CS-747 Assignment 1 Report
Arun Verma, 154190002

Observation about epsilon value in epsilon-greedy algorithm:

As shown in Figure 1 (5 Arms) and Figure 2 (25 Arms), the optimal value of epsilon is 0.01 and 0.05 respectively. The deviation from these value leads to imbalance between exploration and exploitation. If value of epsilon is less than there will be less exploration and more exploitation and other case more exploration and less exploitation. Second case (25 Arms) needs more exploration due to more arms hence larger value of epsilon gives less regret.

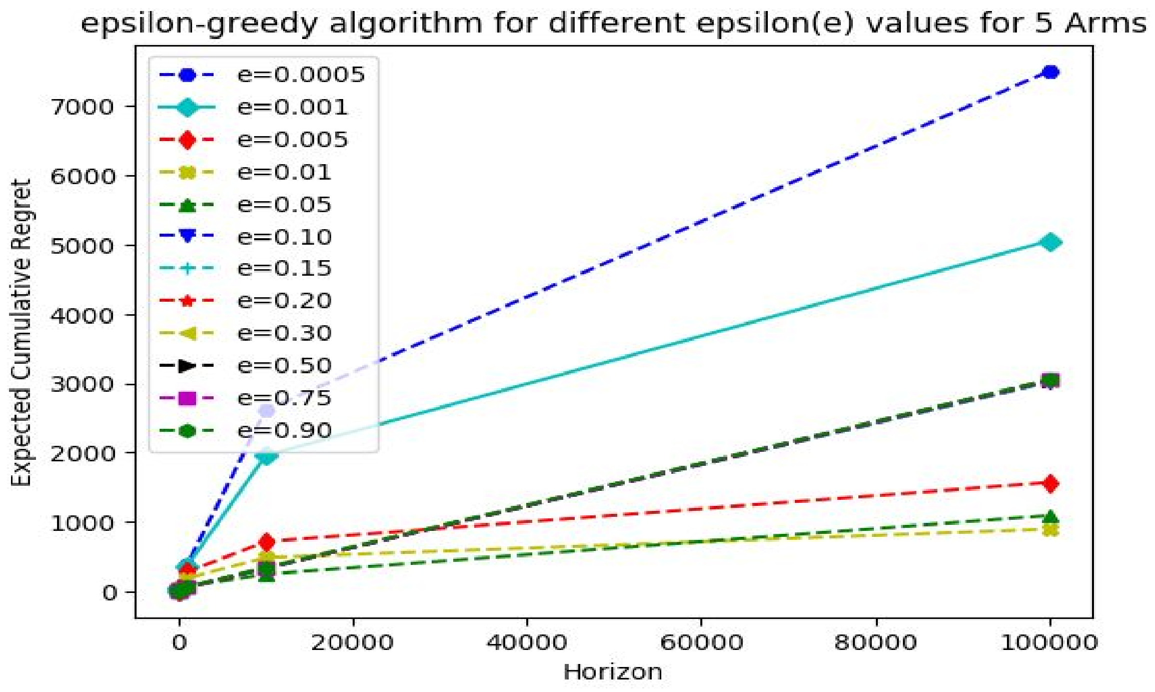

Figure 1

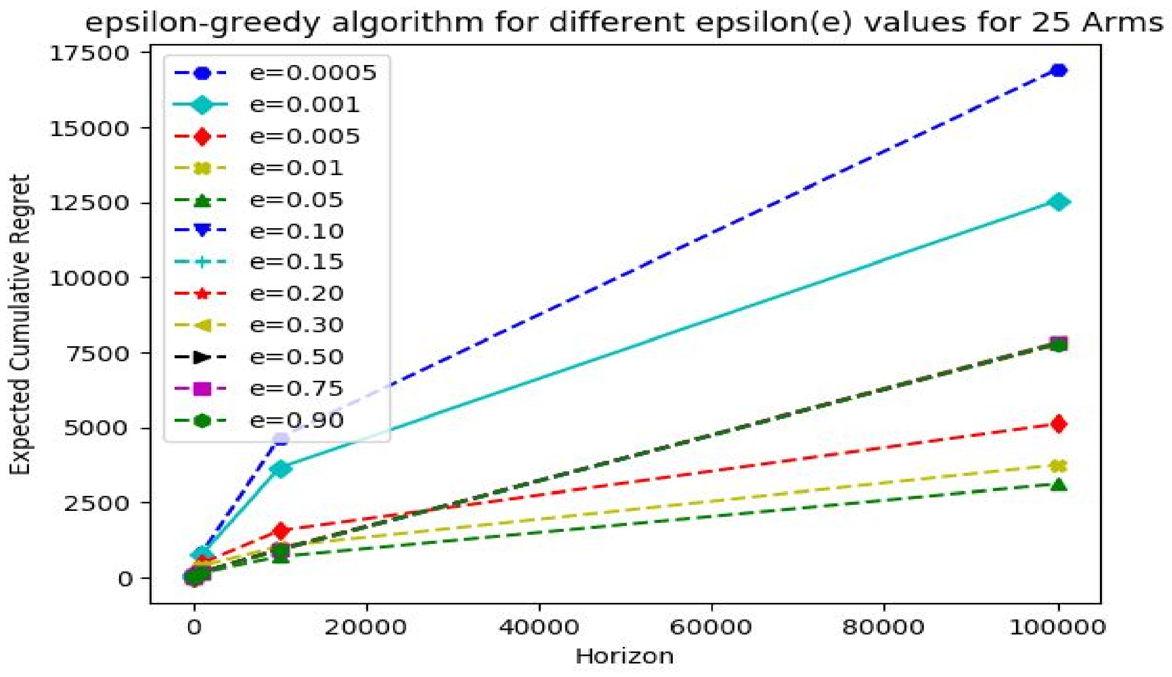

Figure 2

Regret of different Bandit Algorithm:

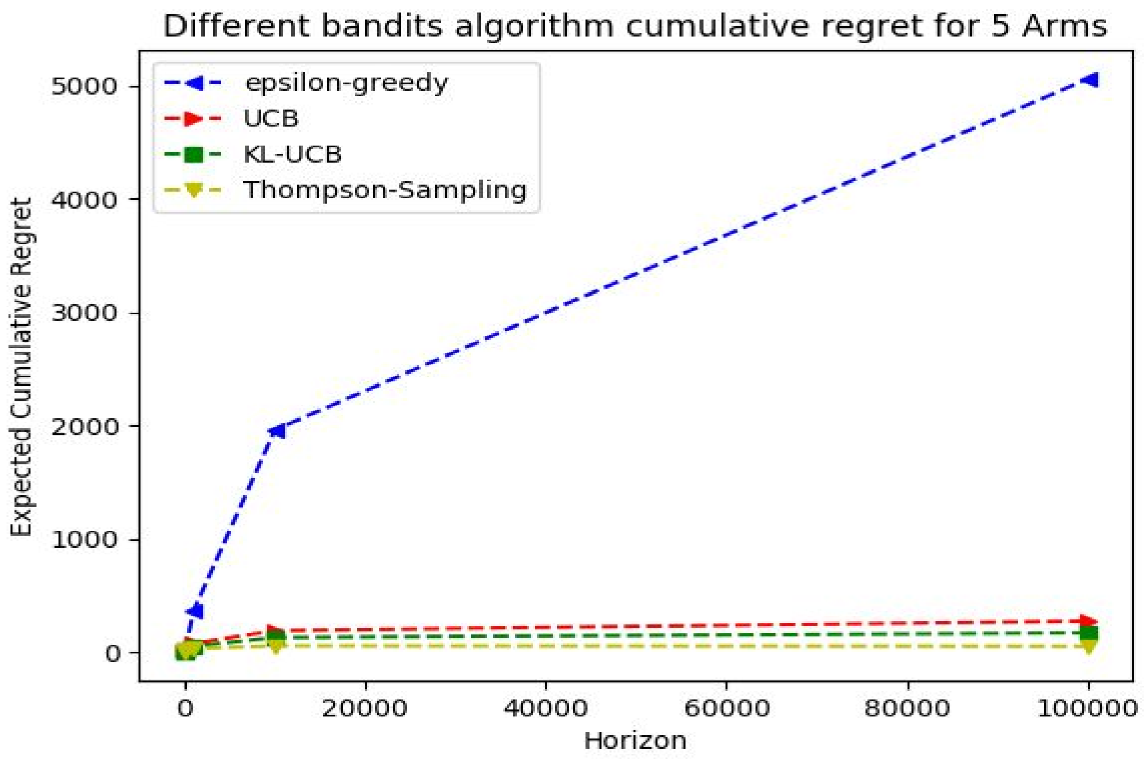

Figure 3

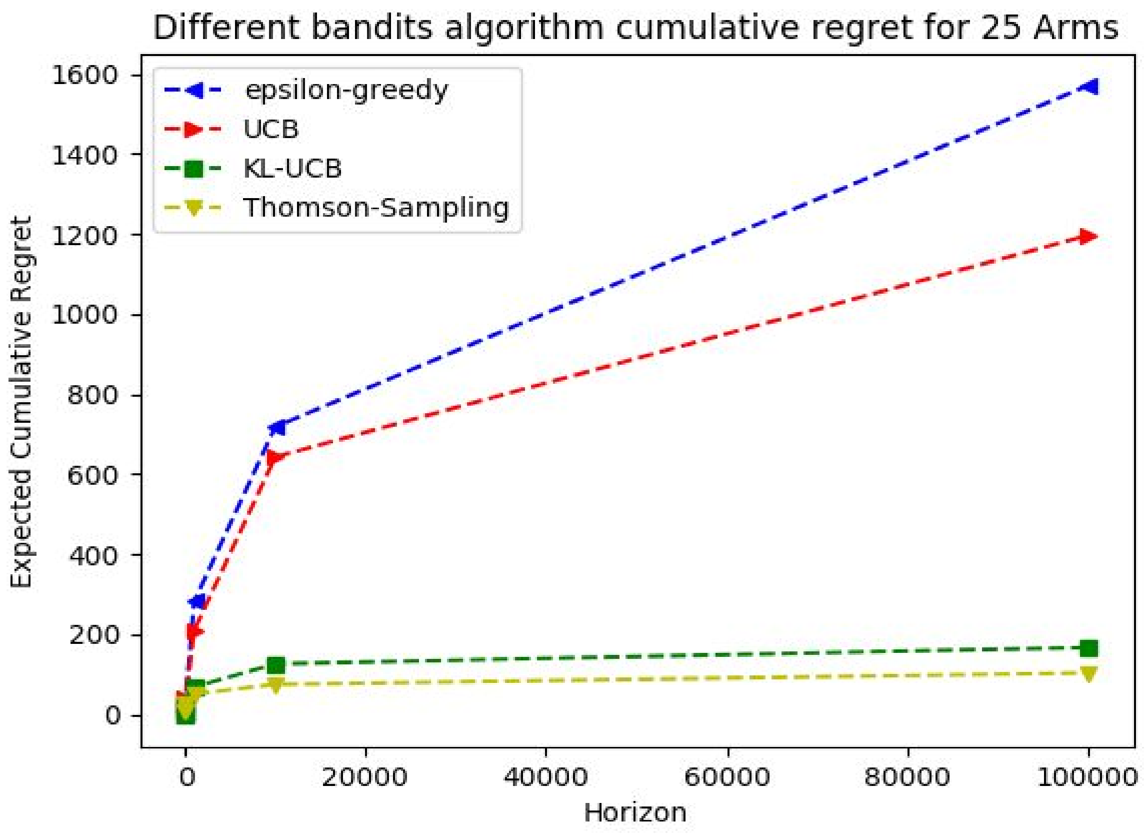

Figure 4

As shown in Figure 1 (5 Arms) and (25 Arms), cumulative regret of epsilon-greedy, UCB, KL-UCB and Thompson-Sampling is shown. It is clearly shown that Thompson-sampling perform better than other algorithms. KL-UCB also gives better results but due to solve a optimization problem in each round to find the upper bound, KL-UCB is very computationally intensive algorithm. As number of arms increases, computation requirements also grows multiplicative factor (by number of arms) in KL-UCB algorithm. While Thompson-Sampling works better as a bayesian update needs in each round.

Running code without using TCP server:

Sometimes the connection with TCP server is breaking when I was running the code for longer number of runs. So I have used different code to implement the same TCP server functionality. The code is kept inside ../client/report

Some file notations:
e_* .log - Logs generated by epsilon-greedy algorithm
u* .log - Logs generated by UCB algorithm
k* .log - Logs generated by KL-UCB algorithm
t* .log - Logs generated by Thompson-Sampling algorithm
*_25.log - Logs generated for 25 arms instance
e_001_*.log - Logs generated by epsilon-greedy algorithm for epision value, e=.001 * 0.1

To run the code please run below commands:

```
./bandit-agent.py algorithm_name numberRuns epsilonValue 0/1 >  /log/*.log
```

0/1 -> 0 for 5 arms instance and 1 for 25 arms instance

Example: `./bandit-agent.py epsilon-greedy 100 0.01 0 > /log/e_1.log`

To plot the collected results:

```
python plot.py
```