Homework 1:
Multi-Armed Bandits

In this problem set you will study multi-armed bandits. The exercises are taken from the book of Sutton and Barto.

Instructions. You are to submit a PDF for your answers and a Python notebok for your code. For the PDF, you can type your answers for example using latex or use handwritten notes as long as they are clearly and easily readable. Do not forget to fully justify your answers.

Question 1. In ε-greedy action selection, for the case of two actions and $\varepsilon = 0.5$, what is the probability that the greedy action is selected?

Question 2. Consider Figure 1 obtained for the 10-armed bandit example seen in class (each arm's reward has a normal distribution, with its own mean). The curves are obtained by averaging over 2000 (independent) runs of each algorithm. The average reward is computed by taking the average over the runs: At a given step, we look at the rewards obtained by all the runs at this step, and we compute the average. Similarly, the percentage of optimal actions correspond to the percentage of runs in which the optimal actions is played at each iteration. So you need to first run the 2000 times algorithm while storing the sequence of rewards and actions, and then compute, for each step, the average rewards and the percentages of optimal actions. Which method will perform best in the long run in terms of cumulative reward and probability of selecting the best action? How much better will it be? Express your answer *quantitatively* (not just qualitatively).

Question 3. Reproduce Figure 1 by implementing your own version using Python. The code must be provided in a notebook. The notebook submitted on Brightspace must show the execution of your code, which should result in two plots, as similar as possible to the ones in Figure 1 (since the results are stochastic, the curves will not be exactly the same, but the values and the comparisons between the curves should be similar). The clarity and the quality of the code will be part of the grade.

Question 4. In Figure 2 the UCB algorithm shows a distinct spike in performance on the 11th step. Why is this? Note that for your answer to be fully satisfactory it must explain both why the reward increases on the 11th step and why it decreases on the subsequent steps. Hint: If $c = 1$ (using the notation used in class c is the constant appearing in the UCB algorithm), then the spike is less prominent.

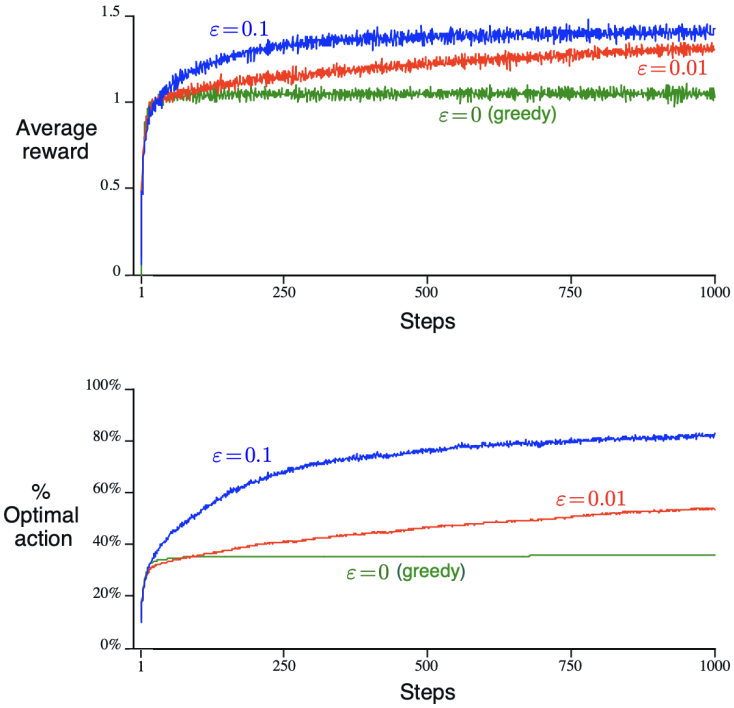

Figure 1: Figure 2.2 from Sutton and Barto's book: "Average performance of ε-greedy action-value methods on the 10-armed testbed. These data are averages over 2000 runs with different bandit problems. All methods used sample averages as their action-value estimates."

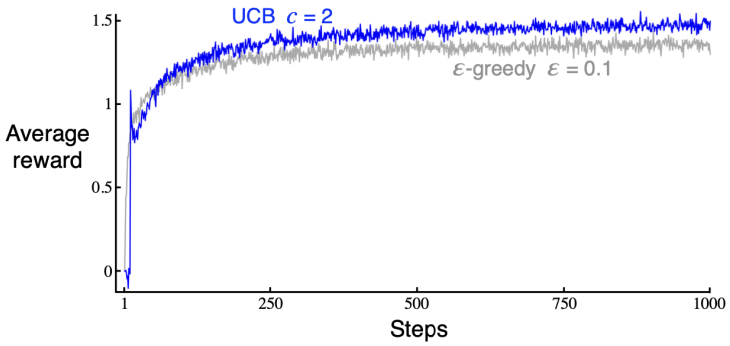

Figure 2: Figure 2.4 from Sutton and Barto's book: "Average performance of UCB action selection on the 10-armed testbed. As shown, UCB generally performs better than ε-greedy action selection, except in the first k steps, when it selects randomly among the as-yet-untried actions."