This assignment consisted of implementing and testing various algorithms including epsilon-greedy exploration, UCB, KL-UCB, and Thompson Sampling for a multi-armed bandit problem.

All these algorithms are implemented in the file 'banditagent.cpp". For each bandit instance from instance-5.txt,instance-25.txt; each algorithm from epsilon-greedy, UCB, KL-UCB, Thompson-Sampling; each horizon in 10, 100, 1000, 10000, 100000, 100 runs were generated by varying the random seed. The average cumulative reward vs the Horizon is plotted for the two bandit instances as shown in Figure 1 and 2 respectively.
Please refer to README.md in the 'report' directory for information about running the experiments.

1. SOME IMPLEMENTATION DETAILS

- The value of epsilon used for the epsilon-greedy method is fixed at 0.1. For the epsilon-greedy, an estimate of the true mean of each bandit arm is kept at each time instance, and the arm with max estimated mean is picked with probability 0.9, while with the remaining probability 0.1, a randomly sampled arm is picked (i.e. pulls % numArms)

- For the UCB/KL-UCB algorithm, the upper confidence bounds(ucb) for each arm is computed at each step and the arm with the max ucb value is pulled at that step. Algorithm 1 from the paper The KL-UCB Algorithm for Bounded Stochastic Bandits and Beyond is used as reference with the value of c = 3.

- For solving the inequality involved in the KL-UCB, since $f(q) = KL\_bernoulli(p, q)$ is an increasing function of q when $q \in (p, 1)$ for a fixed $p \in [0, 1]$, binary search with a error bound of 1e-8 is used to find the maximum value of q satisfying the inequality of the form $KL\_bernoulli(p, q) \leq C$ where C is some constant.

- Sampling from beta distribution is implemented using the gamma distribution provided by STL in C++ used in Thompson Sampling.

It can be easily observed from Figures 1 and 2 that the regret increases as we increase the horizon for all the algorithms. Also, note that Thompson Sampling performs much better as compared to other algorithms in terms of regret minimization. The order of performance is roughly: Thompson Sampling > KL-UCB > UCB > epsilon-greedy.

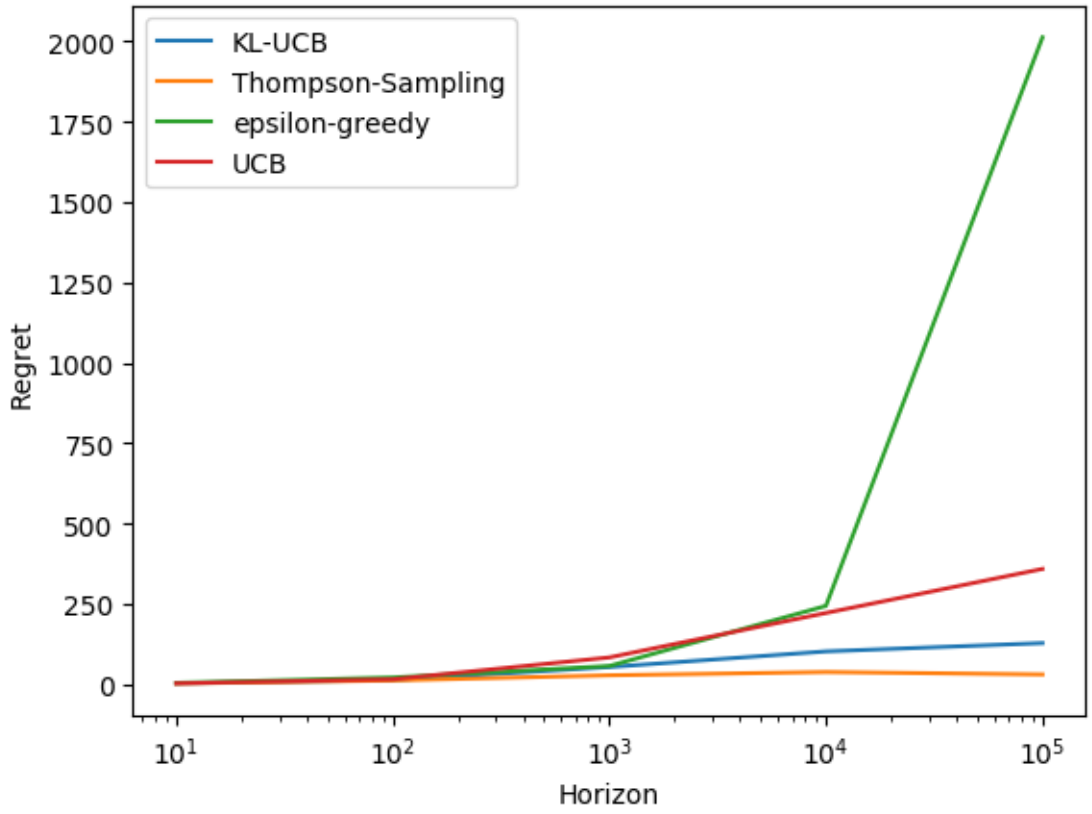

FIGURE 1. Horizon vs Expected Cumulative regret on the bandit instance instance-5.txt

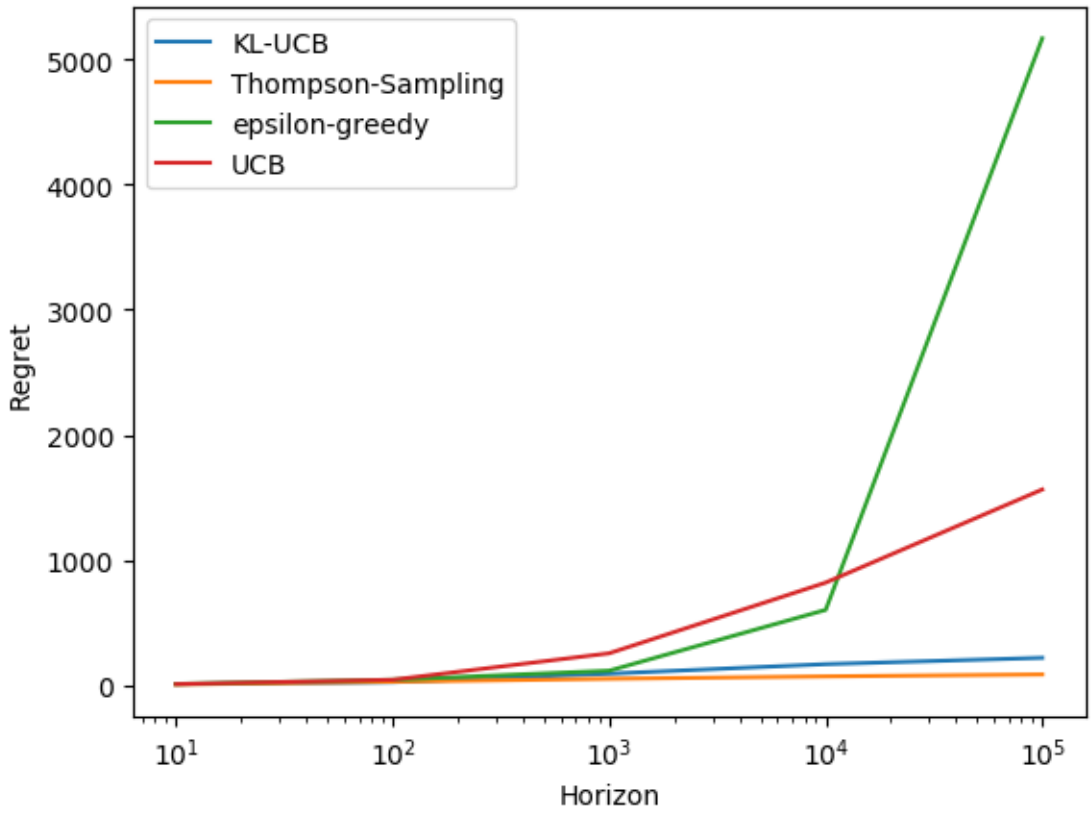

FIGURE 2. Horizon vs Expected Cumulative regret on the bandit instance instance-25.txt