CSCI 397: Q-learning for Frozen Lake

Due Date: Monday, November 4th @ 11:59 PM

Information:

Implement the Q-learning algorithm to find the optimal policy for navigating the FrozenLake environment in OpenAI's Gym. The FrozenLake environment models a grid world where an agent must navigate from a starting position to a goal position while avoiding holes. The environment is both slippery and stochastic, meaning that the agent does not always move in the direction it intends to.

Tasks:

Environment Exploration:

Instantiate the FrozenLake environment with is_slippery=True.

Explore the environment by taking random actions and observe the rewards and next states.

Understand the reward structure and the total number of states and actions.

Implement Q-learning:

Initialize a Q-table with zeros.

Set parameters: learning rate (α), discount factor (γ), and exploration rate (ε).

For a given number of episodes:

- Interact with the environment using an ε-greedy policy derived from the current Q-table.
- Update the Q-values using the Q-learning update rule.

Note: Due to the stochastic nature of the environment, ensure you handle the environment's transition probabilities and rewards correctly.

Extract Policy:

For each state, choose the action with the maximum Q-value.

Evaluation:

Use the extracted policy to evaluate its performance over multiple episodes in the environment.

Compute the average reward and report the percentage of episodes in which the agent successfully reached the goal.

Experiment:

Compare the performance of the agent when is_slippery=True vs. is_slippery=False.

How does the stochasticity of the environment affect the Q-learning process and the resulting policy?

Turn in:

Python code implementing Q-learning for FrozenLake.

A brief report discussing your findings, including:

The final Q-table.

The extracted policy in a readable format.

Evaluation results.

The number of episodes taken for the Q-values to stabilize.