Unit 5: Reinforcement Learning in AI - Theory Answers

Q1: Explain Passive Reinforcement Learning. How is it different from Active Reinforcement Learning?

Passive Reinforcement Learning is a learning setting where the **policy is fixed**. The agent's goal is not to find a new policy but to **evaluate** the existing one. The agent learns the utilities of different states by observing the environment or replaying historical data.

- The agent executes the given policy and estimates the **utility** values (U(s)) from observed rewards.
- No exploration is required.
- Example: A robot watching another robot navigate a maze.

Active Reinforcement Learning, in contrast, involves learning the **optimal policy** through **trial and error**. The agent explores the environment, takes actions, receives feedback, and adjusts its behavior to maximize cumulative rewards.

- Involves exploration and exploitation.
- Example: A robot playing table tennis and learning through practice.

(Refer to slides 18–21.)

Q2: What is Direct Utility Estimation in reinforcement learning? Explain the steps involved with an example.

Direct Utility Estimation is a method in passive reinforcement learning where the utility of a state is estimated directly from multiple episodes or trials, without learning the environment’s transition model.

Steps Involved:

1. Execute multiple episodes following a fixed policy.
2. Observe sequences of states and rewards.
3. For each state, calculate the **average total reward** received from that state onwards.
4. Update the utility U(s) using the average of observed returns.

Example: In healthcare, the utility of treatments can be directly estimated from patient histories without modeling health transitions. If treatment A leads to better outcomes in 80% of cases, its utility is estimated from those outcomes.

(Refer to slides 22–23.)

Q3: What is Adaptive Dynamic Programming in reinforcement learning? How does it improve over passive learning?

Adaptive Dynamic Programming (ADP) is a reinforcement learning method where the agent **learns the environment's model** and updates the utilities accordingly.

Improvements over Passive Learning:

- In ADP, the agent estimates the transition model P(s'|s, a) and reward function R(s), then uses this model to solve the **Bellman equations** to improve the utility estimates.
- ADP allows **planning**: the agent can simulate different strategies to find better policies.

Example: An autonomous drone learns to navigate a complex environment by learning which movements lead to which outcomes, improving decision-making over time.

(Refer to slides 24–25.)

Q4: What is Temporal Difference Learning? How does it combine ideas from Monte Carlo methods and Dynamic Programming?

Temporal Difference (TD) Learning is a reinforcement learning method that combines **Monte Carlo methods** and **Dynamic Programming (DP)** principles.

Key Features:

- Updates utility based on **difference between successive states** (bootstrapping).
- Does not require the transition model (like MC).
- Updates U(s) using U(s') from the next state (like DP).

Formula: $U(s) \leftarrow U(s) + \alpha [R(s) + \gamma U(s') - U(s)]$

Example: In chess, a program updates its evaluation of a position based on the evaluation of the next position encountered during the game.

(Refer to slides 26–27.)

Q5: What is Q-Learning in Active Reinforcement Learning? Explain the Q-value update formula with an example.

Q-Learning is a model-free, off-policy reinforcement learning algorithm used in **Active RL**. It learns the value of taking a specific action in a given state (Q(s, a)) without needing the environment’s model.

Q-Value Update Formula: $Q(s, a) \leftarrow Q(s, a) + \alpha [R + \gamma \max Q(s', a') - Q(s, a)]$

Where:

- α = learning rate
- γ = discount factor
- R = reward
- s' = next state

- max Q(s', a') = maximum Q value of next state

Example: A robot in a maze updates Q-values based on paths that lead to goals with higher rewards. Over time, it prefers actions that lead to higher cumulative rewards.

(Refer to slide 29.)

Q6: Differentiate between Supervised and Semi-supervised Learning

Supervised Learning:

- Uses **labeled data**.
- Learns mapping from input (X) to output (Y).
- Requires large labeled datasets.
- Accurate but expensive.

Semi-Supervised Learning:

- Uses **small labeled + large unlabeled data**.
- Aims to reduce labeling costs.
- Less accurate than supervised but more efficient.
- Balances between manual labeling and unsupervised methods.

Example: Classifying documents where only a few are labeled and the rest are inferred using the labeled data.

(Refer to slides 27–32 from Unit 4.)

Q7: Explain capabilities of expert systems.

Expert Systems provide intelligent behavior similar to human experts in specific domains. They exhibit various capabilities that make them powerful AI tools:

Key Capabilities:

1. Advising: Offers expert-level advice and recommendations.

2. Decision Making: Helps make complex decisions, such as medical or financial decisions.

3. Demonstrating Devices: Explains the working and features of new tools or software.

4. Problem Solving: Identifies and resolves domain-specific problems.

5. Explaining Problems: Provides clear reasoning and explanations for its conclusions.

6. Input Interpretation: Understands user queries.

7. Result Prediction: Predicts outcomes based on existing data.

8. Diagnosis: Used extensively in medical diagnosis without human intervention.

Examples:

- Medical Expert Systems like MYCIN.
- Financial advisors that detect fraud.
- Troubleshooting tools for electronics.