MSDS 684 – Reinforcement Learning

Lab 1 Report – Multi-Armed Bandits & MDP Foundations

Section 1: Project Overview

This lab explores the exploration–exploitation problem through the multi-armed bandit setting and connects it to the early reinforcement learning ideas discussed in Sutton & Barto. The bandit problem is the simplest RL setup because there is no state transition—only actions and immediate rewards. Even though it is simple, bandits highlight one of the most important RL challenges: deciding when to explore new actions and when to exploit what is already known. Sutton & Barto explain how this balance directly affects learning performance, especially in environments with uncertainty.

In this lab, I worked with a custom 10-armed Gaussian bandit environment. Each arm has a hidden true reward value sampled from a normal distribution. The agent never sees these true values, but it must estimate them through experience. The state space in the bandit environment is trivial: it does not change and is represented by a constant dummy value. The action space is discrete with 10 available actions, each corresponding to one arm. Rewards come from a normal distribution centered around each arm's true value. Episodes do not terminate because the bandit is a continuing task.

I implemented two common exploration strategies: ε-greedy and Upper Confidence Bound (UCB). ε-greedy mixes exploitation with random exploration by selecting a random action with probability ε. UCB uses the idea of "optimism under uncertainty," giving an exploration bonus to arms with fewer samples. Both methods come from Chapter 2 of Sutton & Barto, and both show different ways to handle uncertainty about action values.

Before running experiments, I expected ε-greedy with a small ε to eventually learn the best arm but at a slower rate because random exploration is not directed. I expected UCB to perform better because it prioritizes actions with uncertain estimates, especially in the early steps. The theory predicts that UCB should identify the optimal arm faster and spend less time on obviously poor actions once uncertainty has decreased. I also expected higher ε values to lead to more exploration but slower long-term convergence. For UCB, I expected larger confidence values to produce more exploration early on, possibly improving discovery of the best arm but also causing temporary regret.

Overall, this lab focuses on understanding how simple exploration strategies behave, why they work, and how their behavior connects to the basic reinforcement learning ideas explained in the textbook.

Section 2: Deliverables

GitHub Repository

Lab1 link -
https://github.com/nikhsnona/MSDS_684_ReinforcementLearning/tree/main/Lab1

clone link - https://github.com/nikhsnona/MSDS_684_ReinforcementLearning.git

Implementation Summary

I implemented a custom Gymnasium-style 10-armed Gaussian bandit and two agent classes: ε-greedy and UCB. Each agent keeps estimated action values and updates them using incremental means. I ran 1,000 independent runs, each with 2,000 time steps. For ε-greedy, I tested ε = 0.01, 0.1, and 0.2. For UCB, I tested c = 1.0, 2.0, and 5.0. For each setting, I recorded average reward and the percentage of selecting the optimal arm at every step. In Part 2, I inspected two Gymnasium environments—FrozenLake-v1 and Taxi-v3—by printing their observation and action spaces and running a random policy agent for 200 episodes. All code, experiments, and plots are stored in GitHub.

Key Results & Analysis

Across all ε-greedy runs, the agent gradually improved its action estimates, but the amount of improvement depended on ε. The smallest value, ε=0.01, had the slowest early exploration but reached fairly stable performance in later steps because it spent more time exploiting once it had good estimates. ε=0.1 gave a more balanced exploration pattern and reached the optimal arm more consistently during the middle portion of training. ε=0.2 explored the most but also took longer to settle on good estimates because it frequently chose suboptimal actions. These results match the theory that too much exploration slows convergence, while too little exploration delays initial learning.

For UCB, the performance difference was much clearer. All UCB versions quickly identified the best arm and achieved higher average reward earlier in training compared to ε-greedy. The c=1.0 version explored steadily without being excessive, which allowed it to lock onto the best arm early. The higher settings, c=2.0 and c=5.0, explored even more aggressively in the beginning. This helped them test a wide range of actions but slightly slowed their early reward accumulation. As uncertainty decreased, all UCB versions consistently selected the optimal arm for the rest of training. These results follow Sutton & Barto's explanation that UCB reduces unnecessary exploration after uncertainty lowers, making it more efficient than random exploration strategies.

The optimal-action plots reflected the same pattern: UCB reached a high optimal-action percentage sooner than all ε-greedy settings. ε-greedy improved gradually but always stayed below UCB because it continues to explore randomly even after learning the correct action values. UCB, in contrast, cuts down exploration when confidence is high, which explains why its curve flattens near 100% optimal selection.

In the Gymnasium environments, FrozenLake-v1 and Taxi-v3 showed how MDPs differ from bandits. FrozenLake has 16 discrete states and 4 actions; Taxi-v3 has 500 states and 6 actions. The random agent performed poorly in both environments, which showed the importance of informed decision-making. FrozenLake's average reward was close to zero because the agent rarely reached the goal. Taxi-v3's negative rewards showed the penalties for invalid pickups and long paths. These results highlight how much more complex full MDPs are compared to bandits.

Figures

Figure 1 — Average Reward for ε-Greedy:
The ε-greedy curves show slower early improvement compared to UCB. Higher ε values create more early exploration but lower long-term stability. The small gap between ε settings matches the theory that random exploration is not directed.

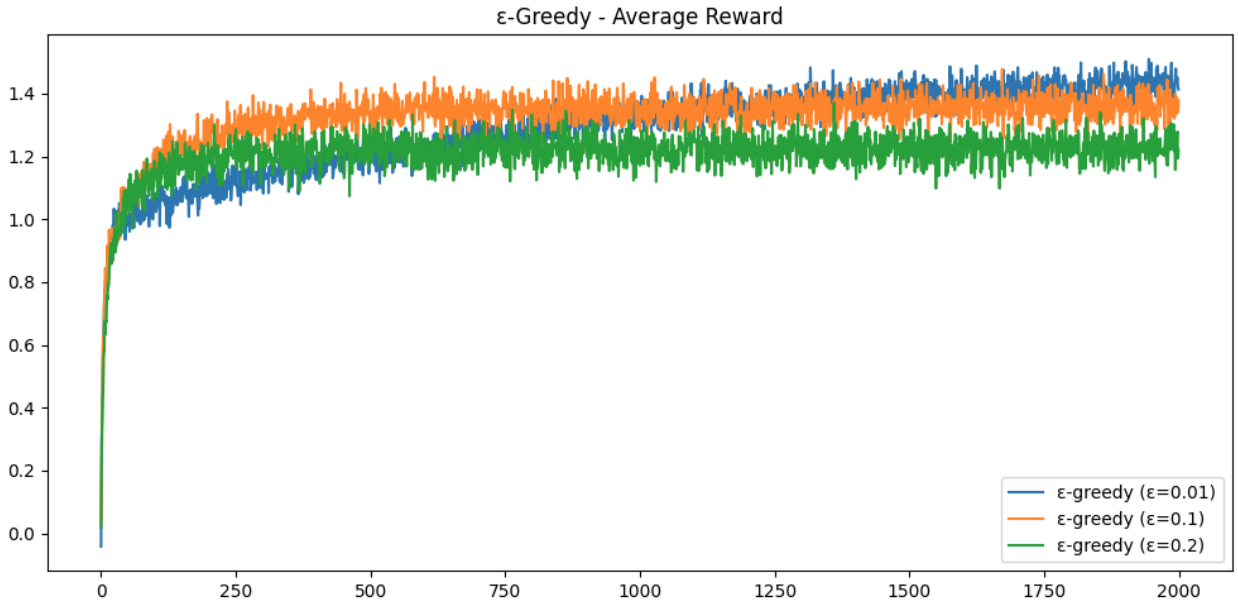


Figure 2 — Optimal Action % for ε-Greedy:
All ε-greedy settings eventually rise, but none reach stable dominance like UCB. The constant randomness prevents perfect optimal-action selection.

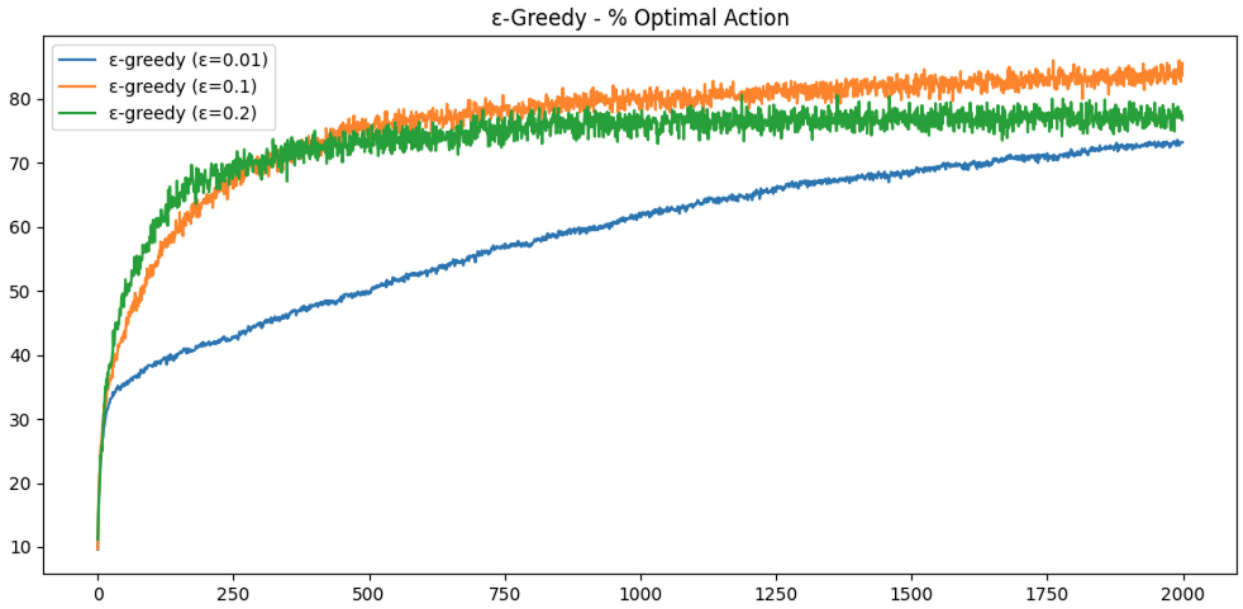


Figure 3 — Average Reward for UCB:
UCB curves show faster learning and quicker identification of the best arm. Larger c values explore more early on but quickly converge as uncertainty decreases, exactly as the theory predicts.

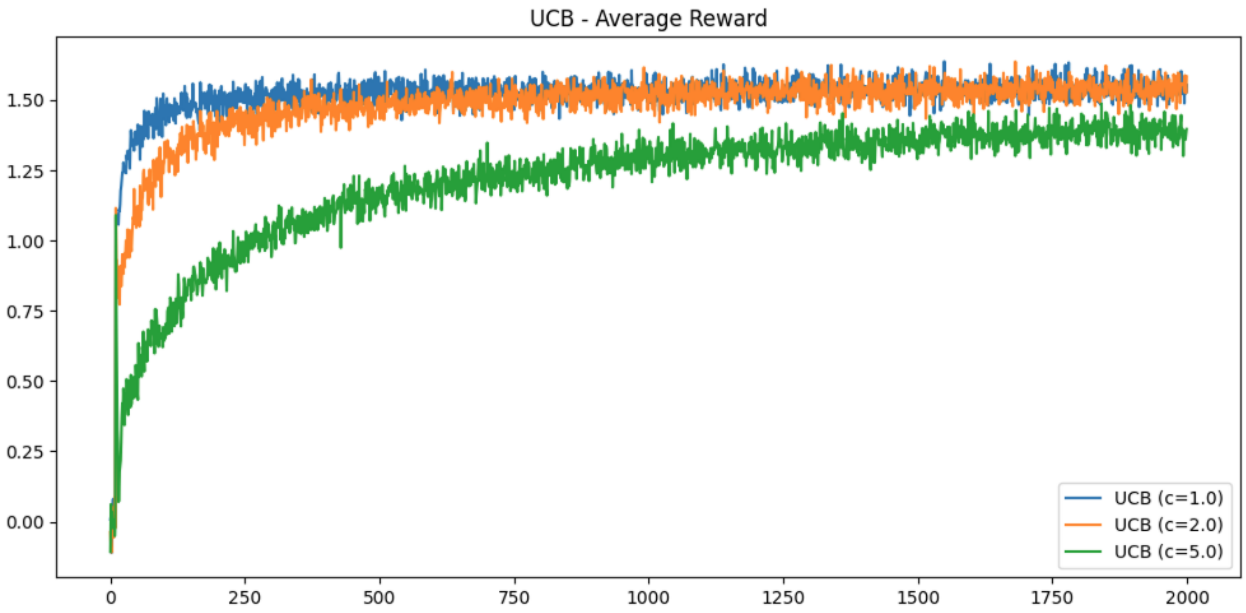
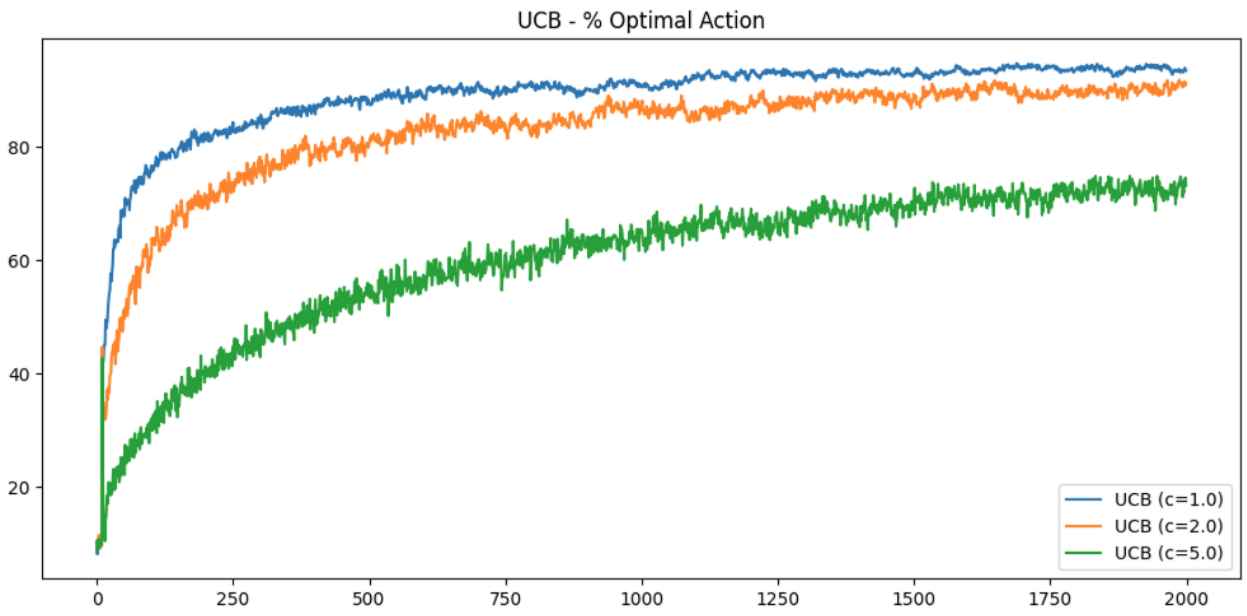


Figure 4 — Optimal Action % for UCB:
All UCB settings reach near-100% optimal selection, demonstrating that uncertainty-based exploration is more efficient than random exploration.



Section 3: AI Use Reflection

My first interaction with AI was asking for a full implementation of a 10-armed bandit environment and agents for ε-greedy and UCB. The model provided a complete structure, but the first version did not run correctly. This began the debugging process.

In the first iteration, I received several warnings and a shape error when resetting the environment. The observation returned as a float instead of an array. I asked the AI why Gymnasium expected an array. It explained that Gym environments require observations to follow the declared observation space. I updated the reset method to return a NumPy array, which fixed the issue.

In the second iteration, the UCB agent crashed because some arms had zero counts during the log calculation. I shared the error message with the AI. It suggested selecting each arm once before applying the UCB formula. I added a warm-up condition, which stopped the crash and allowed UCB to run correctly.

In the third iteration, the plotting functions failed due to mismatched lengths. I asked the AI for help, and it identified a missing key in the dictionary returned by the experiment function. I corrected the dictionary structure, and all plots were generated successfully.

The AI gave helpful fixes, but I still checked each suggestion to make sure it made sense. I confirmed correctness by printing intermediate values and running multiple seeds to ensure stable results. I also reviewed parts of the code against Sutton & Barto to make sure the formulas were implemented correctly.

From this process, I learned not just how ε-greedy and UCB work, but why directed exploration is more effective than random exploration. I also learned that AI is most useful when I combine its suggestions with my own checking and understanding.

Section 4: Speaker Notes

- **Goal:** Explore the exploration–exploitation problem by comparing ε-greedy and UCB in a 10-armed Gaussian bandit.
- **Approach:** Built a custom Gymnasium bandit environment and tested ε-greedy with three ε values and UCB with three different confidence levels.
- **Design Decision:** Used incremental mean updates and ran 1,000 repeats to get stable, smooth average performance curves.
- **Main Finding:** UCB identified the best arm much faster and stayed close to 100% optimal action selection, while ε-greedy was slower depending on the ε value.
- **Key Insight:** UCB's uncertainty bonus makes exploration more targeted and efficient compared to random ε-based exploration.
- **MDP Comparison:** Testing FrozenLake and Taxi highlighted how full MDP environments introduce states, transitions, and episodes, which bandits do not have.
- **Learning Outcome:** The debugging cycles improved my understanding of how action values update over time and how exploration strategies affect learning speed.