Redesigning Common Mind & Business Towards Excellence

3P Purpose Process People Culture

SNS INSTITUTIONS
www.snsgroups.com

1st GenAI Powered Design Thinking FrameWork

Build an Entrepreneurial Mindset Through Our Design Thinking FrameWork

SNS COLLEGE OF TECHNOLOGY

Coimbatore-35
An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A++' Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

23AMB201 - MACHINE LEARNING

II YEAR IV SEM

UNIT V – REINFORCEMENT LEARNING

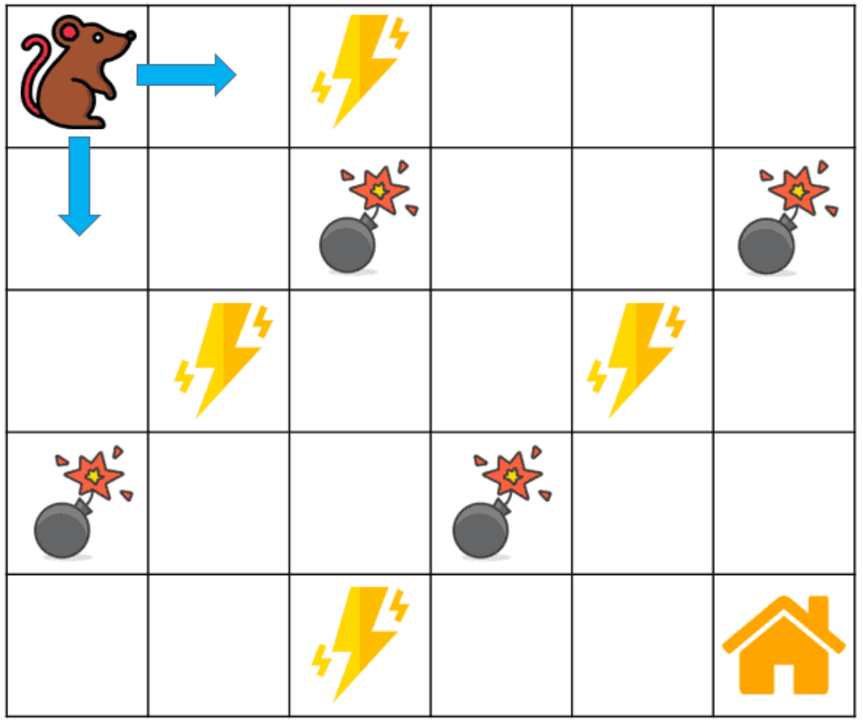
TOPIC 6 – Q Learning Algorithm

Q-Learning in Reinforcement Learning

- Q-learning is a **model-free reinforcement learning algorithm** used to train agents (computer programs) to make optimal decisions by interacting with an environment.
- It helps the agent explore different actions and learn which ones lead to better outcomes.
- The agent uses trial and error to determine which actions result in rewards (**good out**
- **comes)** or **penalties (bad outcomes**).
- It works by assigning a "Q-value" to each action the agent can take in a given state, representing the expected reward from taking that action

Example

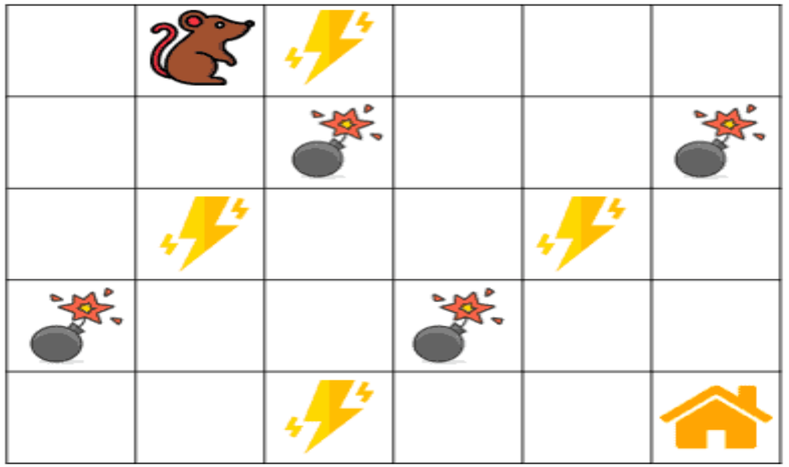
Our agent is a rat that has to cross a maze and reach the end (its house) point.
There are mines in the maze, and the rat can only move one tile (from one square to one another) at a time
If the rat steps onto a mine, it will be dead. The rat wants to reach its home in the shortest time possible:



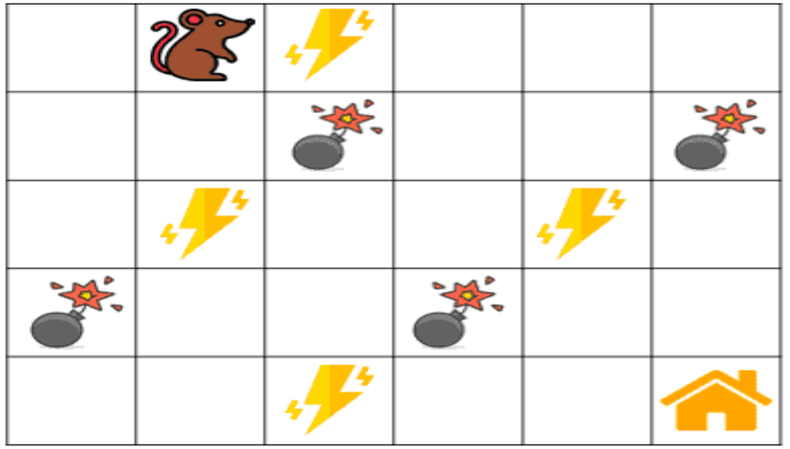
Example

rat should take the shortest path and reach its home as fast as possible
If the rat steps on a mine, it loses 100 points, and the game is over
If the rat gets power (the yellow lightning icon), it gains point
If the rat reaches its home, it is rewarded 100 points
The rat has four possible actions (move up, down, right, or left)
we can initialize the Q-table as a matrix of size 54, where the rows represent the rat’s possible states (positions), and the columns represent the possible actions (moving in 4 directions)

Example



STATES	ACTIONS			
	↑	↓	→	←
Start	0	0	0	0
Blank	0	0	0	0
Power	0	0	0	0
Mines	0	0	0	0
Home	0	0	0	0



STATES	ACTIONS			
	↑	↓	→	←
Start	0	0	0	0
Blank	0	-1	1	-1
Power	0	0	0	0
Mines	0	0	0	0
Home	0	0	0	0

Applications of Q-learning

Key Components of Q-learning

1. Q-Values or Action-Values

Q-values represent the expected rewards for taking an action in a specific state. These values are updated over time using the Temporal Difference (TD) update rule.

2. Rewards and Episodes

The agent moves through different states by taking actions and receiving rewards. The process continues until the agent reaches a terminal state, which ends the episode.

Key Components of Q-learning

3. Temporal Difference or TD-Update

The agent updates Q-values using the formula:

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

Where,

S is the current state.

A is the action taken by the agent.

S' is the next state the agent moves to.

A' is the best next action in state **S'**.

R is the reward received for taking action **A** in state **S**.

γ (Gamma) is the **discount factor**, which balances immediate rewards with future rewards.

α (Alpha) is the **learning rate**, determining how much new information affects the old Q-values.

Key Components of Q-learning

4. ϵ-greedy Policy (Exploration vs. Exploitation)
The ϵ-greedy policy helps the agent decide which action to take based on the current Q-value estimates:
Exploitation: The agent picks the action with the highest Q-value with probability $1–\epsilon$. This means the agent uses its current knowledge to maximize rewards.
Exploration: With probability $\epsilon\epsilon$, the agent picks a random action, exploring new possibilities to learn if there are better ways to get rewards. This allows the agent to discover new strategies and improve its decision-making over time.

How does Q-Learning Works?

Agent: The entity that makes decisions and takes actions within the environment.
States: The variables that define the agent's current position in the environment.
Actions: The operations the agent performs when in a specific state.
Rewards: The feedback the agent receives after taking an action.
Episodes: A sequence of actions that ends when the agent reaches a terminal state.
Q-values: The estimated rewards for each state-action pair.

Steps of Q-learning

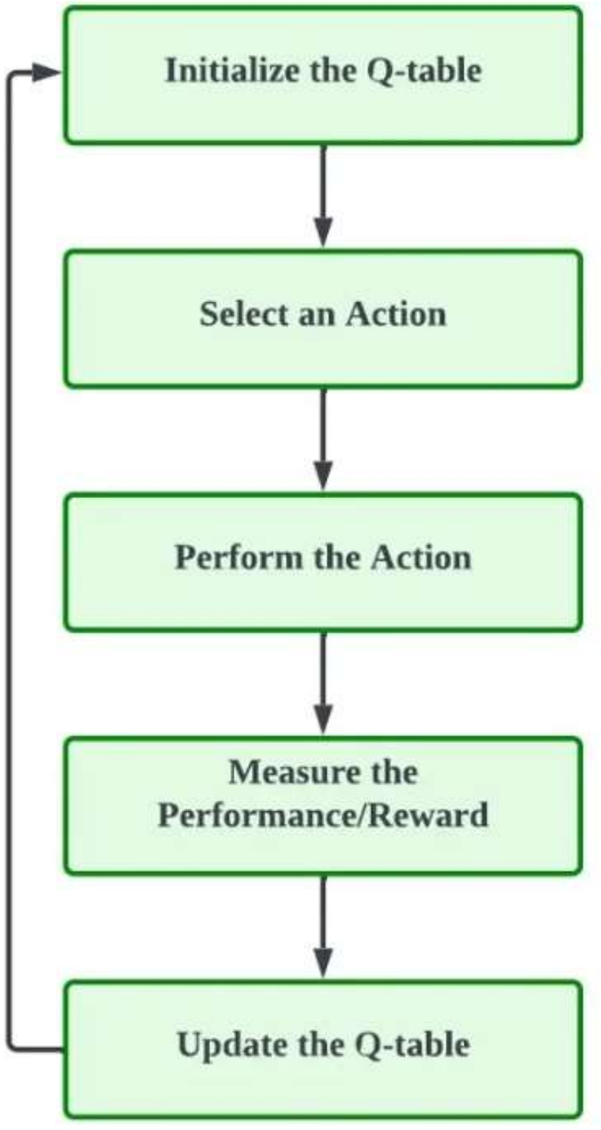
Initialization: The agent starts with an initial Q-table, where Q-values are typically initialized to zero.

Exploration: The agent chooses an action based on the ϵ-greedy policy (either exploring or exploiting).

Action and Update: The agent takes the action, observes the next state, and receives a reward. The Q-value for the state-action pair is updated using the TD update rule.

Iteration: The process repeats for multiple episodes until the agent learns the optimal policy.

The higher the iterations number, better is the Q-table results



Methods for Determining Q-values

Temporal Difference (TD):
Temporal Difference is calculated by comparing the current state and action values with the previous ones. It provides a way to learn directly from experience, without needing a model of the environment.
2. Bellman's Equation:
Bellman's Equation used to calculate the value of a given state and determine the optimal action.

What is a Q-table?

The **Q-table** is essentially a **memory structure** where the agent stores information about which actions is the best rewards in each state.
It is a table of Q-values representing the agent's understanding of the environment
Structure of a Q-table:
Rows represent the states.
Columns represent the possible actions.
Each entry in the table corresponds to the Q-value for a state-action pair.