide more conservative approach by minimizing.

⟹ KL-UCB :-

- It is the another version of the UCB algo, but it uses more refined way to balance exploration & exploitation

→ The key difference with regular UCB is that KL-UCB uses a more probabilistic approach to measure uncertainty.

→ Instead of just considering how many times an action has been selected, it looks at how far the current distribution of rewards for each action is from what you expected and adjust the exploration accordingly.

→ KL-UCB uses KL-divergence to calculate the UCB for each action.

→ It helps the agent make decisions by considering both estimated value of an action & uncertainity of information

→ KL-UCB tries to make smarter decision using a more mathematical & detailed understanding of the uncertainity in the reward.

Working :-

1) The agent start with an initial estimate of the reward distribution for each action

2) The agent calculates the KL divergence blw estimated reward distribution and a uniform distribution

3) The agent uses KL divergence to calculate a confidence bound for each action.

4) The agent chooses the action with highest UCB.

5) The agent updates its estimate of the reward for the choosen action based on the observed reward.

6) The agent repeats steps 2-5.

→ KL-divergence measures the difference blw 2 probability distributions.

→ confidence bound estimates the uncertainity of the reward distribution

→ UCB is the maximum possible reward for an action

KL-UCB Formula

$$UCB(a) = Q(a) + \sqrt{\frac{2\log(t)}{N(a)}}$$

$KL(P,q)$

where

$UCB(a)$ = UCB of an action a.

$Q(a)$ = Estimated reward for action a.

P is estimated reward distribution

q is uniform distribution.

$KL(P,q)$ is KL divergence blw P & q

t is current time step.

$N(a)$ is no. of times an action has been selected / choosen.

Expt. No..... Sheet No..... Date .....

Thompson Sampling :-

It is a method used in RL to help an agent make decisions based on uncertainty.

- It is an algo. that helps agent to choose actions in an uncertain environment.
- It involves in maintaining a probability distribution over possible actions to decide next action.

→ There are 2 methods.

1) Bernoulli Thompson Sampling
2) Gaussian " "

→ BTS is used for binary outcomes (Sucess/ failure)

Ex: Coin a Toss. if Head sucess else fail

→ GTS is used for continuous outcomes.

Working :-

1) Initialize a probability distribution over the possible rewards for each action.
2) Sample a reward from the probability distribution for each action.
3) Chose the action with highest Sampled reward
4) Update the PD based on observed reward.

Formula :-

$$\alpha(a) \leftarrow \alpha(a) + \gamma$$

$$\beta(a) \leftarrow \beta(a) + 1 - \gamma$$

$\alpha(a)$ & $\beta(a)$ are the shape parameters of the β distribution for action (a).

$\gamma \Rightarrow$ observed reward.