Homework 1

6501 Reinforcement Learning (Fall 2025)

Deadline: 11:59pm, September 28, 2025

You may type or handwrite your solution. If handwritten, take photos and include them as figures in latex. Submit the answer as a .pdf and the code (for Problem 3) as a .py file to Gradescope.

The latex template can be accessed at `https://www.overleaf.com/read/xhpccjwhbcjs#f9dad9`

1 ϵ-greedy for Contextual Bandits

In this problem, we will derive the regret bound of ϵ-greedy in contextual bandits with a regression oracle (Page 41 here). Consider the algorithm below.

```
Algorithm 1 $\epsilon$-Greedy for contextual bandits
Parameter: $\epsilon \in [0, 1]$, $A$ (number of actions)
Given: A regression oracle
for $t = 1, 2, \ldots, T$ do
    Receive $x_t$, and obtain $\hat{R}_t$ from the regression oracle.
    Define
        $\pi_t(a) = \begin{cases} 1 - \epsilon + \frac{\epsilon}{A} & \text{if } a = \text{argmax}_{a'} \hat{R}_t(x_t, a') \\ \frac{\epsilon}{A} & \text{otherwise} \end{cases}$
    Sample $a_t \sim \pi_t$, and receive $r_t = R(x_t, a_t) + w_t$, where $R$ is the underlying true reward function and $w_t$ is a zero-mean noise.
```

Define $a_t^\star = \text{argmax}_a R(x_t, a)$. Assume that $R(x, a) \in [0, 1]$ and $\hat{R}_t(x, a) \in [0, 1]$ for any x, a, t.

(a) (5%) Show the following inequality. Note that the left-hand side is the expected regret at round t, and the right-hand side is ϵ plus the estimation error of the regression oracle.

$$R(x_t, a_t^\star) - \mathbb{E}_{a\sim\pi_t}\left[R(x_t, a)\right] \leq \epsilon + \underbrace{\mathbb{E}_{a\sim\pi_t}\left[\hat{R}_t(x_t, a) - R(x_t, a)\right]}_{\text{estimation error on } a_t} + \underbrace{R(x_t, a_t^\star) - \hat{R}_t(x_t, a_t^\star)}_{\text{estimation error on } a_t^\star}.$$

(b) (5%) Show that the two estimation error terms in (a) can be bounded as

$$\mathbb{E}_{a\sim\pi_t}\left[\hat{R}_t(x_t, a) - R(x_t, a)\right] \leq \sqrt{\mathbb{E}_{a\sim\pi_t}\left[\left(\hat{R}_t(x_t, a) - R(x_t, a)\right)^2\right]},$$

$$R(x_t, a_t^\star) - \hat{R}_t(x_t, a_t^\star) \leq \sqrt{\frac{1}{\pi_t(a_t^\star)}\mathbb{E}_{a\sim\pi_t}\left[\left(\hat{R}_t(x_t, a) - R(x_t, a)\right)^2\right]},$$

respectively.

(c) (5%) Combining (a) and (b), show that the one-step expected regret at round t can be upper bounded as

$$R(x_t, a_t^\star) - \mathbb{E}_{a\sim p_t}\left[R(x_t, a)\right] \le \epsilon + 2\sqrt{\frac{A}{\epsilon} \cdot \mathbb{E}_{a\sim\pi_t}\left[\left(\hat{R}_t(x_t, a) - R(x_t, a)\right)^2\right]}.$$

(Hint: note that $\pi_t(a_t^\star) \ge \frac{\epsilon}{A}$)

(d) (5%) Show that expected total regret $\mathbb{E}\left[\sum_{t=1}^{T}\left(R(x_t, a_t^\star) - R(x_t, a_t)\right)\right]$ can be upper bounded by

$$\epsilon T + 2\sqrt{\frac{AT\mathbb{E}\left[\text{Err}\right]}{\epsilon}},$$

where Err is the total regression error defined as

$$\text{Err} = \sum_{t=1}^{T} \mathbb{E}_{a\sim\pi_t}\left[\left(\hat{R}_t(x_t, a) - R(x_t, a)\right)^2\right].$$

2 $\tilde{O}(\sqrt{AT})$ Regret Bound for UCB

In this problem, we will show that UCB for multi-armed bandits ensures a $\tilde{O}(\sqrt{AT})$ regret (Page 49 here). We first define the following quantities: Let $R(a) \in [0, 1]$ be the true mean of the reward of arm a. Define $a^\star \triangleq \text{argmax}_{a\in[A]} R(a)$.

We consider the UCB algorithm described in Algorithm 2. Assume that the number of arms A is less or equal to the number of rounds T, and assume w_t is zero-mean and 1-sub-Gaussian.

```
Algorithm 2 UCB for multi-armed bandits
Input: $A$ (number of arms), $T$ (total number of rounds), $\delta$ (failure probability).
for $t = 1, \dots, A$ do
    Draw $a_t = t$ and observe $r_t = R(a_t) + w_t$.
for $t = A + 1, \dots, T$ do
    Define
    $N_t(a) = \sum_{s=1}^{t-1} \mathbb{I}\{a_s = a\}$,   $\hat{R}_t(a) = \frac{\sum_{s=1}^{t-1} \mathbb{I}\{a_s = a\} r_s}{N_t(a)}$,   $\textit{conf}_t(a) = \sqrt{\frac{2\log(2/\delta)}{N_t(a)}}$,   $\tilde{R}_t(a) = \hat{R}_t(a) + \textit{conf}_t(a)$.
    Draw $a_t = \text{argmax}_a \tilde{R}_t(a)$ and observe $r_t = R(a_t) + w_t$.
```

Recall that the regret is defined as Regret $= TR(a^\star) - \sum_{t=1}^{T} R(a_t)$.

(a) (5%) Use Theorem 1 to show that with probability $1 - AT\delta$, $|\hat{R}_t(a) - R(a)| \le \textit{conf}_t(a)$ for all time t and arm a.

(b) (5%) Assume the inequality in (a) holds for all t and a. Prove that for all $t > A$:

$$R(a^\star) - R(a_t) \le 2\textit{conf}_t(a_t).$$

(Hint 1: Decompose the left-hand side as $\left(\tilde{R}_t(a^\star) - \tilde{R}(a_t)\right) + \left(\tilde{R}_t(a_t) - R(a_t)\right) + \left(R(a^\star) - \tilde{R}_t(a^\star)\right)$)
(Hint 2: Bound the second and the third parts in Hint 1 with the help of (a))

(c) (5%) Show the inequality:

$$\sum_{t=A+1}^{T} \textit{conf}_t(a_t) \le 2\sqrt{2\log(2/\delta)AT}.$$

(Hint 1: Write the left -hand side as $\sum_{a=1}^{A}\sum_{t=A+1}^{T}\mathbb{I}\{a_t = a\}conf_t(a)$)
(Hint 2: Use the inequality $1+\sqrt{\frac{1}{2}}+\sqrt{\frac{1}{3}}+\cdots+\sqrt{\frac{1}{m}} \le 2\sqrt{m}$)
(Hint 3: Use the Cauchy-Schwarz inequality $\sqrt{m_1}+\sqrt{m_2}+\cdots+\sqrt{m_A} \le \sqrt{A(m_1+m_2+\cdots+m_A)}$)

(d) (5%) Conclude that with probability at least $1-AT\delta$, Regret $\le A+4\sqrt{2\log(2/\delta)AT}$.

3 Implementing Contextual Bandit Algorithms

In this homework, we will implement value-based contextual bandit algorithms covered in the lecture. Specifically, we will implement

- Algorithms based on regression oracle and exploration mechanisms including ϵ-greedy (EG), Boltzmann exploration (BE), and inverse gap weighting (IGW).

The starter code can be accessed at `http://bahh723.github.io/rl2025fa_files/bandits.py`.

Note: Please maintain your code nicely, as we will implement more algorithms in the same python file in HW2.

3.1 Data

To simulate a contextual bandit environment, we use existing classification dataset for supervised learning. Specifically, in this homework, we use the mnist dataset with some pruning and modification. Originally, it is a classification dataset where features are 28 pixel × 28 pixel gray-scale images of digits, and the classes correspond to 10 digits.

3.2 Data Conversion

For simplicity, we trim the dataset so that it only contains 4 classes corresponding to digits '0', '1', '2', and '3', each having 2500 samples, summing up to 10000 samples. In each round, the environment will reveal the image (context), and the learner has to pick one of the classes (action). We artificially make the reward function change once in the middle. Specifically, in our case the time horizon is $T = 10000$. For $t \le 5000$ (Phase 1), the reward function is the following:

$$R(x,a) = \begin{cases} 0.5 & \text{if } a \text{ is the correct digit of image } x \\ 0 & \text{otherwise} \end{cases}$$

For $t > 5000$ (Phase 2), the reward is the following:

$$R(x,a) = \begin{cases} 0.5 & \text{if } a \text{ is the correct digit of image } x \\ 1 & \text{if } (a-1) \bmod 4 \text{ is the is the correct digit of image } x \\ 0 & \text{otherwise} \end{cases}$$

For example, in Phase 2, if the learner chooses action 3 when seeing an image of digit 2, then it receives a reward of 1. We assume that the learner sees the true noiseless reward $R(x_t, a_t)$.

This conversion is already implemented in the starter code, so you don't need to do anything here.

3.3 CB Algorithm based on regression oracle

In this part, we will implement the simple value-based algorithm based on regression on the reward function. In the class, we have introduce three common exploration mechanism in this case: ϵ-Greedy, Boltzmann Exploration, and Inverse Gap Weighting. They share the same pseudo-code outlined in Algorithm 3.

Algorithm 3 An algorithm based on regression

```
Randomly initialize a reward network $R_\theta$ that takes the context as input and outputs the reward of each action.
Let $\theta_1$ be the initial weights for the reward network.
for $t = 1, \dots, T$ do
    for $n = 1, \dots, N$ do
        Receive context $x_{t,n}$.
        Sample action $a_{t,n} \sim \pi(\cdot \mid x_{t,n})$ where $\pi(\cdot \mid x) = f\left(R_{\theta_t}(x, \cdot)\right)$
        Receive reward $r_{t,n}$.
    $\theta \leftarrow \theta_t$
    for $m = 1, \dots, M$ do
        $\theta \leftarrow \theta - \lambda \nabla_\theta \left( \frac{1}{N} \sum_{n=1}^{N} \left(R_\theta(x_{t,n}, a_{t,n}) - r_{t,n}\right)^2 \right).$    (1)
    $\theta_{t+1} \leftarrow \theta$
```

In Algorithm 3, $f : \mathbb{R}^A \to \Delta_A$ is a link function that maps an A-dimensional real vector to a distribution over A actions. The three exploration mechanisms correspond to the following three choices of f:

- ϵ-Greedy: $[f(v)]_a = \frac{\epsilon}{A} + (1-\epsilon)\mathbb{I}\{a = \operatorname{argmax}_{a'} v(a')\}$.
- Boltzmann Exploration: $[f(v)]_a = \frac{e^{\lambda v(a)}}{\sum_{a'} e^{\lambda v(a')}}$.
- Inverse Gap Weighting: $[f(v)]_a = \begin{cases} \frac{1}{A + \lambda(\max_{a'} v(a') - v(a))} & \text{if } a \neq \operatorname{argmax}_{a'} v(a') \\ 1 - \sum_{a' \neq a} [f(v)]_{a'} & \text{if } a = \operatorname{argmax}_{a'} v(a') \end{cases}$.

Algorithm 3 may be slightly different from the standard way of implementing such algorithm — usually, in Eq. (1) we may reuse data collected in the past by having a *replay buffer* that stores old data. But since our artificial problem has a non-stationary reward function, reusing data from the past becomes not a great idea.

Please complete the following tasks in (a)-(g). Note that the starter code already calculate the average reward in Phase 1, Phase 2, and overall horizon, respectively. The starter code also already implement the regression oracle (1). A figure of running average is also generated for a single parameter. The major task is to code up the three link functions above, run the experiments, and combine the figures for multiple parameters.

Note that you are allowed to change the default hyperparameters in the starter code (N, M, optimizer, learning rates, etc.). In the table below, you may also change the values of hyperparameters or add additional ones if you feel that the given values cannot reflect the trend.

The starter code already implements two baseline methods: one uniformly randomly chooses an action, and the other greedily chooses an action with the highest estimated reward. Run them with the command line:

```
python bandits.py --algorithm Rand
python bandits.py --algorithm Greedy
```

For each method, a correct implementation must achieve an "overall reward" of at least 0.6 under the *best choice* of the hyperparameter. Meeting this requirement is necessary to get full credit.

(a) (5%) Implement ϵ-Greedy and, for different values of ϵ, record in the table below the average reward in Phase 1, Phase 2, and over the entire horizon.

ϵ	Phase 1	Phase 2	Overall
0.1			
0.03			
0.01			
0.003			
0.001			
0			

(b) (5%) Plot the running average reward over time for every parameter setting from part (a) on the same figure, following the example provided in the appendix.

(c) (5%) Implement Boltzmann Exploration and, for different values of λ, record in the table below the average reward in Phase 1, Phase 2, and over the entire horizon.

λ	Phase 1	Phase 2	Overall
2			
5			
10			
20			
50			

(d) (5%) Plot the running average reward over time for every parameter setting from part (c) on the same figure.

(e) (5%) Implement Inverse Gap Weighting and, for different values of λ, record in the table below the average reward in Phase 1, Phase 2, and over the entire horizon.

λ	Phase 1	Phase 2	Overall
10			
30			
100			
300			
1000			

(f) (5%) Plot the running average reward over time for every parameter setting from part (e) on the same figure.

(g) (5%) Have you noticed any trends in the experiments from (a)–(f)? In particular, how does the parameter influence the average reward in Phase 1 and Phase 2, respectively? What is the underlying explanation for this effect?

4 Survey

(a) (5%) How much time did you use to complete each of the problems in this homework? Do you have any suggestion for the course?

Appendix

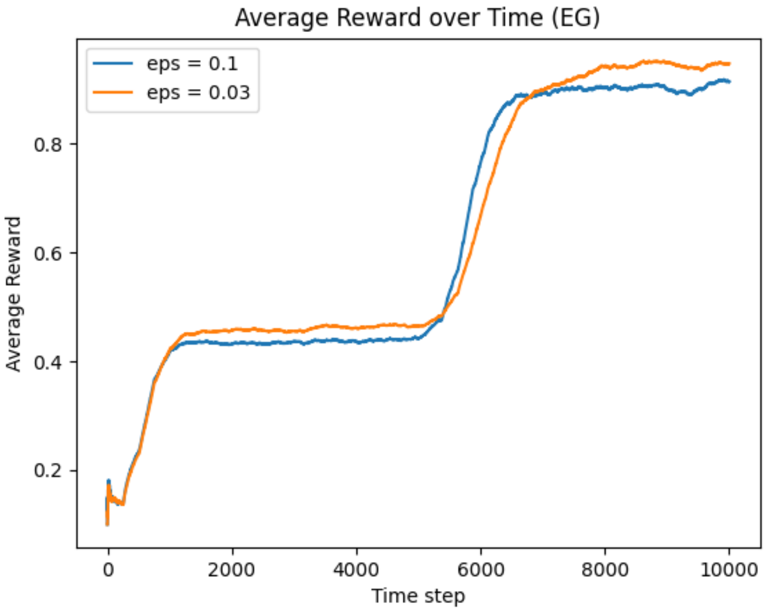
A Inequality

Theorem 1 (Hoeffding's Inequality)**.** *Let $X_1, X_2, \ldots, X_N$ be i.i.d. σ-sub-Gaussian random variables with mean μ. Then with probability at least $1-\delta$,*

$$\left| \frac{1}{N} \sum_{i=1}^{N} X_i - \mu \right| \le \sigma \sqrt{\frac{2\log(2/\delta)}{N}}.$$

B Plot

In the starter code, we plot the learning curve for a single parameter. You need to plot the learning curves for multiple parameter values on the same figure. Below is an example that shows how to plot two curves, although in usually need to plot 5-6 curves in the problems above.


Average Reward over Time (EG)
eps = 0.1
eps = 0.03
Average Reward
Time step