Package inputenc Error: Unicode character (U+000F)not set up for use with LaTeXSee the inputenc package documentation for explanation.You may provide a definition with- greedy algorithms for 1000arm bandit test bed for 1000 plays over the average performance 2000 experiments. For- greedy=0.1 , softmax=0.1 1210lot of Average reward comparing UCB, softmax, ϵ-greedy algorithms for 1000arm bandit test bed for 1000 plays over the average performance 2000 experiments. For ϵ- greedy=0.1 , softmax=0.1 for figure.caption.16 - greedy algorithms for 1000arm bandit test bed for 10000 plays over the average performance 2000 experiments. For- greedy=0.1 , softmax=0.1 Package inputenc Error: Unicode character (U+000F)not set up for use with LaTeXSee the inputenc package documentation for explanation.You may provide a definition with- greedy algorithms for 1000arm bandit test bed for 10000 plays over the average performance 2000 experiments. For- greedy=0.1 , softmax=0.1 1311lot of Average reward comparing UCB, softmax, ϵ-greedy algorithms for 1000arm bandit test bed for 10000 plays over the average performance 2000 experiments. For ϵ-greedy=0.1 , softmax=0.1 figure.caption.18

Programming Assignment 1

CS6700 Reinforcement Learning

Author:

M V A Suhas Kumar
EE17B109
Department of Electrical Engineering

February 4, 2020

Contents

List of Figures

List of Tables

1 Introduction

In this assignment we are going to conduct experiments on 10-arm bandit test bed as described in the textbook. Algorithms that were experimented are as follows:

- ϵ-greedy
- Softmax action selection
- UCB1
- Median Elimination Algorithm

In each experiment the graphs are run for 1000 plays (except for MEA), with each curve being the average of the performance of 2000 different bandit problems. For every run, the true mean of each arm is sampled again from standard normal distribution as stated in the textbook and rewards are sampled from a Gaussian distribution with variance 1 and mean of the arm selected.

For all the algorithms we initially all the estimates of expectation of all the arms are set to zero

$$Q_0(a) = 0 \, \forall a \in A$$

Updating the estimation of expectation of the arm after sampling is done as follows:

$$Q_{t+1}(a) = Q_t(a) + \frac{1}{N(a)}[r_t - Q_t(a)]$$

2 ϵ-Greedy

ϵ - greedy algorithm involves both exploration and exploitation. Here exploration is done with a probability of ϵ and exploitation of the best arm is done with a probability of $1-\epsilon$ and greedy action is selected based on the current estimated expectation of the arms.

We experimented with this algorithm for different values of ϵ = 0,0.01,0.1 and the graphs obtained for the average reward and percentage optimal action chosen are as follows

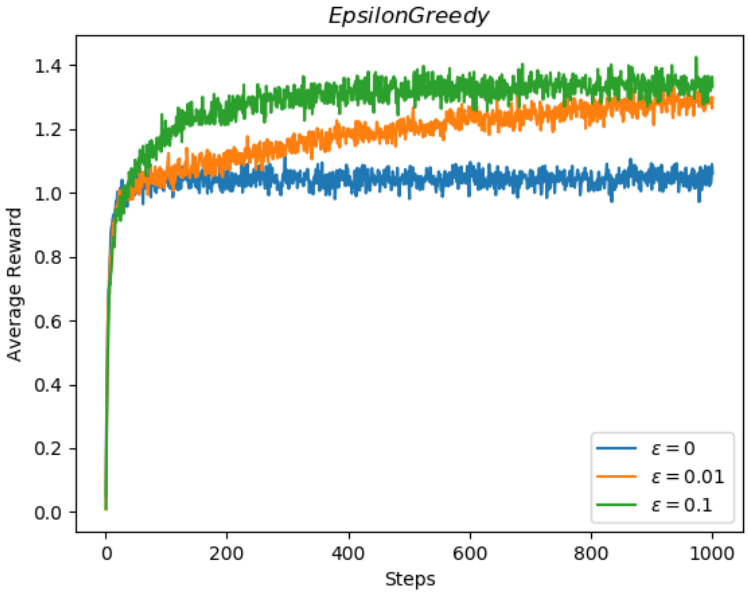

Figure 1: Average Reward Plot for Epsilon Greedy

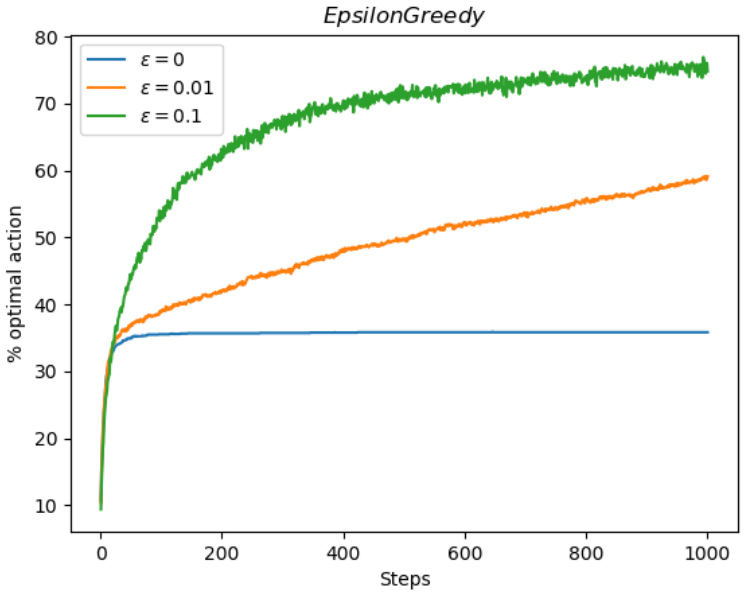

Figure 2: % optimal action Plot for Epsilon Greedy

In the Fig: 1 purely greedy approach has initially higher growth rate but eventually ϵ- greedy method performed better because it continued to explore more and find the optimal arm. $\epsilon = 0.1$ explored fast and eventually found the optimal action earlier and $\epsilon = 0.01$ method is improving slowly and outperforms $\epsilon = 0.1$ if given enough time. Finally we see that from both average reward and % optimal action plot ϵ=0.1 performed the best for 1000 steps

3 Softmax action selection

In softmax algorithm unlike the greedy algorithm each action selection probabilities are a function of estimated expected value of the arm. Instead of selecting the highest probable arm we sample from the distribution although greedy arm will have highest probability out of all. This introduces some sort of exploration which includes high probability of selecting good arms (more reward but less than greedy arm). Here for soft max method we use Gibbs distribution. Probability of choosing arm a at tth play as follows

$$\frac{e^{\frac{Q_t(a)}{\tau}}}{\sum_{b=1}^{n} e^{\frac{Q_t(b)}{\tau}}}$$

where τ is the temperature.

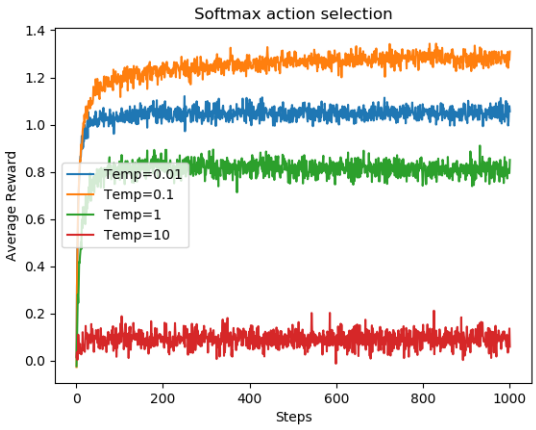

Figure 3: Average Reward Plot for Softmax action selection

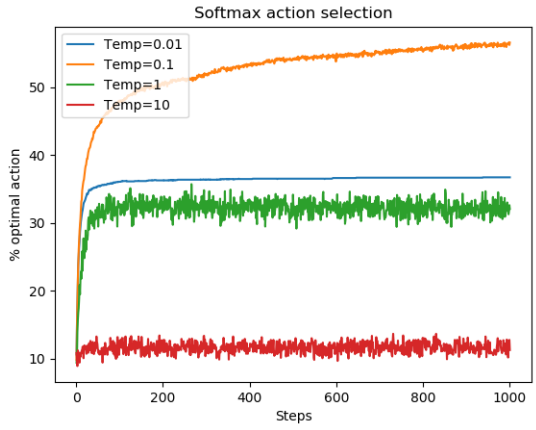

Figure 4: % optimal action Plot for Softmax action selection

From Fig 3 and 4 we see that $\tau = 0.1$ performs the best among all the temperatures. As the temperature increases the domination of temperature over Q(a) increases and make the probabilities go towards equal which in turn mean selecting arm at random which is clearly seen in the plot at $\tau = 10$ where % optimal action selection was close 10% i.e random selection. Lower temperatures cause great difference in probability and as τ tends towards zero it becomes same as greedy action selection. This can be clearly seen τ=0.01 where both average reward and % optimal action values are close to greedy algorithm in Fig 1 and Fig 2 respectively. Finally we could say that $\tau = 0.1$ is a correct balance between exploration and exploitation and performs the best.

4 UCB1

In UCB algorithm unlike ϵ greedy algorithm while selecting the non greedy it would consider how close estimates are being to maximum and the uncertainties in those estimates. Here actions are selected as follows:

$$A_t = argmax\ [Q_t(a) + c\sqrt{\frac{ln(t)}{N_t(a)}}]$$

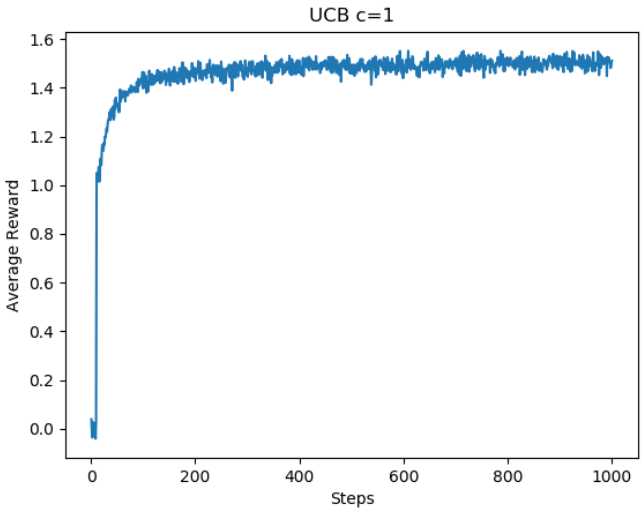

Figure 5: Average Reward plot for UCB

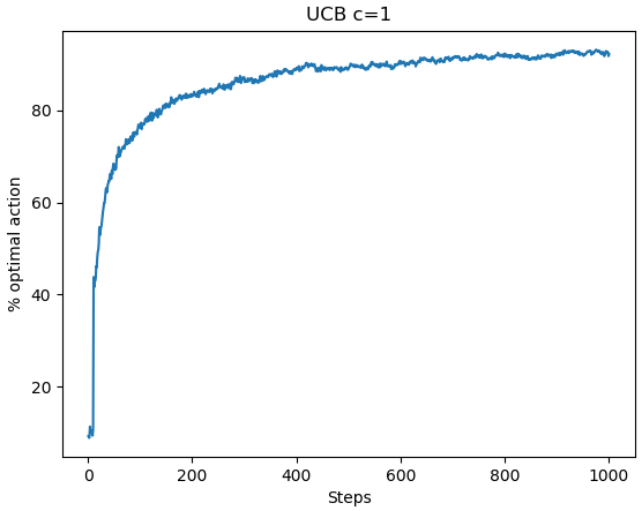

Figure 6: optimal action plot for UCB

5 UCB vs ϵ-Greedy vs Softmax

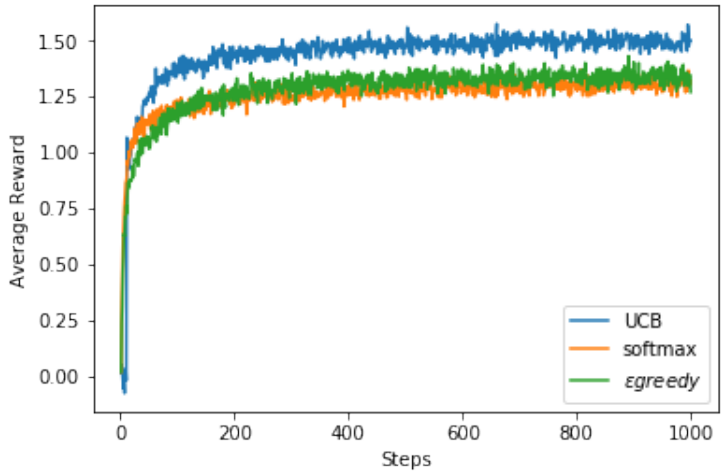

Figure 7: Plot of Average reward comparing UCB, softmax,ϵ - greedy algorithms for 10 arm bandit test bed for 1000 plays over the average performance 2000 experiments. For ϵ - greedy ϵ=0.1 , softmax τ =0.1

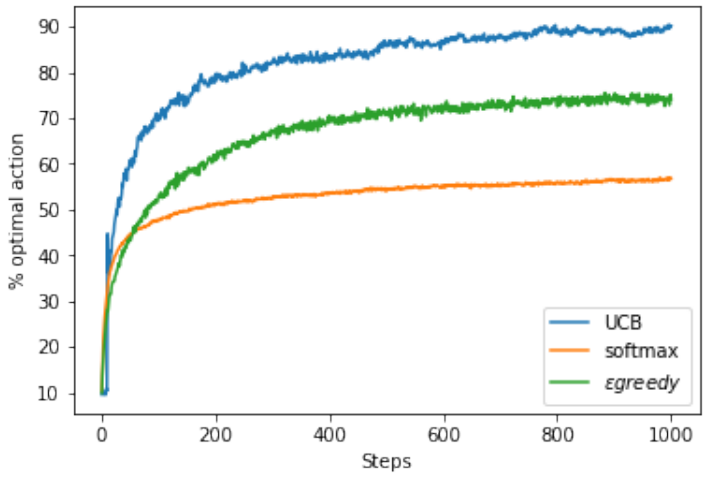

Figure 8: Plot of % optimal action comparing UCB, softmax,ϵ - greedy algorithms for 10 arm bandit test bed for 1000 plays over the average performance 2000 experiments. For ϵ - greedy ϵ=0.1 , softmax τ =0.1

From Fig 7 and Fig 8 we could clearly observe that UCB is performs the best both in terms of average reward and % optimal action. Initially UCB performs poorly than others because it tries to explore all the arms before repeating and eventually it performs the best because in ϵ - greedy exploration is random (all arms are equi probable while exploring) where as in UCB it is smart exploration i.e it would prefer the arms more which are having high chance of becoming optimal arm.

Here for the chosen values of epsilon and temperature ϵ- greedy performs better than softmax.

6 Median Elimination Algorithm

6.1 Comparing different epsilon and delta

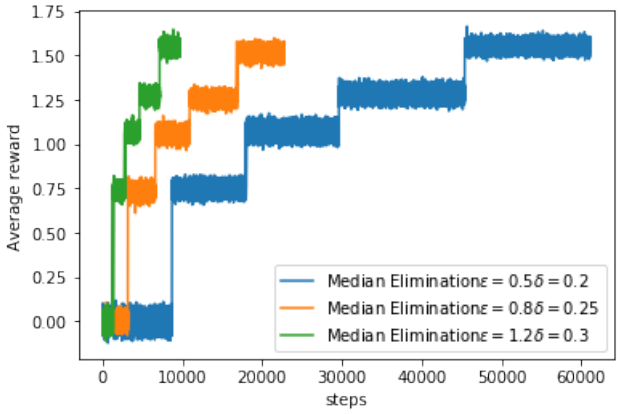

Figure 9: Plot of Average reward for median elimination Algorithm by varying epsilon and delta

Here we ran the Median Elimination Algorithm for different values of epsilon and delta. In the above we see that for different values of epsilon and delta the final average reward is settling to almost value but if we observe by printing the final values in the python code we see that $\epsilon = 1.2$ and $\delta = 0.3$ is giving the highest reward.

Yes computing the median is the rate determining step. The complexity for finding median if in lth round there are k_l arms are present is $k_l log(k_l)$ and no of rounds in MEA are log(k) where k is total number of arms.

6.2 Comparing among different Algorithms

Here we compare all 4 algorithms for 10 arm bandit testbed.

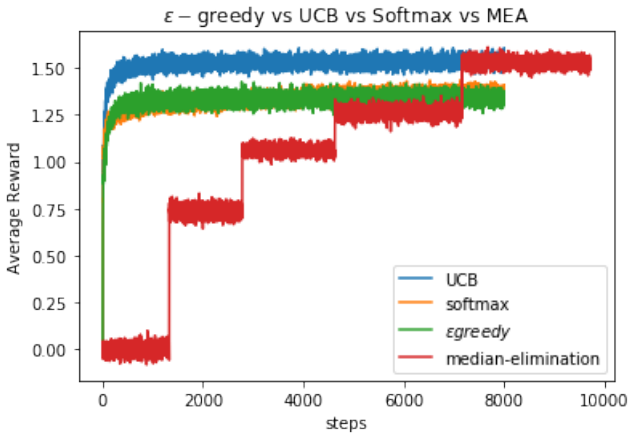

Figure 10: Average reward for 10 arm bandit testbed comparing MEA with $\epsilon = 1.2$ and δ=0.3 , UCB1 , ϵ-greedy with $\epsilon = 0.1$ and softmax with temp $= 0.1$ over the average performace of 2000 experiments

7 1000-Arm bandit test bed

Now, Here we are comparing the algorithms for 1000 arm bandit test bed. Firstly, to notice as the number of arms increases the number of steps required for median elimination all increases highly. So it is not practical to possible to run other algorithms for that many steps. Even though other algorithms also take more steps compared to 10 arm case. So, here we plot median elimination algorithm separately and plot remaining algorithms for comparison

7.1 MEA for 1000 arms

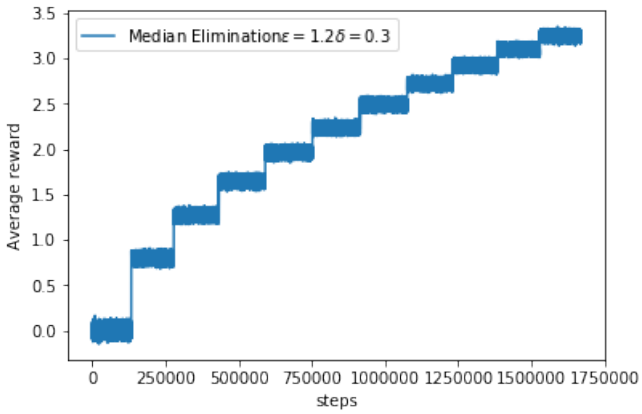

Figure 11: Plot of Average reward for Median elimination algorithm for 1000 arm bandit testbed performance average for 2000 runs with ϵ =1.2 and $\delta = 0.3$

In the above figure we clearly see that as the arm grows no of steps required for median elimination algorithm is high and MEA average reward converges to true mean.

7.2 UCB VS Epsilon Greedy VS Softmax

UCB VS Epsilon Greedy VS Softmax for 1000arm and 1000 steps

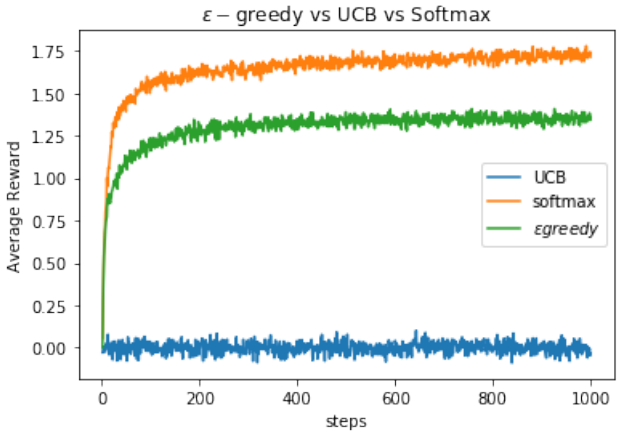

Figure 12: lot of Average reward comparing UCB, softmax, ϵ-greedy algorithms for 1000arm bandit test bed for 1000 plays over the average performance 2000 experiments. For ϵ- greedy=0.1 , softmax=0.1 for

If we run the 1000 arm case only for 1000 steps UCB algorithm performs the worst and we could say that ϵ greedy and softmax better than UCB and cant compare between those two because inherently both are dependent on ϵ and temperature values. We could only say that for the given values of those softmax is performing better than ϵ greedy.

Here UCB for 1000 time-steps performing bad just because UCB continues to play each arm once while ϵgreedy and softmax action selection behave greedily majority of the time. In the whole steps UCB picked the optimal arm only once.

UCB VS Epsilon Greedy VS Softmax for 1000arm and 10000 steps

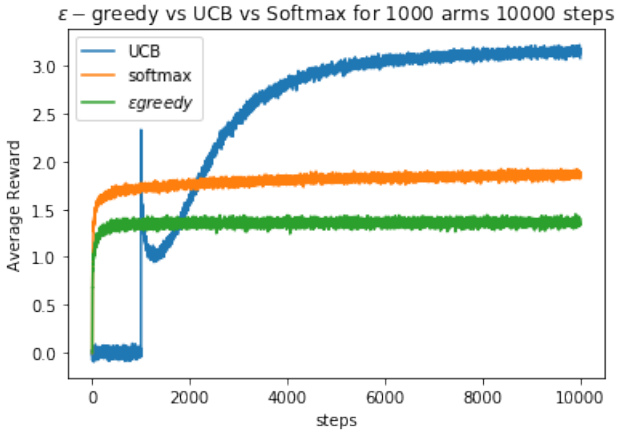

Figure 13: lot of Average reward comparing UCB, softmax, ϵ-greedy algorithms for 1000arm bandit test bed for 10000 plays over the average performance 2000 experiments. For ϵ-greedy=0.1 , softmax=0.1

Here when we give UCB Algorithm enough time we see that it clearly outperforms ϵ greedy and softmax algorithms. Also to note we see that UCB gets close to the optimal reward compared to softmax and ϵ greedy. So we can say that as the number of arms increases UCB performs better than other two algorithms.