2023Fall Probability & Statistics for EECS 2023/12/08

Final Project Option 1

Professor: Ziyu Shao & Dingzhu Wen *Due: 2024/1/21 10:59pm*

This is the Final Project Option 1 entitled "Performance Evaluation of Bandit Learning Algorithms", which has two parts. Part I is **classical bandit algorithms**, and Part II is **Bayesian bandit algorithms**. This project can be done by a team with no more than three students. Your team is required to use Python for the programming part. Your team needs to submit all things in Jupyter Notebook format, including Python codes, simulation results, analysis, discussions, tables, figures, *etc.* Always keep the **academic integrity** in mind and remember to give credit to any source that inspires you.

- **Part I: Classical Bandit Algorithms**

  We consider a time-slotted bandit system ($t = 1, 2, \ldots$) with three arms. We denote the arm set as $\{1, 2, 3\}$. Pulling each arm j ($j \in \{1, 2, 3\}$) will obtain a random reward r_j, which follows a Bernoulli distribution with mean θ_j, *i.e.*, Bern(θ_j). Specifically,

  $$r_j = \begin{cases} 1, & w.p.\ \theta_j, \\ 0, & w.p.\ 1 - \theta_j, \end{cases}$$

  where $\theta_j, j \in \{1, 2, 3\}$ are parameters within $(0, 1)$.

  Now we run this bandit system for N ($N \gg 3$) time slots. In each time slot t, we choose one and only one arm from these three arms, which we denote as $I(t) \in \{1, 2, 3\}$. Then we pull the arm $I(t)$ and obtain a random reward $r_{I(t)}$. Our objective is to find an optimal policy to choose an arm $I(t)$ in each time slot t such that the expectation of the aggregated reward over N time slots is maximized, *i.e.*,

  $$\max_{I(t), t=1,\ldots,N} \quad \mathbb{E}\left[\sum_{t=1}^{N} r_{I(t)}\right].$$

  If we know the values of $\theta_j, j \in \{1, 2, 3\}$, this problem is trivial. Since $r_{I(t)} \sim \text{Bern}(\theta_{I(t)})$,

  $$\mathbb{E}\left[\sum_{t=1}^{N} r_{I(t)}\right] = \sum_{t=1}^{N} \mathbb{E}[r_{I(t)}] = \sum_{t=1}^{N} \theta_{I(t)}.$$

  Let $I(t) = I^* = \arg\max_{j \in \{1,2,3\}} \theta_j$ for $t = 1, 2, \ldots, N$, then

  $$\max_{I(t), t=1,\ldots,N} \quad \mathbb{E}\left[\sum_{t=1}^{N} r_{I(t)}\right] = N \cdot \theta_{I^*}.$$

However, in reality, we do not know the values of $\theta_j, j \in \{1,2,3\}$. We need to estimate the values $\theta_j, j \in \{1,2,3\}$ via empirical samples, and then make the decisions in each time slot. Next we introduce three classical bandit algorithms: ϵ-greedy, UCB, and TS, respectively.

(1). ϵ-greedy Algorithm ($0 \leq \epsilon \leq 1$)

Algorithm 1 ϵ-greedy Algorithm

Initialize $\widehat{\theta}(j) \leftarrow 0, \text{count}(j) \leftarrow 0, j \in \{1,2,3\}$

1: **for** $t = 1, 2 \ldots, N$ **do**

2:

$$I(t) \leftarrow \begin{cases} \underset{j \in \{1,2,3\}}{\arg\max}\ \widehat{\theta}(j) & w.p.\ 1-\epsilon \\ \text{randomly chosen from} \{1,2,3\} & w.p.\ \epsilon \end{cases}$$

3: $\text{count}(I(t)) \leftarrow \text{count}(I(t)) + 1$

4: $\widehat{\theta}(I(t)) \leftarrow \widehat{\theta}(I(t)) + \frac{1}{\text{count}(I(t))} \left[r_{I(t)} - \widehat{\theta}(I(t)) \right]$

5: **end for**

(2). UCB (Upper Confidence Bound) Algorithm

Algorithm 2 UCB Algorithm

1: **for** $t = 1, 2, 3$ **do**

2: $I(t) \leftarrow t$

3: $\text{count}(I(t)) \leftarrow 1$

4: $\widehat{\theta}(I(t)) \leftarrow r_{I(t)}$

5: **end for**

6: **for** $t = 4, \ldots, N$ **do**

7:

$$I(t) \leftarrow \underset{j \in \{1,2,3\}}{\arg\max} \left(\widehat{\theta}(j) + c \cdot \sqrt{\frac{2\log(t)}{\text{count}(j)}} \right)$$

8: $\text{count}(I(t)) \leftarrow \text{count}(I(t)) + 1$

9: $\widehat{\theta}(I(t)) \leftarrow \widehat{\theta}(I(t)) + \frac{1}{\text{count}(I(t))} \left[r_{I(t)} - \widehat{\theta}(I(t)) \right]$

10: **end for**

Note: c is a positive constant with a default value of 1.

(3). TS (Thompson Sampling) Algorithm

Recall that $\theta_j, j \in \{1, 2, 3\}$ are unknown parameters over $(0, 1)$. From the Bayesian perspective, we assume their priors are Beta distributions with parameters (α_j, β_j).

Algorithm 3 TS Algorithm

Initialize Beta parameter $(\alpha_j, \beta_j), j \in \{1, 2, 3\}$

1: **for** $t = 1, 2 \ldots, N$ **do**
2: *# Sample model*
3: **for** $j \in \{1, 2, 3\}$ **do**
4: Sample $\widehat{\theta}(j) \sim \text{Beta}(\alpha_j, \beta_j)$
5: **end for**
6: *# Select and pull the arm*

$$I(t) \leftarrow \arg\max_{j \in \{1,2,3\}} \widehat{\theta}(j)$$

7: *# Update the distribution*

$$\alpha_{I(t)} \leftarrow \alpha_{I(t)} + r_{I(t)}$$
$$\beta_{I(t)} \leftarrow \beta_{I(t)} + 1 - r_{I(t)}$$

8: **end for**

1. Now suppose we obtain the parameters of the Bernoulli distributions from an oracle, which are shown in the following table. Choose $N = 5000$ and compute the theoretically maximized expectation of aggregate rewards over N time slots. We call it the oracle value. Note that these parameters $\theta_j, j \in \{1, 2, 3\}$ and oracle values are unknown to all bandit algorithms.

Arm j	1	2	3
θ_j	0.7	0.5	0.4

2. Implement classical bandit algorithms with following settings:
   - $N = 5000$
   - ϵ-greedy with $\epsilon \in \{0.1, 0.5, 0.9\}$.
   - UCB with $c \in \{1, 5, 10\}$.
   - TS with $\{(\alpha_1, \beta_1) = (1, 1), (\alpha_2, \beta_2) = (1, 1), (\alpha_3, \beta_3) = (1, 1)\}$ and $\{(\alpha_1, \beta_1) = (601, 401), (\alpha_2, \beta_2) = (401, 601), (\alpha_3, \beta_3) = (2, 3)\}$
3. Each experiment lasts for $N = 5000$ time slots, and we run each experiment 200 trials. Results are averaged over these 200 independent trials.

4. Compute the gaps between the algorithm outputs (aggregated rewards over N time slots) and the oracle value. Compare the numerical results of ϵ-greedy, UCB, and TS. Which one is the best? Then discuss the impacts of ϵ, c, and α_j, β_j, respectively.
5. Give your understanding of the exploration-exploitation trade-off in bandit algorithms.
6. We implicitly assume the reward distribution of these three arms are independent. How about the dependent case? Can you design an algorithm to exploit such information to obtain a better result?

- **Part II: Bayesian Bandit Algorithms**

  There are two arms which may be pulled repeatedly in any order. Each pull may result in either a success or a failure. The sequence of successes and failures which results from pulling arm i ($i \in \{1, 2\}$) forms a Bernoulli process with unknown success probability θ_i. A success at the t^{th} pull yields a reward γ^{t-1} ($0 < \gamma < 1$), while an unsuccessful pull yields a zero reward. At time zero, each θ_i has a Beta prior distribution with two parameters α_i, β_i and these distributions are independent for different arms. These prior distributions are updated to posterior distributions as arms are pulled. Since the class of Beta distributions is closed under Bernoulli sampling, posterior distributions are all Beta distributions. How should the arm to pull next in each time slot be chosen to maximize the total expected reward from an infinite sequence of pulls?

  1. One intuitive policy suggests that in each time slot we should pull the arm for which the current expected value of θ_i is the largest. This policy behaves very good in most cases. Please design simulations to check the behavior of this policy.
  2. However, such intuitive policy is unfortunately not optimal. Please provide an example to show why such policy is not optimal.
  3. For the expected total reward under an optimal policy, show that the following recurrence equation holds:

$$\begin{aligned} R_1(\alpha_1, \beta_1) =& \frac{\alpha_1}{\alpha_1 + \beta_1}[1 + \gamma R(\alpha_1 + 1, \beta_1, \alpha_2, \beta_2)] \\ &+ \frac{\beta_1}{\alpha_1 + \beta_1}[\gamma R(\alpha_1, \beta_1 + 1, \alpha_2, \beta_2)]; \\ R_2(\alpha_2, \beta_2) =& \frac{\alpha_2}{\alpha_2 + \beta_2}[1 + \gamma R(\alpha_1, \beta_1, \alpha_2 + 1, \beta_2)] \\ &+ \frac{\beta_2}{\alpha_2 + \beta_2}[\gamma R(\alpha_1, \beta_1, \alpha_2, \beta_2 + 1)]; \\ R(\alpha_1, \beta_1, \alpha_2, \beta_2) =& \max\left\{R_1(\alpha_1, \beta_1), R_2(\alpha_2, \beta_2)\right\}. \end{aligned}$$

  4. For the above equations, how to solve it exactly or approximately?
  5. Find the optimal policy.