CS229T/STATS231: Statistical Learning Theory

Lecturer: Tengyu Ma
Scribe: William McCloskey

Lecture 18
November 28, 2018

1 Review and Overview

Last lecture we studied the multi-armed bandit problem in the oblivious setting. Recall that the multi-armed bandit problem is set up as follows:

- There are N experts and time steps $t = 1, \ldots, T$.
- At each time step t, the player picks an action $a_t \in [N]$ and receives loss $\ell_t(a_t) \in [0, 1]$.

In the oblivious setting, the losses ℓ_t are chosen before the game starts. Notice that, in contrast to the expert problem, we only observe component of the loss $\ell_t(a_t)$ rather than the whole loss vector ℓ_t.

In this lecture, we consider the stochastic setting. In the stochastic setting, the losses $\ell_t \in [0, 1]^N$ are drawn i.i.d. from some distribution P. Thus, the player's job is easier than in the oblivious setting.

We will seek to minimize the regret

$$\text{Regret}_{\text{stoch}} \triangleq \mathop{\mathbb{E}}_{\ell_t, \mathcal{A}} \left(\sum_{t=1}^{T} \ell_t(a_t) - \sum_{t=1}^{T} \ell_t(a^*) \right)$$

where a^* is the best possible action in terms of expected loss we could have taken at the start, i.e.

$$a^* \triangleq \mathop{\text{argmin}}_{a \in [N]} \mathop{\mathbb{E}}_{\ell_t \sim P} \left(\sum_{t=1}^{T} \ell_t(a) \right).$$

In the regret, the expectation is over the randomness loss vectors ℓ_t and the algorithm $\mathcal{A}$ used to play the actions a_t. Also notice that, unlike in the oblivious setting, the sum $\sum_{t=1}^{T} \ell_t(a^*)$ is inside the expectation.

We will cover two algorithms to minimize the regret in the multi-armed bandit stochastic setting. The algorithms seek a good balance to the explore-exploit tradeoff. Recall that these heuristics are:

- **Explore:** Each action i a player takes gives information about the distribution $\ell_t(i)$. By exploring a variety of actions in $[N]$, the player gains more information about the distribution P of the loss vectors ℓ_t.
- **Exploit:** The losses $\ell_1(a_1), \ldots, \ell_{t-1}(a_{t-1})$ give the player information about the distribution P of the loss vector ℓ_t. A player can exploit this information to play an estimate a_t of the best expert.

The first algorithm is a simple algorithm called Explore-Then-Exploit. Essentially, the player explores actions uniformly for the first T_0 time steps and exploits this information for the rest of the game. We prove a regret bound for Explore-Then-Exploit that is $O(\log(T))$ for appropriate choice of T_0. Usually, we can only hope for regret bounds that are $O(T^{1/2})$, and in the oblivious setting we only managed to show $O(T^{3/4})$. The performance of this simple algorithm shows how much easier the stochastic setting is.

We then introduce a more sophisticated algorithm called Upper Confidence Bound (UCB). This algorithm builds confidence intervals around estimates for $\mathbb{E}(\ell_t(i))$ where $i = 1, \ldots, N$ and plays an expert optimistically. We will see that UCB strikes a better balance between the exploring and exploiting than the first algorithm.

2 Notation

For completeness, we record the notation from the first section. We have N experts, actions $a_t \in [N]$, and loss vectors $\ell_t \in [0,1]^N$ where $\ell_t \overset{iid}{\sim} P$ for $t = 1, \ldots, T$. The player seeks to minimize the regret

$$\text{Regret}_{\text{stoch}} = \mathop{\mathbb{E}}_{\ell_t, \mathcal{A}} \left(\sum_{t=1}^{T} \ell_t(a_t) - \sum_{t=1}^{T} \ell_t(a^*) \right).$$

We will denote by μ the mean of the loss vector $\ell \sim P$, by $\hat{\mu}_t$ the player's estimate of μ at time t, and by $n_t(a)$ the number of times action a is played during the first t iterations:

$$\begin{aligned}
\mu(a) &\triangleq \mathop{\mathbb{E}}_{\ell \sim P} (\ell(a)) \\
n_t(a) &\triangleq \sum_{i=1}^{t} \mathbb{1}[a_i = a] \\
\hat{\mu}_t(a) &\triangleq \frac{1}{n_t(a)} \sum_{i=1}^{t} \ell_i(a) \mathbb{1}[a_i = a].
\end{aligned}$$

(In both algorithms below, we can assume that each action is played once during the first N time steps so that $\hat{\mu}_t(a)$ is well-defined.) Observe that $a^* = \mathop{\text{argmin}}_{a \in [N]} \mu(a)$.

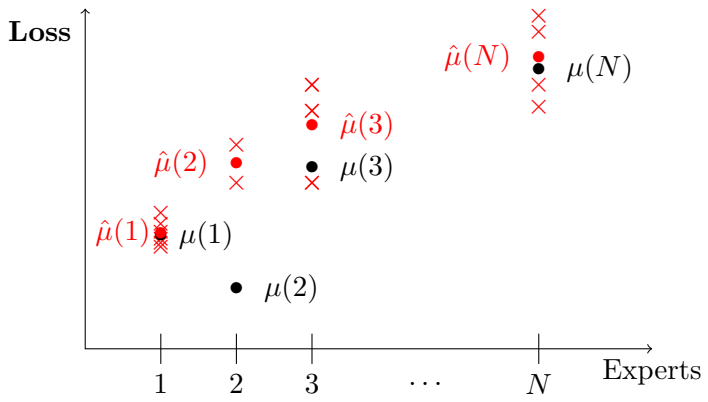

Figure 1: A game at stage T_0. Here the $\times$ signs are the losses $\ell_t(a_t)$. The expected loss μ and sample expected loss $\hat{\mu}_{T_0}$ are also plotted. The best action a^* for the game is $a^* = 2$.

3 First Attempt: The Explore-Then-Exploit Algorithm

In this section, we introduce a baseline algorithm called Explore-Then-Exploit. We then prove that it has a logarithmic bound on the regret under certain conditions.

Explore-Then-Exploit Algorithm
Before the game starts, pick an exploration time T_0. For the first T_0 iterations the player explores, and for the rest of the game the player exploits.

(*Explore*) For $t = 1, \ldots, T_0$, play each action in $[N]$ a total of T_0/N times.

(*Exploit*) For $t = T_0 + 1, \ldots, N$, play the best action according to the estimate $\hat{\mu}_{T_0}$. That is, play the action $\hat{a} = \operatorname{argmin}_{a \in [N]} \hat{\mu}_{T_0}(a)$

In order to choose T_0 and bound the regret for the Explore-Then-Exploit algorithm, we need a few more notations and a lemma. Our bounds and choice of T_0 will depend on how suboptimal each action is in comparison to the best action. To this end, we define quantities Δ_a and Δ.

$$\Delta_a \triangleq \mu(a) - \mu(a^*)$$
$$\Delta \triangleq \min_{a:\Delta_a > 0} \Delta_a.$$

Essentially, Δ_a is the gap in expected loss between a and the best action a^*, and Δ is the gap between second best and best actions.

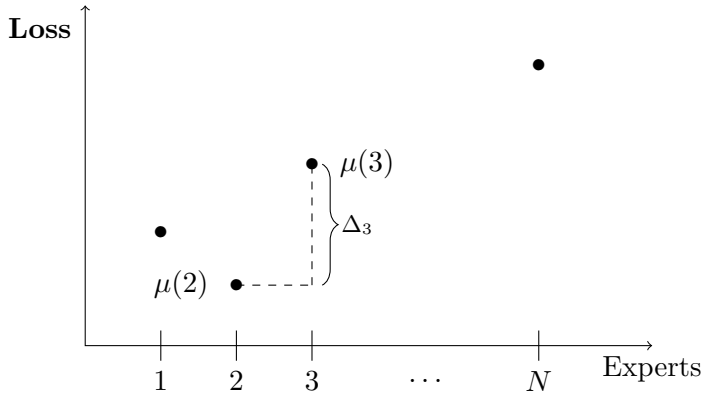

Figure 2: Illustration of the quantity $\Delta_3 = \mu(3) - \mu(a^*)$, where $a^* = 2$.

The first step towards proving the regret bound is to express it in terms of Δ_a and $\mathbb{E}(n_T(a))$. Recall that $n_T(a)$ is the number of times action a is played by the end of the game.

Lemma 1. *For any algorithm $\mathcal{A}$ in the stochastic setting,*

$$\mathit{Regret}_{\mathit{stoch}} = \sum_{a \in [N]} \Delta_a \, \mathbb{E}(n_t(a)).$$

Proof. The proof steps are linearity of expectation and applying our notation:

$$\begin{aligned}
\text{Regret}_{\text{stoch}} &= \mathop{\mathbb{E}}_{\ell_t, \mathcal{A}} \left(\sum_{t=1}^{T} \ell_t(a_t) - \sum_{t=1}^{T} \ell_t(a^*) \right) \\
&= \mathbb{E} \left(\sum_{t=1}^{T} \mu(a_t) - \mu(a^*) \right) \\
&= \mathbb{E} \left(\sum_{t=1}^{T} \Delta_{a_t} \right) \\
&= \mathbb{E} \left(\sum_{t=1}^{T} \sum_{a \in [N]} \Delta_{a_t} \mathbb{1}[a_t = a] \right) \\
&= \sum_{a \in [N]} \Delta_{a_t} \mathbb{E} \left(\sum_{t=1}^{T} \mathbb{1}[a_t = a] \right) \\
&= \sum_{a \in [N]} \Delta_{a_t} \mathbb{E} \left(n_T(a) \right),
\end{aligned}$$

which completes the proof. □

3.1 Bounding The Regret

In this section, we state and prove the regret bound for the Explore-Then-Exploit algorithm.

Theorem 1. *The following regret bound holds for the Explore-Then-Exploit algorithm:*

$$\textit{Regret}_{\textit{stoch}} \leq \sum_{a:\Delta_a > 0} \left[\frac{T_0}{N} + 2T \exp\left(\frac{-T_0 \Delta_a^2}{8N} \right) \right] \Delta_a.$$

Note that the two terms inside the summation respond differently to increasing the value of T_0. To get a good bound, we need to pick a T_0 that balances these two terms. One choice is $T_0 = \frac{16N}{\Delta^2} \log(2T)$, in which case the second term is

$$2T \exp\left(\frac{-T_0 \Delta_a^2}{8N} \right) \leq 2T \exp(-2 \log(2T)) \ll 1.$$

Thus, we get the bound

$$\begin{aligned}
\text{Regret}_{\text{stoch}} &\leq \sum_{a:\Delta_a > 0} \left(\frac{T_0}{N} + 1 \right) \Delta_a \\
&= O\left(\frac{\log(2T)}{\Delta^2} \sum_{a:\Delta_a > 0} \Delta_a \right).
\end{aligned}$$

Corollary 1. *For $T_0 = \frac{16N}{\Delta^2} \log(2T)$, we have the regret bound $\textit{Regret}_{\textit{stoch}} = O\left(\frac{\log(2T)}{\Delta^2} \sum_{a:\Delta_a > 0} \Delta_a \right)$.*

It should be noted that the choice of T_0 here depends on Δ, which is a property of the distribution P generally not assumed to be known by the player. Nevertheless, it is

still heartening to know that the baseline algorithm performs logarithmically well for some choice of T_0.

It remains to prove Theorem 1. To outline the proof, notice that the Explore-Then-Exploit algorithm depends heavily on how well $\hat{\mu}_{T_0}$ estimates μ. The proof uses Hoeffding's inequality to show that for actions a with $\mu(a) > \mu(a^*)$, we also have $\hat{\mu}_{T_0}(a) > \hat{\mu}_{T_0}(a^*)$ with high probability (depending on T_0).

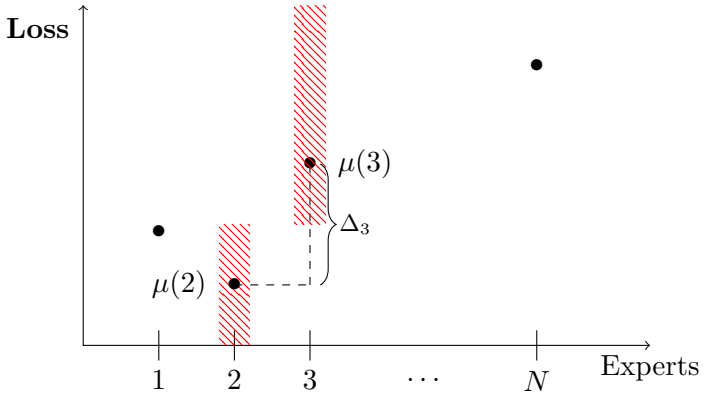

Figure 3: The case in the proof of Theorem 1 where $a = 3$, $a^* = 2$, and neither E_1 or E_2 happens. Then $\hat{\mu}_{T_0}(2)$ and $\hat{\mu}_{T_0}(3)$ lie in the shaded red regions.

Proof of Theorem 1. By Lemma 1, we have

$$\text{Regret}_{\text{stoch}} = \sum_{a \in [N]} \Delta_a \, \mathbb{E}(n_t(a)).$$

Therefore it suffices to bound

$$\mathbb{E}\left(n_T(a)\right) \leq \frac{T_0}{N} + 2T \exp\left(\frac{-T_0 \Delta_a^2}{8N}\right)$$

for each action a with $\Delta_a > 0$.

For any action a, notice that at the end of the explore phase $\hat{\mu}_{T_0}(a)$ is an average of T_0/N i.i.d. draws $\ell_i(a) \in [0, 1]$. By Hoeffding's inequality,

$$\underbrace{P\left(\mu(a) - \hat{\mu}_{T_0}(a) \geq \frac{\Delta_a}{2}\right)}_{E_1} \leq \exp\left(-\frac{T_0 \Delta_a^2}{8N}\right)$$

$$\underbrace{P\left(\hat{\mu}_{T_0}(a^*) - \mu(a^*) \geq \frac{\Delta_a}{2}\right)}_{E_2} \leq \exp\left(-\frac{T_0 \Delta_a^2}{8N}\right).$$

Suppose neither event E_1 or event E_2 happens. Then $\hat{\mu}_{T_0}(a) > \hat{\mu}_{T_0}(a^*)$. (See Figure 3.) As a result, the algorithm never plays action a for any time t during the exploit phase. Thus $n_T(a) = \frac{T_0}{n}$ and therefore $\mathbb{E}(n_T(a) \mid E_1^c \cap E_2^c) \leq \frac{T_0}{N}$.

Also

$$\begin{aligned} P(E_1^c \cap E_2^c) &\leq 1 \\ \mathbb{E}(n_T(a) \mid E_1 \cup E_2) &\leq T \\ P(E_1 \cup E_2) &\leq 2\exp\left(-\frac{T_0\Delta_a^2}{8N}\right), \end{aligned}$$

so by the law of total expectation

$$\mathbb{E}\left(n_T(a)\right) \leq \frac{T_0}{N} + 2T\exp\left(\frac{-T_0\Delta_a^2}{8N}\right),$$

as required. □

The Explore-Then-Exploit algorithm takes a rather crude approach to the explore-exploit tradeoff. The algorithm wastes time exploring bad actions during the explore phase, and potentially misses out on better actions during the exploit phase. In the next section, we cover the UCB algorithm, which balances exploration and exploitation at each step.

4 The UCB Algorithm

The Upper Confidence Bound algorithm takes into account how confident the player is in the estimate $\hat{\mu}_t(a)$. The algorithm builds a confidence interval $I_t = [x_{a,t}, y_{a,t}]$ around $\hat{\mu}_t(a)$ so that $\mu(a)$ lies in I_t with high probability. The algorithm then picks $a_t = \operatorname{argmin}_{a\in[N]} x_{a,t}$ at each step. (The reason the algorithm is called "Upper" Confidence Bound and not "Lower" Confidence Bound is because in the literature researchers focus on maximizing reward whereas in our problem we minimize loss.)

This setup allows UCB to explore and exploit simultaneously. For little-explored actions, the interval I_t will be large. This encourages exploration because the player is incentivized to pick a since the lower bound $x_{a,t}$ will be small (relative to $\hat{\mu}_t(a)$). At the same time, the algorithm avoids wasting excessive time on actions with large values of $\mu(a)$. And UCB exploits its information at each step by picking $a_t = \operatorname{argmin}_{a\in[N]} x_{a,t}$.

To state the algorithm in detail, we need to get explicit expressions for $x_{a,t}, y_{a,t}$. We will use Hoeffding's inequality for this.

Recall that $\hat{\mu}_t(a)$ is the average of $n_t(a)$ i.i.d. draws $l_i(a) \in [0,1]$, and that $\mathbb{E}(\hat{\mu}_t(a)) = \mu(a)$. By Hoeffding's inequality[1] with probability $1 - 2\exp(-2c^2)$ we have

$$|\hat{\mu}_t(a) - \mu(a)| \leq \frac{c}{\sqrt{n_t(a)}}.$$

[1]Here it seems to me that there were a few small mistakes in lecture. For example it was stated that with probability $1-\exp(-c^2/2)$ we have $|\hat{\mu}_t(a)-\mu(a)| \leq \sqrt{\frac{c}{n_t(a)}}$ by Hoeffding's inequality. Also, in lecture we plugged in $c = \sqrt{4\log(T)}$, but then $\sqrt{\frac{c}{n_t(a)}} = \sqrt{\frac{\sqrt{4\log(T)}}{n_t(a)}}$ and in lecture the double square root sign disappeared. Nonetheless (if everything here is correct), we still arrive at basically the same proposition. (In lecture we differed by a factor of 2 and had $x_{t,a} = \hat{\mu}_t(a) - 2\sqrt{\frac{\log(T)}{n_t(a)}}$.)

Plugging in $c = \sqrt{\log(T)}$, we get that with probability $1 - \frac{2}{T^2}$, we get

$$|\hat{\mu}_t(a) - \mu(a)| \leq \sqrt{\frac{\log(T)}{n_t(a)}}.$$

Taking a union bound over $t = 1, \ldots, T$, we establish the following proposition.

Proposition 1. *Fix an action a in $[N]$. With probability $1 - \frac{2}{T}$, we have*

$$\mu(a) \in \left[\hat{u}_t(a) - \sqrt{\frac{\log(T)}{n_t(a)}}, \hat{u}_t(a) + \sqrt{\frac{\log(T)}{n_t(a)}}\right]$$

for each $t = 1, \ldots, T$.

We are ready to state the UCB algorithm. Define the lower confidence bound $\mathrm{LCB}_t(a)$ at time t based on Proposition 1:

$$\mathrm{LCB}_t(a) \triangleq \hat{\mu}_{t-1}(a) - \sqrt{\frac{\log(T)}{n_{t-1}(a)}}.$$

The UCB algorithm picks optimistically according to $\mathrm{LCB}_t(a)$.

UCB Algorithm

At each step t, the player picks the action $\operatorname{argmin}_{a \in [N]} \mathrm{LCB}_t(a)$.

One feature of the confidence bound in Proposition 1 is that the interval grows arbitrarily large as $T \to \infty$. Thus every action will be explored by UCB arbitrarily many times. At the same time, the UCB algorithm exploits at each step.

We will continue our study of the UCB algorithm in the next lecture.