Introduction

January 17, 2025

[ ]:
```
'''
    Reinforcement Learning -->

    Reinforcement Learning (RL) is a type of machine learning where
    an agent learns how to behave in an environment by performing actions
    and receiving rewards or penalties. The goal of the agent is to learn
    a policy that maximizes the cumulative reward over time.

    Key Components of Reinforcement Learning -->

    Agent: The learner or decision-maker.
    Environment: Everything the agent interacts with.
    State (S): A representation of the current situation of the agent in the␣
 ↪environment.
    Action (A): A set of all possible moves the agent can make.
    Reward (R): Feedback from the environment to evaluate an action.
    Policy ( ): A strategy that the agent uses to determine actions based on␣
 ↪the current state.
    Value Function (V(s)): Predicts the long-term reward of a state under a␣
 ↪policy.
    Q-Function (Q(s,a)): Predicts the long-term reward of taking action a in␣
 ↪state
    s and following the policy thereafter.
'''
```

[ ]:
```
'''
    Workflow of Reinforcement Learning -->

    The agent observes the current state of the environment.
    Based on the policy, it takes an action.
    The environment transitions to a new state and provides a reward.
    The agent updates its policy based on the reward and the observed␣
 ↪transition.
    This loop continues until the agent learns an optimal policy.
'''
```

[ ]:
```
'''
   In the context of Reinforcement Learning, Upper Confidence Bound (UCB) and
   Thompson Sampling (TS) are commonly used for tackling the multi-armed bandit
   (MAB) problem. These methods aim to balance exploration (trying out␣
↪less-tested
   actions) and exploitation (choosing actions that seem to perform best based
   on current information).

   Multi-Armed Bandit Problem -->

   You are given multiple "arms" (choices/actions), each with an unknown
   probability distribution of rewards.
   The goal is to pull the arms in such a way that maximizes the cumulative
   reward over time.

   Upper Confidence Bound (UCB) -->

   UCB is a deterministic approach to address the exploration-exploitation␣
↪trade-off.
   It assumes that the true mean reward of an arm is within a certain␣
↪confidence
   interval based on the number of times the arm has been pulled.

   Workflow:
   At each time step, calculate the UCB for all arms.
   Select the arm with the highest UCB.
   Update the rewards and the pull count for the selected arm.

   Advantages:
   Guarantees logarithmic regret.
   Effective when the distribution of rewards has clear separation.

   Limitations:
   Works best with stationary reward distributions.
   Computationally expensive for a large number of arms due to repeated␣
↪calculations.

   Thompson Sampling -->

   Thompson Sampling is a probabilistic approach based on Bayesian inference.
   It assumes a prior distribution for each arm's reward probabilities and
   updates this posterior based on observed rewards.

   Workflow:
   Initialize a prior distribution for each arm
   (e.g., Beta distribution for binary rewards).
   For each arm, sample from its posterior distribution.
```

```
    Select the arm with the highest sampled value.
    Observe the reward and update the posterior distribution for the selected␣
 ↪arm.

    Advantages:
    Naturally balances exploration and exploitation through random sampling.
    Handles non-stationary rewards better than UCB.

    Limitations:
    Computationally more intensive for complex reward distributions.
    Requires well-defined priors for reward distributions.
'''
```

[ ]:
```
'''
    Feature                              UCB                                   ␣
 ↪      Thompson Sampling

    Approach                          Deterministic                            ␣
 ↪ Probabilistic
    Exploration-Exploitation        Explicit control via confidence␣
 ↪term        Implicit through sampling
    Computational Complexity        Moderate                                 ␣
 ↪High (due to sampling)
    Assumptions                          Works well with stationary␣
 ↪rewards            Handles non-stationary rewards
    Ease of Implementation            Simple                                 ␣
 ↪    Slightly complex
'''
```

[ ]:
```
'''
    Applications -->

    Both methods are widely used in :

    Recommendation Systems (e.g., which ad to show a user).
    Clinical Trials (e.g., testing multiple treatments).
    Online Platforms (e.g., optimizing website layouts).
'''
```