Homework 3

CSE 599i: Interactive Learning
Instructor: Kevin Jamieson
Due 11:59 PM on March 12, 2021 (late homework not accepted)

Contextual Bandits

1. Problem 18.8 of [SzepesvariLattimore].

2. In this exercise we will implement several contextual bandit algorithms. We will "fake" a contextual bandit problem with multi-class classification dataset where each example is context, and the learner chooses an "action" among the available class labels, and receives a reward of 1 if the guess was correct, and 0 otherwise. However, keeping with bandit feedback, we assume the learner only knows the reward of the action played, not all actions.

We will use the MNIST dataset[1]. The MNIST dataset contains 28x28 images of handwritten digits from 0-9. Download this dataset and use the python-mnist library[2] to load it into Python. Rather than using the full images, you may run PCA on the data to come up with a lower dimensional representation of each image. You will have to experiment with what dimension, d, to use. Scale all images so that they are norm 1.

Let the d dimensional representation of the tth image in the dataset, c_t, be our "context." Our action set $\mathcal{X} = \{0, 1, \ldots, 9\}$ has 10 actions associated with each label. For each $i \in \mathcal{X} = \{0, 1, \ldots, 9\}$ define the feature map $\phi(c, i) = \text{vec}(c\mathbf{e}_i^\top) \in \mathbb{R}^{10d}$. If $v(c, x)$ is the expected reward of playing action $x \in \mathcal{X}$ in response to context c, then let us "model the world" with the simple linear model so that $v(c, x) \approx \langle \theta_*, \phi(c, x) \rangle$ for some unknown $\theta_* \in \mathbb{R}^{10d}$. Of course, when actually playing the game we will observe image features c_t, choose $x_t \in \mathcal{X}$, and receive reward $r_t = \mathbf{1}\{x_t = y_t\}$ where y_t is the true label of the image c_t and x_t is the action played.

Implement the Explore-Then-Commit algorithms, Follow-The-Leader, LinUCB, and Thompson Sampling algorithms for this problem. You can use just the training set of $T = 50000$ examples. Draw c_t uniformly at random with replacement from this set at each time for $t = 1, \ldots, T$. The algorithms work as follows:

- **Explore-Then-Commit** ("Model the world"): Fix $\tau \in [T]$. For the first τ steps, select each action $x \in \mathcal{X}$ uniformly at random. Compute $\widehat{\theta} = \arg\min_\theta \sum_{t=1}^{\tau} (r_t - \langle \phi(c_t, x_t), \theta \rangle)^2$. For $t > \tau$ play $x_t = \arg\max_{x \in \mathcal{X}} \langle \phi(c_t, x), \widehat{\theta} \rangle$. Choose a value of τ and justify it.
- **Explore-Then-Commit** ("Model the bias"): Fix $\tau \in [T]$. For the first τ steps, select each action $x \in \mathcal{X}$ uniformly at random. Note that here $p_t = 1/|\mathcal{X}$ for all t, so we can ignore the importance weights. Train a 10-class logistic classifier $\widehat{\pi} : \mathcal{C} \to \mathcal{X}$ on a dataset $\{(c_t, x_t)\}_{t \leq \tau : r_t = 1}$. That is, only include those labels in which the randomly chosen action was correct (note that this is equivalent to weighted-classification when $r_t \in \{0, 1\}$ and $p_t = 1/|\mathcal{X}|$). For $t > \tau$ play $x_t := \widehat{\pi}(c_t)$. Choose the same value of τ as "Model the world".
- **Follow-The-Leader**: Fix $\tau \in [T]$. For the first τ steps, select each action $x \in \mathcal{X}$ uniformly at random. For $t > \tau$ play $x_t = \arg\max_{x \in \mathcal{X}} \langle \phi(c_t, x), \widehat{\theta}_{t-1} \rangle$ where $\widehat{\theta}_t = \arg\min_\theta \sum_{s=1}^{t} (r_s - \langle \phi(c_s, x_s), \theta \rangle)^2$. Choose a value of τ and justify it.
- **LinUCB** Using Ridge regression with an appropriate $\gamma > 0$ ($\gamma = 1$ may be okay) construct the confidence set $\mathcal{C}_t$ derived in class (and in the book). At each time $t \in [T]$ play $x_t = \arg\max_{x \in \mathcal{X}} \max_{\theta \in \mathcal{C}_t} \langle \theta, \phi(c_t, x) \rangle$.
- **Thompson Sampling** Fix $\gamma > 0$ ($\gamma = 1$ may be okay). At time $t \in [T]$ draw $\widetilde{\theta}_t \sim \mathcal{N}(\widehat{\theta}_{t-1}, V_{t-1}^{-1})$ and play $x_t = \arg\max_{x \in \mathcal{X}} \langle \widetilde{\theta}_t, \phi(c_t, x) \rangle$ where $\widehat{\theta}_t = \arg\min_\theta \sum_{s=1}^{t} (r_s - \langle \theta, \phi(c_s, x_s) \rangle)^2$ and $V_t = \gamma I + \sum_{s=1}^{t} \phi(c_s, x_s)\phi(c_s, x_s)^\top$.

Implement each of these algorithms and show a plot of the regret (all algorithms on one plot) when run on MNIST for good choices of τ, γ. Hint, for computing V_t^{-1} efficiently see `https://en.wikipedia.org/wiki/Sherman%E2%80%93Morrison_formula`.

[1] `http://yann.lecun.com/exdb/mnist/`
[2] `https://pypi.org/project/python-mnist/`