Machine Learning for Personalization
Homework 2

You will build a contextual bandit algorithm by implementing the (disjoint) LinUCB algorithm that we covered in class and that is summarized in the suggested reading paper *A Contextual-Bandit Approach to Personalized News Article Recommendation*.

Start by downloading the data-set at

`http://www.cs.columbia.edu/~jebara/6998/dataset.txt`

This is a personalization data-set stored as a text file. It contains $t = 1, \ldots, 10000$ data points (rows) each having 102 values (columns). Each data-point is a row in the file and the dimensions (or columns) of the data-point are separated by spaces.

The first column in the data-set is the action that was performed which is a value $a_t \in \{1, \ldots, 10\}$. The second column is a binary value indicating if a reward was obtained $y_t \in \mathbb{B}$. The remaining 100 columns are the context which is represented as a vector $\mathbf{x}_t \in \mathbb{R}^{100}$ which is stored a integer values in the file simply for numerical efficiency.

Perform linUCB in an online manner on this data-set and compute your performance in an online manner using Replay methods as described in the suggested reading paper *Unbiased Offline Evaluation of Contextual-bandit-based News Article Recommendation Algorithms*.

Compute the cumulative take-rate of your actions as follows for any time t as you scroll through the data-set. Let π_{t-1} be your algorithm trained on data up to time $t-1$. Let $\pi_{t-1}(\mathbf{x}_t)$ be the action that algorithm π_{t-1} chooses for the context $\mathbf{x}_t$ which it observes at time t. Let a_t be the real action that was taken in the data-set at time t. Let y_t be the real reward that was obtained at time t. Then, the cumulative take-rate replay at time T is

$$C(T) \quad = \quad \frac{\sum_{t=1}^{T} y_t \times \mathbf{1}[\pi_{t-1}(\mathbf{x}_t) = a_t]}{\sum_{t=1}^{T} \mathbf{1}[\pi_{t-1}(\mathbf{x}_t) = a_t]}. \tag{1}$$

Note that the take-rate for a random policy which does not learn and takes actions randomly should hover around 10 percent. Show your cumulative take-rate $C(T)$ as a timeseries plot for $T = 2, \ldots, 10000$. Since linUCB has a parameter called α, try sweeping α in various ways to improve your take-rate. For instance, try various constant choices of α or choices that depend on t such as $\alpha = 1/\sqrt{t}$. Show at least 3 different strategies for choosing alpha and the resulting time series cumulative take-rate. Try to get the $C(T)$ to be as large as possible by choosing a formula for α that balances exploration and exploitation.

Upload all work to courseworks.columbia.edu. You can use ANY language to implement this homework. Please organize your code into separate files when appropriate to help us better understand it. The goal of the code and the writeup is to facilitate the understanding of the work so use your judgement there. Your write-up should be in Adobe Portable Document Format (.pdf).

Please do not submit Microsoft Office documents, LaTeX source code, or something more exotic since we will not be able to read it. LaTeX is preferred to generate your report and highly recommended, but it is not mandatory. You can use any document editing software you wish, as long as the final product is in .pdf. Even if you do not use LaTeX to prepare your document, you can use LaTeX notation to mark up complicated mathematical expressions, for example, in comments in your code. Please submit all your source files, each function in a separate file. Clearly denote what each function does, which problem it belongs to, and what the inputs and the outputs are. Do not resubmit code or data provided to you or that you downloaded. Do not submit code written by others, simply reference or cite it. Identical submissions will be detected and both parties will get zero credit. In general, shorter code is better. You may include figures directly in your write-up or include them separately as .jpg, .gif, .ps or .eps files, and refer to them by filename.

Optional Extra Work

Instead of the dataset above, also consider the dataset

`http://www.cs.columbia.edu/~jebara/6998/classification.txt`

We hunted around and found the original classification data-set (classification.txt) that was used to create the initial contextual bandit data-set (dataset.txt). Here, the first column is the true class c_t of the observation (a value from 1 to 10) followed by 100 features that define the context $\mathbf{x}_t$. Therefore, if your after seeing $\mathbf{x}_t$, your bandit selects the wrong action $a_t \neq c_t$ then it gets a reward of zero. Otherwise it gets a reward of 1. To be fair, you are not allowed to query the exact value of c_t but rather only query if your chosen action was equal to c_t. You should be able to run you LinUCB code on this problem in a straightforward manner. Be careful however, not to let your code cheat by using the c_t value directly since you would never be able to know c_t unless you tried all actions to see which one paid off.