E0 370 Statistical Learning Theory **Lecture 22 (Nov 7, 2013)**

Stochastic Multi-Armed Bandits

Lecturer: Shivani Agarwal *Scribe: Arpit Agarwal*

1 Introduction

In this lecture we will start to look at the multi-armed bandit (MAB) problem, which can be viewed as a form of online learning in which the learner receives only *partial information* at the end of each trial. Specifically, in an n-armed bandit problem, there are n different options that can be chosen, or n different *arms* that can be pulled, on each trial; each of these arms, when pulled, generates some reward (or loss).[1] On each trial, the learner selects one arm to pull, and observes the reward associated with *only the pulled arm*; the rewards associated with the other arms remain hidden from the learner (it is in this sense that the learner receives only partial information). The goal of the learner is accumulate as much reward as possible over a sequence of trials, e.g. compared to the best fixed arm in hindsight. Such problems arise in a variety of applications, e.g. in clinical trials, where a doctor must select one of n available treatments to give to each incoming patient, and gets to observe the outcome of only the particular treatment delivered to the patient; in ad placements, where one needs to decide which of n available ads to display to any given user, and gets to observe the click behavior/revenue generated only for the ad displayed; and so on. In all these applications, there is an inherent trade-off between *exploration* (trying a new arm that might yield better rewards) and *exploitation* (selecting an arm that has been observed to give good rewards so far), and any good MAB algorithm must somehow balance these two aspects.

In the following, we will let n denote the number of arms, t denote the trial, and x_i^t denote the reward obtained on pulling arm i on trial t; for simpliciy, we will assume the rewards x_i^t are all bounded, say $x_i^t \in [0,1]$. The general MAB problem can then be described as follows:

Multi-Armed Bandit (MAB) Learning

For $t = 1, \ldots, T$:
- Algorithm selects an arm $i_t \in [n]$
- Simultaneously, environment selects a reward vector $\mathbf{x}^t = (x_1^t, \ldots, x_n^t) \in [0,1]^n$
- Algorithm observes reward $x_{i_t}^t$

A play over T trials consists of the sequence $(i_1, \mathbf{x}^1), (i_2, \mathbf{x}^2), \ldots, (i_T, \mathbf{x}^T)$ of the arms and reward vectors selected by the algorithm and environment. Formally, an MAB algorithm $\mathcal{A}$ is a policy which, given a sequence $(i_1, x_{i_1}^1), (i_2, x_{i_2}^2), \ldots, (i_{t-1}, x_{i_{t-1}}^{t-1})$ of the arms pulled and rewards obtained in the first $t-1$ trials, returns an arm $i_t \in [n]$ to play on trial t. Note that i_t in general depends on $x_{i_1}^1, \ldots, x_{i_{t-1}}^{t-1}$ (and $i_1, \ldots, i_{t-1}$).

The *gain/reward* of an algorithm $\mathcal{A}$ on a sequence of reward vectors $(\mathbf{x}^1, \ldots, \mathbf{x}^T)$ is given by

$$G_T[\mathcal{A}] \equiv G_{(\mathbf{x}^1, \ldots, \mathbf{x}^T)}[\mathcal{A}] = \sum_{t=1}^{T} \mathbf{x}_{i_t}^t$$

Similarly, the *gain/reward* of a fixed arm i on a sequence of reward vectors $(\mathbf{x}^1, \ldots, \mathbf{x}^T)$ is given by

$$G_T[i] \equiv G_{(\mathbf{x}^1, \ldots, \mathbf{x}^T)}[i] = \sum_{t=1}^{T} \mathbf{x}_i^t \,.$$

[1] The term 'n-armed bandit' comes from the colloquial term 'one-armed bandit' used to refer to a slot machine in a casino (traditionally, such slot machines had one lever/arm, and more often than not, pulling the arm led to loss of money!)

The *regret* of an algorithm $\mathcal{A}$ over the sequence $(\mathbf{x}^1, \ldots, \mathbf{x}^T)$ is then given by

$$R_T[\mathcal{A}] \equiv R_{(\mathbf{x}^1,\ldots,\mathbf{x}^T)}[\mathcal{A}] = \max_{i\in[n]} G_{(\mathbf{x}^1,\ldots,\mathbf{x}^T)}[i] - G_{(\mathbf{x}^1,\ldots,\mathbf{x}^T)}[\mathcal{A}]\,.$$

The goal of an MAB algorithm is typically to minimize the above regret, either in a worst case sense or in expectation or with high probability, depending on how the rewards $(\mathbf{x}^1, \ldots, \mathbf{x}^T)$ are assumed to be generated. Indeed, there are various settings in which MAB problems have been studied:

- **Stochastic MABs**: In this setting, each arm $i \in [n]$ is associated with an (unknown) probability distribution ν_i on $[0, 1]$, and rewards from arm i are assumed to be drawn i.i.d. from ν_i.
- **Adversarial MABs**: Here there are no probabilistic assumptions on the rewards x_i^t. Instead, the rewards can be generated by an adversary.
- **Markovian MABs**: Here the rewards of each arm follow a Markov process on some underlying state space. The tools used to analyze Markovian MABs are quite different from those used to analyze stochastic and adversarial MABs, and are more similar to tools used in reinforcement learning.

In this lecture, we will focus on the stochastic MAB setting; the next lecture will discuss adversarial MABs. Much of our discussion is based on that in [4], which is also a great resource for further details on MABs.

2 Stochastic MABs

As noted above, in the stochastic setting, there is an (unknown) probability distribution ν_i on $[0, 1]$ for each arm i, from which rewards for that arm are assumed to be drawn i.i.d. Let us introduce some notation that will be useful in this setting. Let μ_i denote the (unknown) mean reward for arm i: $\mu_i = \mathbf{E}_{x_i\sim\nu_i}[x_i]$. Let

$$\mu^* = \max_{i\in[n]} \mu_i$$

be the highest mean reward of any arm, and for each $i \in [n]$, let

$$\Delta_i = \mu^* - \mu_i$$

be the gap between the mean reward of an optimal arm and the mean reward of arm i.

In the stochastic setting, it makes sense to bound the regret of an algorithm in expectation or with high probability over the draw of the rewards. The *expected regret* of an algorithm $\mathcal{A}$ over T trials is given by

$$\mathbf{E}_{(\mathbf{x}^1,\ldots,\mathbf{x}^T)}\left[\max_{i\in[n]} \sum_{t=1}^{T} x_i^t - \sum_{t=1}^{T} x_{i_t}^t\right].$$

In practice, however, the above quantity is difficult to work with, due to the max term inside the expectation. Therefore, when designing and analyzing algorithms for stochastic MABs, one often considers minimizing the *pseudo-regret* instead, given by

$$\bar{R}_T[\mathcal{A}] = \max_{i\in[n]} \mathbf{E}_{(\mathbf{x}^1,\ldots,\mathbf{x}^T)}\left[\sum_{t=1}^{T} x_i^t - \sum_{t=1}^{T} x_{i_t}^t\right].$$

The pseudo-regret can be simplified as follows:

$$\begin{aligned}
\bar{R}_T[\mathcal{A}] &= \max_i \mathbf{E}_{(\mathbf{x}^1,\ldots,\mathbf{x}^T)}\left[\sum_{t=1}^T x_i^t\right] - \mathbf{E}_{(\mathbf{x}^1,\ldots,\mathbf{x}^T)}\left[\sum_{t=1}^T x_{i_t}^t\right] \\
&= \max_i T\mu_i - \sum_{t=1}^T \mathbf{E}_{(\mathbf{x}^1,\cdots,\mathbf{x}^t)}\left[x_{i_t}^t\right] \\
&= T\mu^* - \sum_{t=1}^T \mathbf{E}_{(\mathbf{x}^1,\ldots,\mathbf{x}^{t-1})}\Big[\mathbf{E}_{\mathbf{x}^t}\left[x_{i_t}^t \,|\, \mathbf{x}^1,\cdots,\mathbf{x}^{t-1}\right]\Big] \\
&= T\mu^* - \sum_{t=1}^T \mathbf{E}_{(\mathbf{x}^1,\cdots,\mathbf{x}^{t-1})}\left[\mu_{i_t}\right], \qquad \text{since } \mathbf{x}^t \text{ is independent of } \mathbf{x}^1,\ldots,\mathbf{x}^{t-1} \\
&\equiv T\mu^* - \sum_{t=1}^T \mathbf{E}\left[\mu_{i_t}\right].
\end{aligned}$$

Note that the pseudo-regret is upper bounded by the expected regret, and therefore an upper bound on the pseudo-regret does not imply an upper bound on the expected regret (i.e. an upper bound on the pseudo-regret is a weaker statement than an upper bound on the expected regret). When there is randomization in the algorithm, one also takes expectation over this randomness in defining the above quantities.

In order to minimize the pseudo-regret, one essentially tries to find an optimal arm, i.e. an arm with highest mean reward μ^*. Various strategies have been proposed to do this; in most cases, one makes use of sample means of the rewards obtained with different arms to estimate the true means μ_i. One such example is the ϵ-greedy algorithm, which selects the 'best' previously explored arm (one with highest sample mean reward so far) with probability $(1-\epsilon)$, and randomly (uniformly) picks an arm with probability ϵ. In the following, we will study the popular Upper Confidence Bound (UCB) algorithm proposed by Auer et al. [2], which uses a somewhat more sophisticated approach based on construction of confidence bounds for the means μ_i, and which yields good pseudo-regret guarantees.

3 Upper Confidence Bound (UCB) Algorithm

As noted above, most algorithms for stochastic MABs essentially try to estimate the mean rewards μ_i using sample means of the rewards observed with different arms. The basic idea behind the UCB algorithm [2] is to maintain confidence intervals for the various mean rewards μ_i through the use of appropriate concentration inequalities. On any given trial t, the algorithm picks an arm with the highest current upper confidence bound on the mean reward for a suitable confidence parameter δ_t. The intuition is that the upper confidence bound for the mean reward of an arm can be high in two cases: (a) when we have not sampled from the arm a sufficient number of times to get a tight confidence bound; or (b) when the mean reward is high. The first case leads to exploration, while the second case leads to exploitation. As an arm is sampled more and more times, the confidence interval around the sample mean becomes tighter, giving a more accurate estimate of the true mean reward. Moreover, the parameter δ_t shrinks fast enough with t such that overall, after a sufficient number of trials, with high probability, one selects an optimal arm.

In order to describe the algorithm, let us introduce some notation. Specifically, let N_i^t denote the number of times arm i has been pulled in the first t trials:

$$N_i^t = \sum_{s=1}^t \mathbf{1}(i_s = i)\,,$$

and let $\widehat{\mu}_i^t$ denote the sample mean of the rewards obtained from pulls of arm i during the first t trials:

$$\widehat{\mu}_i^t = \frac{1}{N_i^t}\sum_{s=1}^t x_i^s \cdot \mathbf{1}(i_s = i)\,.$$

The UCB algorithm works as follows (as the analysis will show, the strategy corresponds to using an upper confidence bound derived from Hoeffding's inequality for a suitable choice of confidence parameter δ_t):

Algorithm **UCB**

Parameter $\alpha > 0$
For $t = 1, \ldots, T$:

- Select arm $i_t \in \text{argmax}_{i \in [n]} \left(\widehat{\mu}_i^{t-1} + \sqrt{\frac{\alpha \ln t}{2N_i^{t-1}}} \right)$
- Observe reward $x_{i_t}^t$

Note that the pseudo-regret of any algorithm $\mathcal{A}$ over T trials can be written in terms of the random variables N_i^T as follows:

$$\begin{aligned} \bar{R}_T[\mathcal{A}] \;=\; T\mu^* - \mathbf{E}\left[\sum_{t=1}^{T} \mu_{i_t}\right] \;&=\; T\mu^* - \mathbf{E}\left[\sum_{i=1}^{n} N_i^T \mu_i\right] \\ &=\; \mathbf{E}\left[\sum_{i=1}^{n} N_i^T\right] \mu^* - \mathbf{E}\left[\sum_{i=1}^{n} N_i^T \mu_i\right] \\ &=\; \sum_{i:\Delta_i > 0} \mathbf{E}\left[N_i^T\right] \Delta_i \,. \end{aligned} \tag{1}$$

Therefore, to bound the pseudo-regret of an algorithm over T trials, it suffices to bound $\mathbf{E}[N_i^T]$, the expected number of times arm i is pulled in the T trials, for each sub-optimal arm $i : \Delta_i > 0$.

In order to analyze the UCB algorithm, we start with the following lemma, which captures the intuition that if a sub-optimal arm is pulled, then either we have not pulled the arm sufficiently many times, or we have been unlucky with the random draws of rewards and have a bad confidence interval constructed for either the pulled arm or an optimal arm:

Lemma 3.1 (Auer et al., 2002)**.** Let i^* denote any optimal arm so that $\mu_{i^*} = \mu^*$, and fix any $i : \Delta_i > 0$. If the UCB algorithm selects the arm i in trial t (i.e. $i_t = i$), then at least one of the following holds:

1. $\widehat{\mu}_{i^*}^{t-1} \leq \mu^* - \sqrt{\frac{\alpha \ln t}{2N_{i^*}^{t-1}}}$
2. $\widehat{\mu}_i^{t-1} \geq \mu_i + \sqrt{\frac{\alpha \ln t}{2N_i^{t-1}}}$
3. $N_i^{t-1} \leq \frac{2\alpha \ln T}{\Delta_i^2}$

Proof. Suppose the UCB algorithm selects arm i in trial t. If (1), (2) and (3) are all false, then we get

$$\begin{aligned} \widehat{\mu}_{i^*}^{t-1} + \sqrt{\frac{\alpha \ln t}{2N_{i^*}^{t-1}}} \;&>\; \mu^* \,, && \text{since (1) is false} \\ &=\; \mu_i + \Delta_i \\ &>\; \mu_i + \sqrt{\frac{2\alpha \ln T}{N_i^{t-1}}} \,, && \text{since (3) is false} \\ &\geq\; \mu_i + \sqrt{\frac{2\alpha \ln t}{N_i^{t-1}}} \\ &>\; \widehat{\mu}_i^{t-1} + \sqrt{\frac{\alpha \ln t}{2N_i^{t-1}}} \,, && \text{since (2) is false,} \end{aligned}$$

which contradicts the fact that $i_t = i$. □

Theorem 3.2 (Auer et al., 2002)**.** Let $x_i^t \in [0,1]\ \forall i,t$, and let $\alpha > 2$. Then

$$\bar{R}_T\big[\mathrm{UCB}(\alpha)\big] \;\le\; \left(\sum_{i:\Delta_i>0} \frac{2\alpha}{\Delta_i}\right) \ln T + \frac{\alpha}{\alpha-2} \sum_{i:\Delta_i>0} \Delta_i \,.$$

Proof. Let i^* be any optimal arm: $\mu_{i^*} = \mu^*$. Fix any sub-optimal arm $i : \Delta_i > 0$, and define $t_0 = \left\lceil \frac{2\alpha \ln T}{\Delta_i^2} \right\rceil$. Then we have

$$\begin{aligned}
\mathbf{E}[N_i^T] &= \mathbf{E}\Big[\sum_{t=1}^{T} \mathbf{1}(i_t = i)\Big] \\
&= \mathbf{E}\Big[\sum_{t=1}^{T} \mathbf{1}(i_t = i, N_i^{t-1} \le t_0) + \sum_{t=1}^{T} \mathbf{1}(i_t = i, N_i^{t-1} > t_0)\Big] \\
&\le t_0 + \sum_{t=t_0+1}^{T} \mathbf{P}\big(i_t = i, N_i^{t-1} > t_0\big) \\
&\le t_0 + \sum_{t=t_0+1}^{T} \left(\mathbf{P}\left(\widehat{\mu}_{i^*}^{t-1} \le \mu^* - \sqrt{\frac{\alpha \ln t}{2N_{i^*}^{t-1}}}\right) + \mathbf{P}\left(\widehat{\mu}_i^{t-1} \ge \mu_i + \sqrt{\frac{\alpha \ln t}{2N_i^{t-1}}}\right)\right), \quad \text{by Lemma 3.1}\,.
\end{aligned}$$

Now,

$$\begin{aligned}
\sum_{t=t_0+1}^{T} \mathbf{P}\left(\widehat{\mu}_{i^*}^{t-1} \le \mu^* - \sqrt{\frac{\alpha \ln t}{2N_{i^*}^{t-1}}}\right) &\le \sum_{t=t_0+1}^{T} \mathbf{P}\left(\bigcup_{s=1}^{t-1} \left\{\widehat{\mu}_{i^*}^{t-1} \le \mu^* - \sqrt{\frac{\alpha \ln t}{2s}}\right\}\right) \\
&\le \sum_{t=t_0+1}^{T} \frac{1}{t^{\alpha-1}}\,, \quad \text{by Hoeffding's Inequality} \\
&\le \sum_{t=2}^{T} \frac{1}{t^{\alpha-1}} \\
&\le \frac{1}{\alpha-2}\,.
\end{aligned}$$

Similarly, we can show

$$\sum_{t=t_0+1}^{T} \mathbf{P}\left(\widehat{\mu}_i^{t-1} \ge \mu_i + \sqrt{\frac{\alpha \ln t}{2N_i^{t-1}}}\right) \le \frac{1}{\alpha-2}$$

This gives

$$\begin{aligned}
\mathbf{E}[N_i^T] &\le t_0 + \frac{2}{\alpha-2} \\
&\le \frac{2\alpha \ln T}{\Delta_i^2} + 1 + \frac{2}{\alpha-2}\,, \quad \text{by definition of } t_0 \\
&= \frac{2\alpha \ln T}{\Delta_i^2} + \frac{\alpha}{\alpha-2}\,.
\end{aligned}$$

From Eq. (1), we thus get the following bound on the pseudo-regret of the UCB algorithm:

$$\begin{aligned}
\bar{R}_T[\mathrm{UCB}(\alpha)] &\le \sum_{i:\Delta_i>0} \left(\frac{2\alpha \ln T}{\Delta_i^2} + \frac{\alpha}{\alpha-2}\right) \Delta_i \qquad (2) \\
&= \left(\sum_{i:\Delta_i>0} \frac{2\alpha}{\Delta_i}\right) \ln T + \frac{\alpha}{\alpha-2} \sum_{i:\Delta_i>0} \Delta_i\,. \qquad (3)
\end{aligned}$$

□

Note that the above bound depends on the reward distributions ν_i via Δ_i; these quantities are typically not known. A few additional notes:

1. The above analysis can be improved to show an upper bound of $\frac{\alpha}{2} \ln T$ for all $\alpha > 1$.
2. For distribution-free bounds: one can show a lower bound of $\sqrt{Tn}$, and can show that UCB(α) achieves a pseudo-regret of at most $\sqrt{\alpha T n \ln T}$.
3. For a slight variant of the UCB algorithm, one can obtain high probability bounds on $T\mu^* - \sum_{t=1}^{T} \mu_{i_t}$.
4. The ϵ-greedy algorithm achieves an $O(n \ln T)$ bound for ϵ_t chosen depending on some knowledge of $\min_{i \neq i^*} \Delta_i$.

More details can be found in [4].

4 Lower Bound

The following is a well known lower bound on the pseudo-regret of any algorithm in the stochastic MAB setting [5]:

Theorem 4.1 (Lai & Robbins, 1985)**.** Let $x_i^t \sim$ Bernoulli(μ_i). Let $\mathcal{A}$ be any algorithm such that for any set of Bernoulli rewards, for any sub-optimal arm $i : \Delta_i > 0$ and any $a > 0$,

$$\mathbf{E}[N_i^T] = o(T^a)\,,$$

i.e. the algorithm eventually samples any sub-optimal arm a vanishing fraction of times. Then

$$\liminf_{T\to\infty} \frac{\bar{R}_T[\mathcal{A}]}{\ln T} \;\geq\; \sum_{i:\Delta_i>0} \frac{\Delta_i}{\mathrm{KL}(\mu_i, \mu^*)}\,.$$

5 Additional Pointers

We note that recently, Garivier & Cappe [3] and Maillard et al. [6] gave the KL-UCB algorithm that makes use upper confidence bounds derived from KL-divergence based concentration inequalities rather than Hoeffding's inequality, and that matches the constant factor in the lower bound above; in particular, for $x_i^t \in [0,1]\ \forall i, t$ and any $\alpha > 1$, the algorithm satisfies

$$\bar{R}_T\big[\text{KL-UCB}(\alpha)\big] \;\leq\; \sum_{i:\Delta_i>0} \frac{\Delta_i}{\mathrm{KL}(\mu_i, \mu^*)} \alpha \ln T + O(1)\,.$$

We also note that an early algorithm used for stochastic MABs, namely Thompson sampling [7], which makes use of a Bayesian approach and has been observed to perform well empirically, was also recently shown to have $O(\ln T)$ pseudo-regret [1].

6 Next Lecture

In the next lecture we will study the adversarial MAB setting, where the reward vectors can be generated by an adversary; the analysis there will be more similar to the online learning settings we have seen earlier.

References

[1] Shipra Agrawal and Navin Goyal. Analysis of Thompson Sampling for the multi-armed bandit problem. In *COLT*, 2012.

[2] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2):235–256, 2002.

[3] Olivier Cappé Aurélien Garivier. The KL-UCB algorithm for bounded stochastic bandits and beyond. In *COLT*, 2011.

[4] Sebastien Bubeck and Nicolo Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends in Machine Learning*, 5(1):1–122, 2012.

[5] T. Lai and H. Robbins. Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics*, 6:4–22, 1985.

[6] Odalric-Ambrym Maillard, Rémi Munos, and Gilles Stoltz. A finite-time analysis of multi-armed bandits problems with Kullback-Leibler divergences. In *COLT*, 2011.

[7] William R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3-4):285–294, 1933.