STATS 710 – Seq. Dec. Making with mHealth Applications Sep 06, 2016

Lecture 1: Introduction

Instructors: Susan Murphy and Ambuj Tewari *Scribe: Ambuj Tewari*

1 Sequential Decision Making in mHealth

This course will focus on the use of sequential decision making algorithms in mobile health applications to enable long term health behavior change and maintenance. Behavior change can lead to improved health outcomes in many health conditions, including cardiovascular diseases, substance abuse, and mental illnesses. The promise of mobile health is that tailored interventions can be delivered to users at the right time, in the right context, as they go about living their daily lives.

The basic sequential decision making protocol in mobile health is as follows.

1: **at** a given decision point **do**
2: mobile phone collects tailoring variables (the context)
3: a decision rule (or policy) maps the tailoring variables into an intervention option (the action)
4: mobile phone records the proximal outcome (interpreted as a reward, so higher is better)
5: **done**

For example, in a project called HeartSteps that aims to maintain physical activity in user after cardiac rehab, context variables include GPS location, outside weather, label (such as walking, running, in vehicle) from google activity recognition, calendar busyness. There are two intervention options for momentary suggestions that are delivered on the smartphone's lock screen: 1, i.e., provide a suggestion and 0, i.e., don't provide a suggestion. The proximal outcome or reward is the number of steps walked (measured via a wristband) in a specified time period, say an hour, following the suggestion.

In this course we will look at increasingly sophisticated methods to sequentially choose actions to maximize sum of received rewards. The first class of methods, called *multi-armed bandits*, apply in situations where there is no contextual information available to guide action selection. This is not a realistic case in mHealth application but serves as a foundation for more sophisticated methods. The second class of methods, called *contextual bandits*, do take into account contextual information but do not reason about delayed effects of actions on future contexts. Delayed effects can easily occur in mHealth applications, e.g., providing too many suggestions to the user can affect the burden involved in using the app and reduce effectiveness of future suggestions since they might be ignored by the user. Finally, we will look at general *reinforcement learning* algorithms that do take into account delayed effects.

2 Multiarmed Bandits: Regret and Basic Definitions

The name "bandits" comes from "one-armed bandit" which refers a slot machine in casinos operated by pulling a long arm at the side. These machines rob you of your money, hence the name "bandits"! Now imagine having several arms to choose from. You do not know which arms are good and which ones are bad. So you will have to *explore* new arms in the hope of finding good ones that have not been used much so far, but also *exploit* your knowledge from previous rounds to stick to ones that have performed well in the past.

Robbins [Rob52] originally formalized the problem that we now refer to as multiarmed bandit. We will use "action" and "arm" interchangeably. He considered the case of two actions but it is not too difficult to extend the framework to consider a set of K actions $\mathcal{A} = \{0, 1, \ldots, K-1\}$. To each action $a \in \mathcal{A}$ we attach a distribution D_a. Assume that D_a has finite mean μ_a and consider a sequence of iid random variables

$$R_1^a, R_2^a, \ldots$$

distributed according to D_a.

The multiarmed bandit (MAB) protocol is as follows.

1: **for** $t = 1$ to T **do**
2: Learner selects action $A_t \in \mathcal{A}$
3: Learner receives reward $R_t = R_t^{A_t}$
4: **end for**

A deterministic learning algorithm $\mathcal{L}$ for MAB problems is a sequence of maps $\mathcal{L}_t$ that map the history at time $\mathcal{H}_t = (A_1, R_1, \ldots, A_{t-1}, R_{t-1})$ to an action in $\mathcal{A}$, i.e., $A_t = \mathcal{L}_t(\mathcal{H}_t)$. A randomized learning algorithm can use its own private source of randomness in action selection.

The expected regret of a learning algorithm $\mathcal{L}$ against distributions $(D_a)_{a\in\mathcal{A}}$ is defined as

$$\mathcal{R}_T(\mathcal{L}, (D_a)_{a\in\mathcal{A}}) := \mu_\star T - \mathbb{E}\left[\sum_{t=1}^{T} R_t\right] \tag{1}$$

where $\mu_\star = \max_{a\in\mathcal{A}} \mu_a$ is the maximum mean reward among all actions. Note that the expectation in the second term is with respect to two sources of randomness. First, the underlying rewards R_t^a are random. Second, the learning algorithm itself can itself be randomized.

There is another expression for the regret that will be convenient to have later. Define the indicator variable $I_{t,a}$ that turns on iff arm a is selected at time t. Define the action counts

$$N_t(a) = \sum_{s=1}^{t} I_{s,a}.$$

We then have,

$$\mathcal{R}_T(\mathcal{L}, (D_a)_{a\in\mathcal{A}}) = \sum_{a\in\mathcal{A}} \Delta_a \mathbb{E}\left[N_T(a)\right] = \sum_{a:\mu_a<\mu_\star} \Delta_a \mathbb{E}\left[N_T(a)\right] \tag{2}$$

where $\Delta_a = \mu_\star - \mu_a$ is the gap between the mean reward of the best action and action a.

Distribution specific regret bounds are allowed to depend on properties of the individual reward distributions D_a such as the gaps Δ_a.

To derive *distribution free* regret bounds, we think of the distributions D_a as coming from a parametric (e.g., Bernoulli distributions for binary valued rewards or Gaussian distributions for real valued rewards) or a nonparametric family (e.g., all distributions with support in $[0, 1]$) denoted by $\mathcal{D}$. We can assess how "good" a learning algorithm $\mathcal{L}$ is by looking at its worst possible regret over distributions D_a:

$$\mathcal{R}_T(\mathcal{L}, \mathcal{D}) = \sup_{D_a\in\mathcal{D}} \mathcal{R}_T(\mathcal{L}, (D_a)_{a\in\mathcal{A}}).$$

A bound on $\mathcal{R}_T(\mathcal{L}, \mathcal{D})$ is a distribution free regret bound that holds no matter which reward distributions are faced by the learning algorithm as long as they are all in the family $\mathcal{D}$.

A learning algorithm is called *consistent* for a class $\mathcal{D}$ of distributions if, for all $D_a \in \mathcal{D}$,

$$\lim_{T\to\infty} \frac{\mathcal{R}_T(\mathcal{L}, (D_a)_{a\in\mathcal{A}})}{T} = 0.$$

Finally, the minimax regret against a class $\mathcal{D}$ of reward distributions is given by

$$\mathcal{R}_T(\mathcal{D}) = \inf_{\mathcal{L}} \mathcal{R}_T(\mathcal{L}, \mathcal{D}).$$

Determination of exact minimax rates is hard: we usually prove upper and lower bounds separately and try to make them match as much as we can. Upper bounds on the minimax regret can be established by proving a distribution free regret bound for a specific learning algorithm $\mathcal{L}$. Lower bounds have to reason about all possible learning algorithms and usually rely on information theoretic arguments.

3 A General Consistency Result

Robbins [Rob52] proved a general consistency result for two armed bandit problems that just relies on law of large numbers and easily extends to the case of K arms. Let $\mathcal{D}$ be all distributions with a finite mean.

Consider the following learning algorithm that is referred to as "certainty equivalence with forcing" in the control literature. Pre-select an infinite sequence of "forced exploration" time points

$$\tau_{a,1} < \tau_{a,2} < \dots$$

such that the time points are disjoint over $a \in \mathcal{A}$ and $\tau_{a,1} = a + 1$ (i.e., we start by selecting each action from 0 through $K - 1$ once). At any given time point t, if it is one of the forced exploration time points for an action a, select action a. Otherwise, form estimates $\hat{\mu}_{t-1,a}$ of μ_a using the sample mean over all previous exploration rounds for action a, and select the one with maximum value of $\hat{\mu}_{t-1,a}$ (breaking ties, say, in favor of the numerically smaller action).

Define the indicator variables $I'_{t,a}, I''_{t,a}$ to be on iff action a was selected at time t because of exploration, exploitation reasons respectively. Define the counts $N'_t(a), N''_t(a)$ in the same way as we defined $N_t(a)$. Note that $N_t(a) = N'_t(a) + N''_t(a)$. From (2), we have

$$\begin{aligned}\mathcal{R}_T(\mathcal{L}, (D_a)_{a\in\mathcal{A}}) &= \sum_{a:\mu_a<\mu_\star} \Delta_a \mathbb{E}\left[N_T(a)\right] \\ &= \sum_{a:\mu_a<\mu_\star} \Delta_a \mathbb{E}\left[N'_T(a)\right] + \sum_{a:\mu_a<\mu_\star} \Delta_a \mathbb{E}\left[N''_T(a)\right]\end{aligned}$$

Note that

$$N'_T(a) = |\left\{\tau_{a,i} \,:\, 1 \le \tau_{a,i} \le T\right\}|$$

is deterministic. Let us make the assumption that the sequence of exploration points have density zero, i.e., $N'_T(a)/T \to 0$. Then, we just need to ensure that, for all $a \in \mathcal{A}$,

$$\frac{\mathbb{E}\left[N''_T(a)\right]}{T} \to 0.$$

Now note that we can write the previous quantity as a Cesaro sum:

$$\frac{\mathbb{E}\left[N_T''(a)\right]}{T} = \frac{\sum_{t=1}^{T} \mathbb{E}\left[I_{t,a}''\right]}{T}.$$

We know that, if a sequence converges to 0, then so does its Cesaro mean. So the only thing that remains to be shown is that $\mathbb{E}\left[I_{t,a}''\right] \to 0$ as $t \to \infty$ for any strictly suboptimal action a. If action a is selected as a result of exploitation at time t, it must be that $\hat{\mu}_{t-1,a} \geq \hat{\mu}_{t-1,a^\star}$ for any optimal action $a^\star$. But this can only occur if either $|\hat{\mu}_{t-1,a} - \mu_a| \geq \Delta_a/2$ or $|\hat{\mu}_{t-1,a^\star} - \mu_{a^\star}| \geq \Delta_a/2$. Therefore,

$$\mathbb{E}\left[I_{t,a}''\right] \leq \mathbb{P}\left(|\hat{\mu}_{t-1,a} - \mu_a| \geq \Delta_a/2\right) + \mathbb{P}\left(|\hat{\mu}_{t-1,a^\star} - \mu_{a^\star}| \geq \Delta_a/2\right).$$

Both terms on the RHS converge to zero by the law of large numbers.

References

[Rob52] Herbert Robbins. Some aspects of the sequential design of experiments. *Bulletin of the American Mathematical Society*, 58(5):527–535, 1952.