1 Q-Learning Explanation

1.1 Q-Learning Overview

Objective: The goal of Q-learning is to find the optimal policy (i.e., a mapping from states to actions) that maximizes the cumulative reward over time.

1.2 Mathematical Foundation

1. **Q-Function (Action-Value Function):** The Q-function $Q(s, a)$ represents the expected cumulative reward of taking action a in state s and following the optimal policy thereafter.

2. **Bellman Equation:** The Q-value is updated based on the Bellman equation, which provides a recursive decomposition of the Q-value:

$$Q(s, a) = \mathbb{E}\left[r + \gamma \max_{a'} Q(s', a') \mid s, a\right]$$

   Where:

   - s: current state
   - a: current action
   - r: reward received after taking action a in state s
   - s': next state
   - a': next action
   - γ: discount factor (how much future rewards are valued compared to immediate rewards)

1.3 Q-Learning Algorithm

1. **Initialization:**

   - Initialize the Q-table with zeros (or small random values), where rows represent states and columns represent actions.
   - Set hyperparameters: learning rate (α), discount factor (γ), exploration rate (ϵ).

2. **Action Selection (Exploration-Exploitation Tradeoff):**

   - Use an epsilon-greedy policy to select actions:
     - With probability ϵ, choose a random action (exploration).
     - With probability $1-\epsilon$, choose the action with the highest Q-value for the current state (exploitation).

3. **Q-Value Update:**

   - Execute the selected action, observe the reward and next state.

- Update the Q-value using the Bellman equation:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left(r + \gamma \max_{a'} Q(s',a') - Q(s,a)\right)$$

  Here, α is the learning rate, and γ is the discount factor.

4. **Update Epsilon:**
   - Gradually decay the exploration rate ϵ to reduce exploration over time.

1.4 Detailed Steps

1. **Initialization:**
   - **q_table:** A 2D array where each cell $Q(s,a)$ is initialized to zero. The table's dimensions are state_size x action_size.
   - α: Learning rate determining how much new information overrides old information.
   - γ: Discount factor determining the importance of future rewards.
   - ϵ: Initial exploration rate to ensure the agent explores the environment.
2. **Action Selection:**
   - The act method decides whether to explore or exploit using the epsilon-greedy strategy.
   - If a random number is less than ϵ, a random action is chosen (exploration).
   - Otherwise, the action with the highest Q-value for the current state is chosen (exploitation).
3. **Q-Value Update:**
   - In update_q_value, the agent updates the Q-table using the Bellman equation.
   - **best_next_action** identifies the best action to take from the next state.
   - **td_target** is the target Q-value considering the immediate reward and the discounted future reward.
   - **td_error** is the difference between the target Q-value and the current Q-value.
   - The Q-value is updated towards the td_target based on the learning rate α.
4. **Epsilon Decay:**
   - The exploration rate ϵ is decayed after each update to gradually shift from exploration to exploitation.

1.5 Explanation of Calculations

- **Exploration vs. Exploitation:** Ensures the agent explores enough initially to gather knowledge about the environment and gradually exploits the learned policy as ϵ decays.
- **Q-Value Update:** Uses the Bellman equation to iteratively improve the estimate of the optimal Q-values.
- **Discount Factor (γ):** Balances the importance of immediate rewards vs. future rewards.
- **Learning Rate (α):** Controls how quickly the Q-values are updated based on new information.

By following these principles, the Q-learning algorithm enables the agent to learn the optimal policy through interaction with the environment, progressively improving its decision-making to maximize cumulative rewards.