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. The MNIST dataset contains 28x28 images of handwritten digits from 0-9. Download this dataset and use the python-mnist library 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 \(t\)th image in the dataset, \(c_t\), be our “context.” Our action set \(\mathcal{A} = \{0, 1, \ldots, 9\}\) has 10 actions associated with each label. For each \(i \in \mathcal{A} = \{0, 1, \ldots, 9\}\) define the feature map \(\phi(c, i) = \text{vec}(ce_i^\top) \in \mathbb{R}^{10d}\). If \(v(c, a)\) is the expected reward of playing action \(a \in \mathcal{A}\) in response to context \(c\), then let us “model the world” with the simple linear model so that \(v(c, a) \approx (\theta_a, \phi(c, a))\) for some unknown \(\theta_a \in \mathbb{R}^{10d}\). Of course, when actually playing the game we will observe image features \(c_t\) as the context, choose an “action” \(a_t \in \{0, \ldots, 9\}\), and receive reward \(r_t = 1\{a_t = y_t\}\) where \(y_t\) is the true label of the image \(c_t\) and \(a_t\) 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. The training set is class balanced meaning that there are 5000 examples of each digit. Important: randomly shuffle the dataset so the probability of any particular class showing up at any given time is \(1/10\). The algorithms work as follows:

- **Explore-Then-Commit** (“Model the world”): Fix \(\tau \in [T]\). For the first \(\tau\) steps, select each action \(a \in \mathcal{A}\) uniformly at random. Compute \(\hat{\theta} = \arg \min_{\theta} \sum_{t=1}^{\tau} (r_t - \langle \phi(c_t, a_t), \theta \rangle)^2\). For \(t > \tau\) play \(a_t = \arg \max_{a \in \mathcal{A}} \langle \phi(c_t, a), \hat{\theta} \rangle\). Choose a value of \(\tau\) and justify it.

- **Explore-Then-Commit** (“Model the bias”): Fix \(\tau \in [T]\). For the first \(\tau\) steps, select each action \(a \in \mathcal{A}\) uniformly at random. Our goal is to identify a policy \(\hat{\pi} : \mathcal{C} \rightarrow \mathcal{A}\) using the dataset \(\{(c_t, a_t, p_t, r_t)\}_{t \leq \tau}\) such that

\[
\hat{\pi} = \arg \max_{\pi \in \Pi} \sum_{t=1}^{\tau} r_t \frac{1\{\pi(c_t) = a_t\}}{p_t}
\]

\[
= \arg \min_{\pi \in \Pi} \sum_{t=1}^{\tau} r_t \frac{1\{\pi(c_t) \neq a_t\}}{p_t}
\]

\[
= \arg \min_{\pi \in \Pi} \sum_{t \in [\tau]; r_t = 1} 1\{\pi(c_t) \neq a_t\}
\]

where the last line uses the fact that \(p_t = 1/10\) due to uniform exploration and the definition of \(r_t\). Note that this is just a multi-class classification problem on dataset \(\{(c_t, a_t)_{t \in [\tau]; r_t = 1}\}\) where one is trying to identify a classifier \(\hat{\pi} : \mathcal{C} \rightarrow \mathcal{A}\) that predicts label \(a_t\) from features \(c_t\). Train a 10-class linear logistic classifier \(\hat{\pi}\) on the data up to time \(\tau\) and then for \(t > \tau\) play \(a_t = \arg \max_{a \in \{0, \ldots, 9\}} \hat{\pi}(c_t)\).

Choose the same value of \(\tau\) as “Model the world”.

---

2. [https://pypi.org/project/python-mnist/](https://pypi.org/project/python-mnist/)
• **Follow-The-Leader**: Fix $\tau \in [T]$. For the first $\tau$ steps, select each action $a \in A$ uniformly at random. For $t > \tau$ play $a_t = \arg \max_{a \in A} \langle \phi(c_t, a), \hat{\theta}_{t-1} \rangle$ where $\hat{\theta}_t = \arg \min_{\theta} \sum_{s=1}^t (r_s - \langle \phi(c_s, a_s), \theta \rangle)^2$. Choose a value of $\tau$ and justify it.

• **LinUCB** Using Ridge regression with an appropriate $\gamma > 0$ ($\gamma = 1$ may be okay) construct the confidence set $C_t$ derived in class (and in the book). At each time $t \in [T]$ play $a_t = \arg \max_{a \in A} \max_{\theta \in C_t} \langle \theta, \phi(c_t, a) \rangle$.

• **Thompson Sampling** Fix $\gamma > 0$ ($\gamma = 1$ may be okay). At time $t \in [T]$ draw $\tilde{\theta}_t \sim N(\hat{\theta}_{t-1}, V_t^{-1})$ and play $a_t = \arg \max_{a \in A} \langle \tilde{\theta}_t, \phi(c_t, a) \rangle$ where $\hat{\theta}_t = \arg \min_{\theta} \sum_{s=1}^t (r_s - \langle \theta, \phi(c_s, a_s) \rangle)^2$ and $V_t = \gamma I + \sum_{s=1}^t \phi(c_s, a_s) \phi(c_s, a_s)^\top$.

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 $\tau, \gamma$. Hint, for computing $V_t^{-1}$ efficiently see [https://en.wikipedia.org/wiki/Sherman%E2%80%93Morrison_formula](https://en.wikipedia.org/wiki/Sherman%E2%80%93Morrison_formula).