ID6002W: Online Learning and Reinforcement Learning
Assignment 1

Course Instructor : Arun Rajkumar.

Release Date : May 15, 2023

Submission Date: On or before 11:59 PM on June-11,2023

SCORING: The assignment contributes to 25 points towards your final grade.

WHAT SHOULD YOU SUBMIT? You should submit a zip file titled 'Solutions_ roll-number.zip' Your assignment will NOT be graded if it does not contain all of the following:

- A PDF file which includes explanations regarding each of the solution as required in the question. Title this file as 'Report.pdf'
- Source code for all the programs that you write for the assignment clearly named.

CODE LIBRARY: You are expected to code all algorithms from scratch. You cannot use standard inbuilt libraries for the algorithms. You are free to use inbuilt libraries for plots. You can code using either Python or Matlab or C.

GUIDELINES: Keep the below points in mind before submission.

- Plagiarism of any kind is unacceptable. These include copying text or code from any online sources. These will lead to disciplinary actions according to institute guidelines.
- Any graph that you plot is unacceptable for grading unless it labels the x-axis and y-axis clearly.
- Don't be vague in your explanations. The clearer your answer is, the more chance it will be scored higher.

LATE SUBMISSION POLICY You are expected to submit your assignment on or before the deadline to avoid any penalty. Late submission incurs a penalty equal to the number of days your submission is late by.

Q1: Cats and Dogs In this question, you must develop an algorithm to play the game of Cats and Dogs. The game has two players (you and an adversary). The adversary has a four letter word in mind which she picks from the list given here. The adversary is allowed to pick only words without repeating alphabets. For instance the adversary cannot pick 'LOSS'. You know this list and your goal is to guess the specific word the adversary has chosen.

The game proceeds in rounds and in every round you are allowed to guess a four letter word from the list which also does not have any repeating alphabets. The adversary then gives you feedback in the form of two ordered numbers: a number C indicating the number of Cats in your guess and a number D indicating the number of Dogs in your guess. A Cat is an alphabet which is common to both your guess and the adversary's chosen word but is not in the same position. A Dog is an alphabet which is common to both your guess and the adversary's chosen word and also is in the same position. For example if the adversary's word is 'COLD' and your guess is 'LAST', the adversary would say '1C 0D' indicating that there is one Cat in your guess (namely the alphabet 'L' which has moved to a different position) and no dogs. If your guess had been 'CLAD', the adversary would say '1C 2D' indicating that there is 1 Cat (namely 'L') and 2 Dogs (namely 'C' and 'D') in your guess. Note that the adversary does not tell you which alphabets are Cats and which are dogs but just gives you 2 numbers indicating the number of Cats and Dogs.

Your goal is to develop an algorithm which will try to guess the adversary's word as quickly as possible. Your algorithm will be judged based on the average number of guesses it takes to guess where the average is over all possible words the adversary is allowed to choose from. In your report, give the pseudo code for your algorithm, explain your algorithm in words and also give the average number of guesses over all possible words. Also give information about which word was the toughest for your algorithm to guess and how many guesses did your algorithm take to figure it out. Argue if your algorithm is optimal. If not, explain what may be the source of sub-optimality and why.

Q2: Randomizing the Leader: Consider the problem of online learning on the simplex Δ_d where d = 1000; At round t, you predict p_t and receive a vector z_t and suffer a loss of $p_t^T z_t$. Assume the adversary picks the vector z_t as the t-th row in the dataset *Dataset_Z*. Implement FTRL with quadratic and entropic regularization for this problem and plot the regret over time.
Now, consider the following algorithm which first picks and fixes a random 1000 dimensional vector R sampled uniformly from $[0, 1/\eta]^d$ and uses the following rule for prediction

$$p_{t+1} = \arg\min_{p \in \Delta_d} \sum_{i=1}^{t} \left(p^T (z_i + R)\right)$$

How would you choose η for this problem? For the value chosen, plot the regret bound for this algorithm as well. How does the regret bound compare with the previous two algorithms for this problem?