Reinforcement Learning

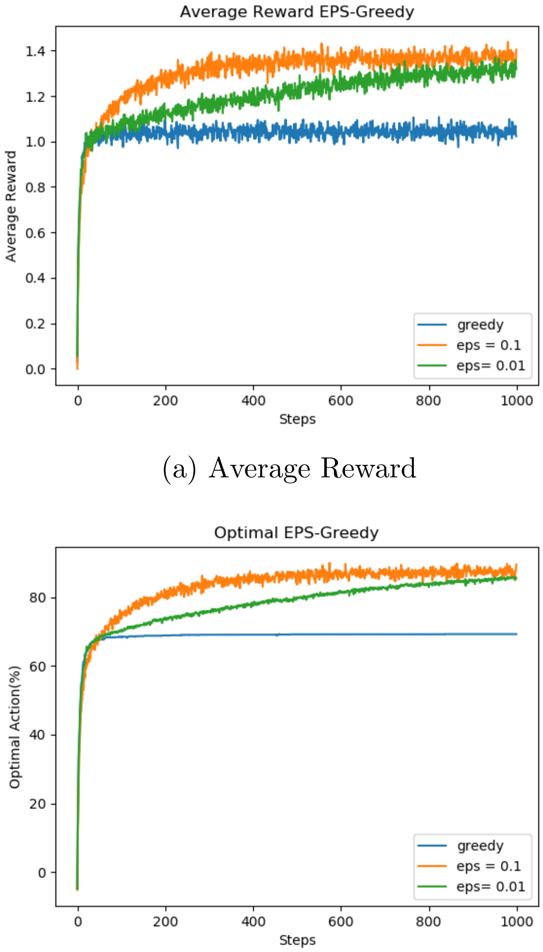
Prakhar Thakuria(EE16B061)

15 February 2019

Question1

Solution:

Below given figures(figure 1 and 2) are the replicas of the graphs in the book for 10-arm test bed and after the implementation of Epsilon - Greedy.



(a) Average Reward

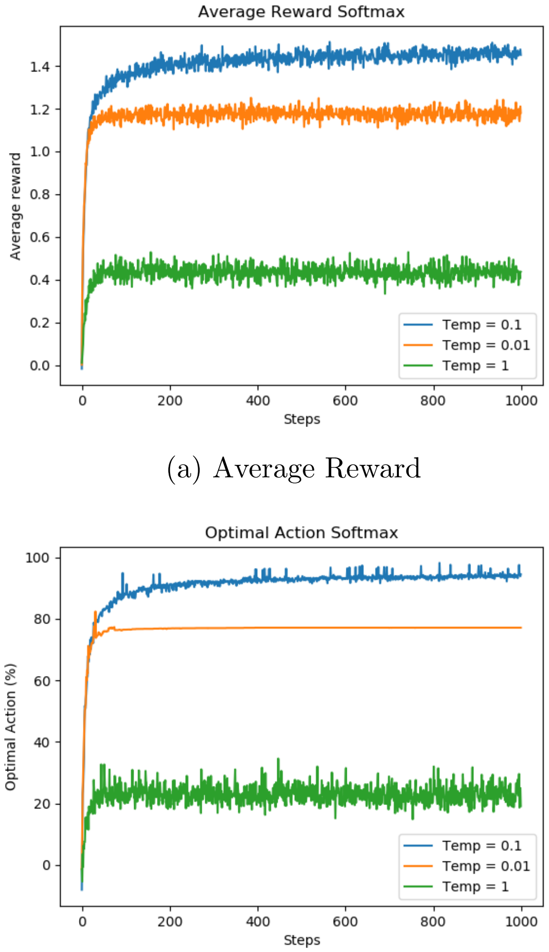
(b) Optimal Action

Figure 1: Eps-Greedy Test Bench

Question2

Solution:

Below given figures(figure 1 and 2) are the replicas of the graphs in the book for 10-arm test bed and after the implementation of Soft-max.



(a) Average Reward

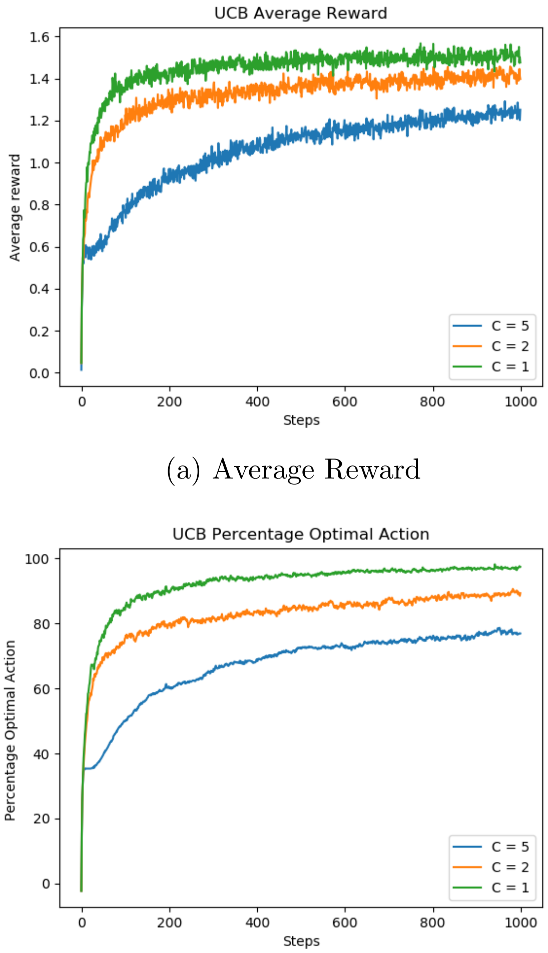
(b) Optimal Action

Figure 2: Soft-Max Test Bench

Question3

Solution:

Below given figures(figure 1 and 2) are the replicas of the graphs in the book for 10-arm test bed and after the implementation of UCB(upper confidence bound) algorithm.



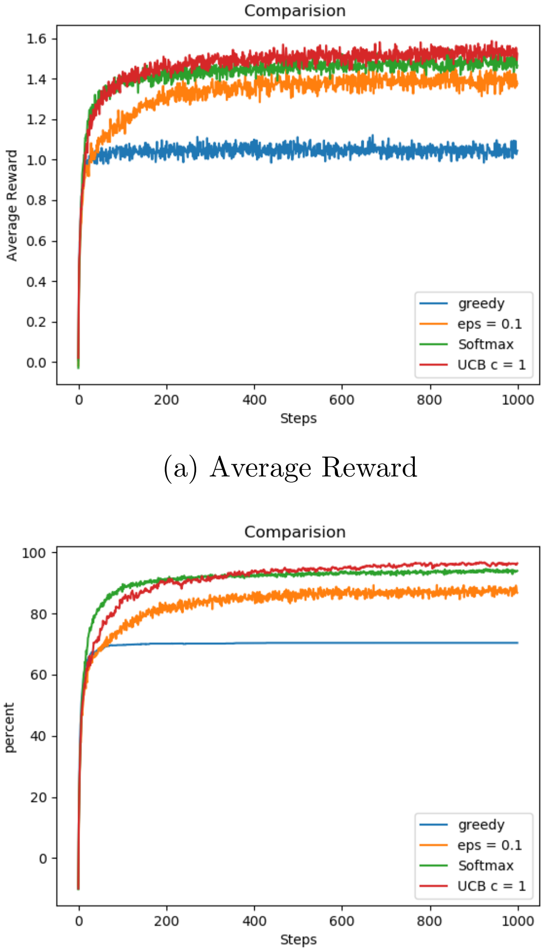
(a) Average Reward

(b) Optimal Action

Figure 3: UCB Test Bed

In comparison to the above 2, UCB converges very close to the optimum much sooner that the other two, but in the long run it seems like epsilon

greedy with the epsilon = 0.01 converges to a better optimum sooner than UCB(although it does much worse than UCB initially)(for proof the above observation refer More plots section and the one with 10000 iteration)
Reason seems to be with only 10 arms and over 10000 iterations there is a lot of chance of epsilon greedy to random check all the other arms and with the difference between the most optimal action being very small it takes a lot for the UCB to find it as it varies with log of number of iterations. Although it becomes much more clear in the 1000 arms testbed .



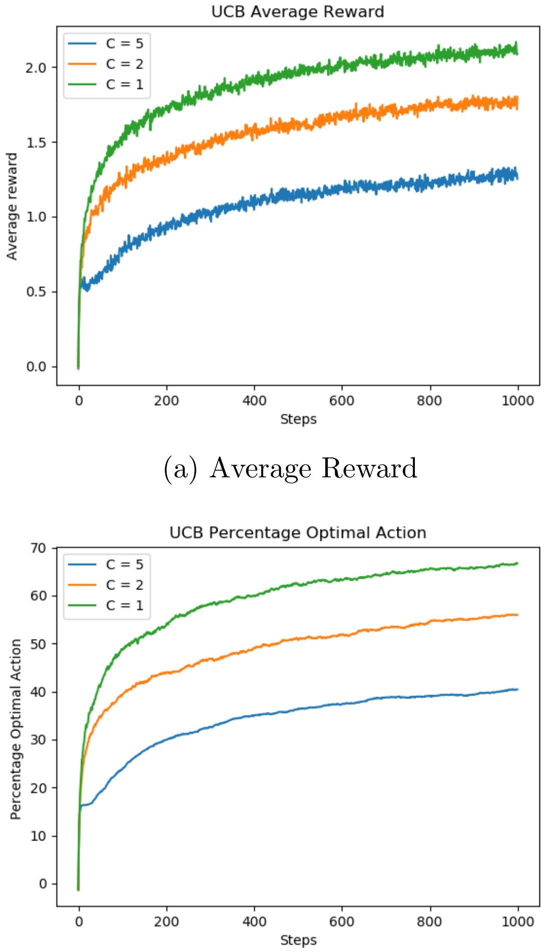
(a) Average Reward

(b) Optimal Action

Figure 4: Comparison between eps-greedy softmax and UCB

Question4

Solution: Different Algorithms performance on the 1000 arms test bench. **Upper confidence Bound**

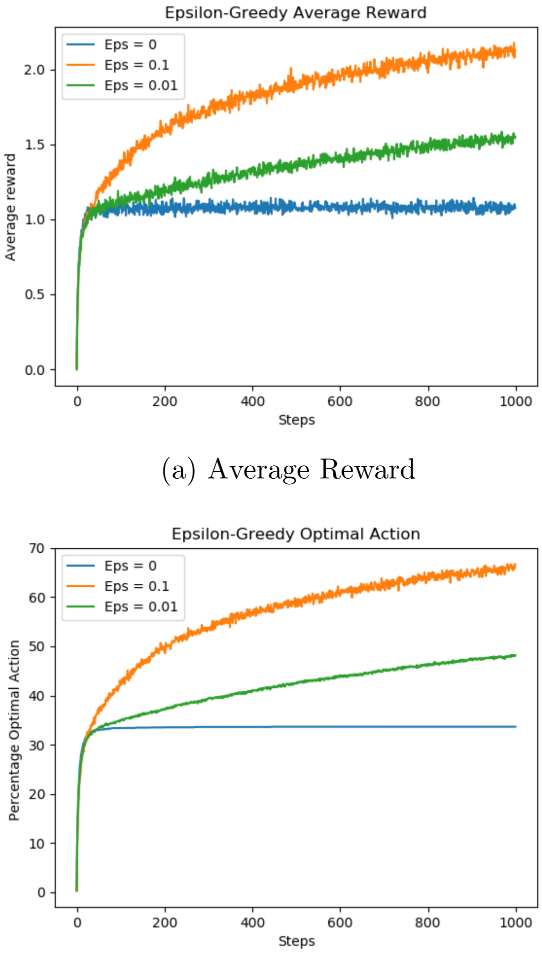


(a) Average Reward

(b) Optimal Action

Figure 5: UCB Test Bed 1000 arms

Epsilon Greedy



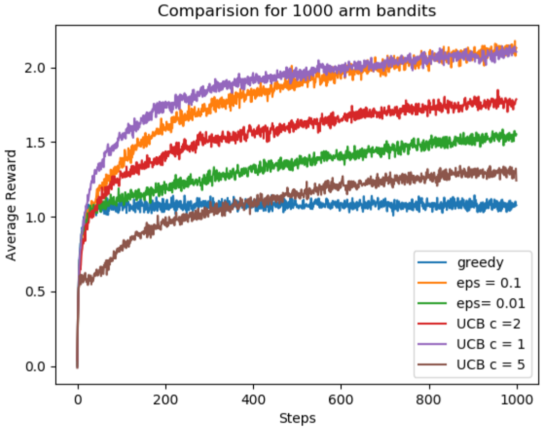
(a) Average Reward

(b) Optimal Action

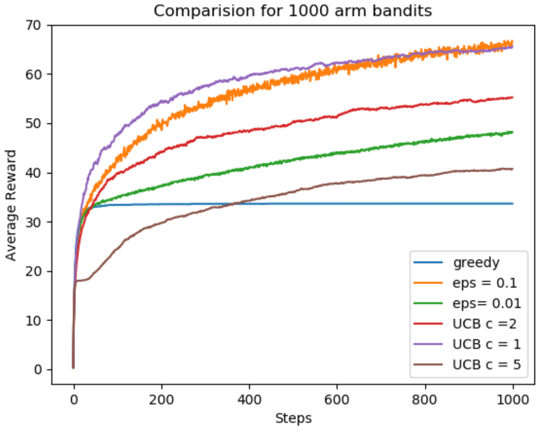
Figure 6: Epsilon-Greedy 1000 arms Test Bed

Comparison

UCB performs better in the starting but it looks like the epsilon greedy doing better in the start but epsilon greedy with epsilon = 0.1 looks like taking over the average returns as well as choosing the more optimal action



(a) Optimal Action



(b) Optimal Action

Figure 7: Comparison

Question5

Solution:

Plot for the average reward of the contextual bandits problems

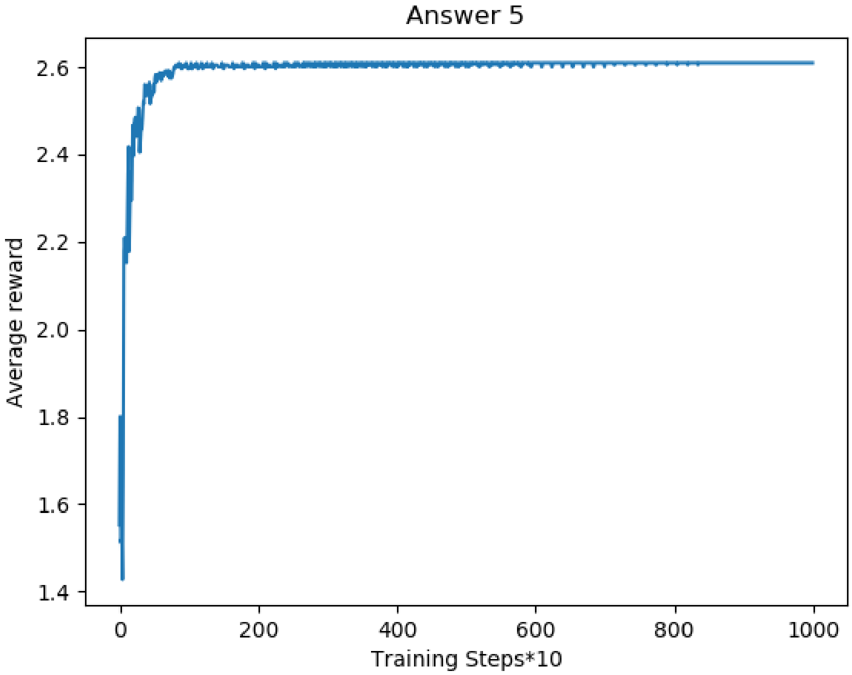


Figure 8: Average Reward contextual bandit

More plots

Comparison of softmax and epsilon

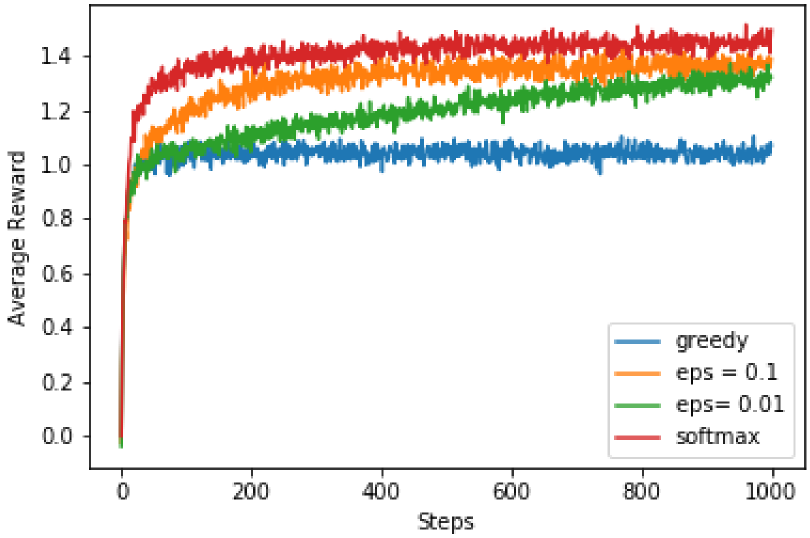


Figure 9: Compare softmax and eps-greedy

Instead of 1000 iterations doing 10000

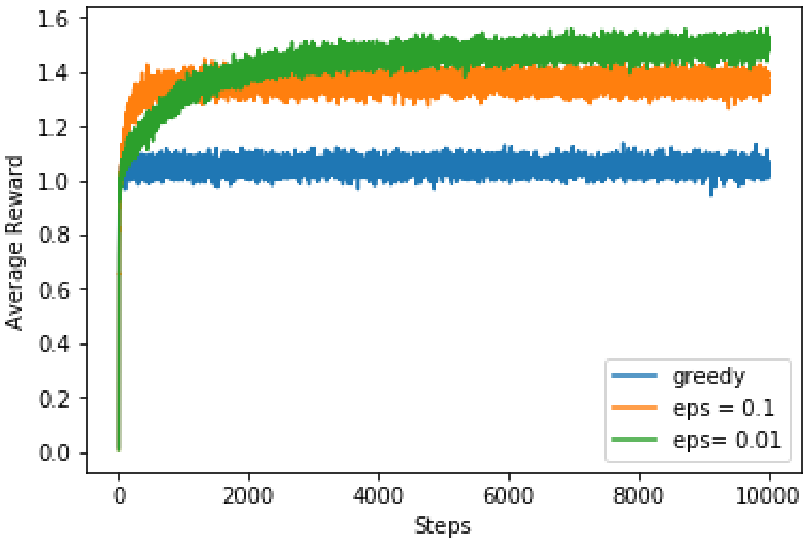


Figure 10: More iteration eps-greedy