Contextual Bandits

General Setting

> **Problem protocol:** Contextual bandits
>
> For each round $t \in [T]$:
> 1. algorithm observes a "context" x_t,
> 2. algorithm picks an arm a_t,
> 3. reward $r_t \in [0, 1]$ is realized.

IID assumption: reward distribution is parametrized by (x_t, a_t)

Motivation: User profile (ad displaying)

The expected reward for an algorithm is defined as $\mathbb{E}[\text{REW(ALG)}] = \sum_{t=1}^{T} \mu(a_t \mid x_t)$

The optimal policy is defined as $\pi^*(x) = \max_{a \in \mathcal{A}} \mu(a \mid x)$

Then regret is defined as

$$R(T) = \texttt{REW}(\pi^*) - \texttt{REW}(\texttt{ALG}). \tag{8.1}$$

Small number of contexts

Algorithm 8.1: Contextual bandit algorithm for a small number of contexts

Initialization: For each context x, create an instance $\texttt{ALG}_x$ of algorithm $\texttt{ALG}$

for *each round* t **do**

  invoke algorithm $\texttt{ALG}_x$ with $x = x_t$

  "play" action a_t chosen by $\texttt{ALG}_x$, return reward r_t to $\texttt{ALG}_x$.

end

Theorem 8.1. Algorithm 8.1 has regret $\mathbb{E}[R(T)] = O(\sqrt{KT\,|\mathcal{X}|\,\ln T})$, provided that the bandit algorithm $\texttt{ALG}$ has regret

$$\mathbb{E}[R_{\texttt{ALG}}(T)] = O(\sqrt{KT \log T}).$$

(Directly apply UCB1 on each individual contexts)

Proof:

$$\begin{aligned}\mathbb{E}[R(T)] = \textstyle\sum_{x\in\mathcal{X}} \mathbb{E}[R_x(T)] &= \sum_{x\in\mathcal{X}} O(\sqrt{K n_x\,\ln T}) \\ &\le O(\sqrt{KT\,|\mathcal{X}|\,\ln T}).\end{aligned}$$

Q.E.D.

Contextual Bandits with Lipchitz Condition

Assuming $\chi \subset [0,1]$

$$|\mu(a|x) - \mu(a|x')| \le L \cdot |x - x'|$$
$$\text{for any arms } a, a' \in \mathcal{A} \text{ and contexts } x, x' \in \mathcal{X}, \qquad (8.2)$$

Using ε-mesh on $[0,1]$ to discretize the interval uniformly

Let $d = \frac{1}{\varepsilon} + 1$ be the number of points on interval and S be the ε-mesh

$f_S(x) = \arg\min_{x'\in S} |x' - x|$

$$\text{In each round } t, \text{ replace } x_t \text{ with } f_S(x_t), \text{ and call } \texttt{ALG}_S. \qquad (8.3)$$

Theorem 8.3. Consider the Lipschitz contextual bandits problem with contexts in $[0,1]$. The uniform discretization algorithm (8.3) yields regret $\mathbb{E}[R(T)] = O(T^{2/3}(LK\ln T)^{1/3})$.

Proof:

Easy to see discretization error $\mathbb{E}[\mathsf{DE}(S)] \leq \varepsilon LT$

Total Regret

$$\mathbb{E}[R(T)] = O(\sqrt{dKT \ln T}) + \varepsilon LT = O(T^{2/3}(LT \ln T)^{1/3})$$

with choosing $\varepsilon = O(L^{-2/3}(T^{-1}K \ln T)^{1/3})$

Q.E.D.

Linear Contextual bandits: LinUCB (no proof)

In linear *contextual* bandits, contexts are of the form (8.5), and the expected rewards are linear:

$$\mu(a|x) = x_a \cdot \theta_a \quad \text{for all arms } a \text{ and contexts } x, \tag{8.6}$$

for some fixed but unknown vector $\theta = (\theta_a \in \mathcal{R}^d : a \in \mathcal{A})$.

Algorithm 8.2: `LinUCB`: UCB-based algorithm for linear contextual bandits

for *each round* $t = 1, 2, \ldots$ **do**
  Form a confidence region $C_t \in \Theta$ s.t. $\theta \in C_t$ with high probability
  For each arm a, compute $\mathtt{UCB}_t(a|x_t) = \sup_{\theta \in C_t} x_a \cdot \theta_a$
  pick arm a which maximizes $\mathtt{UCB}_t(a|x_t)$.
end

To completely specify the algorithm, one needs to specify what the confidence region is, and how to compute the UCBs. This is somewhat subtle, and there are multiple ways to do that. Full specification and analysis of LinUCB is a topic for another lecture.

Suitably specified versions of `LinUCB` allow for rigorous regret bounds, and work well in experiments. The best known regret bound is of the form $\mathbb{E}[R(T)] = \tilde{O}(\sqrt{dT})$, and there is a nearly matching lower bound $\mathbb{E}[R(T)] \geq \Omega(\sqrt{dT})$. Interestingly, the algorithm is known to work well in practice even for scenarios without linearity.

Contextual bandits with a policy class

Setting:

We define a *policy* as a function from contexts to actions, and posit a know class of policies Π.

We assume contexts arrive as independent samples from some fixed distribution $\mathcal{D}$. We define *policy value* as follows:

$$\mu(\pi) = \mathop{\mathbb{E}}_{x \in \mathcal{D}} [\mu(\pi(x)) \mid x]. \tag{8.7}$$

Regret for an algorithm ALG is defined as:

$$R_\Pi(T) = T \max_{\pi \in \Pi} \mu(\pi) - \texttt{REW}(\texttt{ALG}). \tag{8.8}$$

Preliminary result

Theorem 8.4. Consider contextual bandits with policy class Π. Algorithm Exp4 with expert set Π yields regret $\mathbb{E}[R_\Pi(T)] = O(\sqrt{KT \log |\Pi|})$. However, the running time per round is linear in $|\Pi|$.

Known lower bound:

$$\mathbb{E}[R(T)] \geq \min\left(T,\ \Omega\left(\sqrt{KT \log(|\Pi|)/\log(K)}\right)\right). \tag{8.9}$$

Question: can we find a faster algorithm?

Connection to a classification problem

We first consider the full-feedback version:

Problem protocol: Contextual bandits with full feedback

For each round $t = 1, 2, \ldots$:

1. algorithm observes a "context" x_t,
2. algorithm picks an arm a_t,
3. rewards $\tilde{r}_t(a) \geq 0$ are observed for all arms $a \in \mathcal{A}$.

We consider the even easier version of this:

What policy is offline optimal? Want to solve the following optimization problem:

$$\arg\max_{\pi \in \Pi} \tilde{r}(\pi) = \frac{1}{N} \sum_{t=1}^{N} \tilde{r}_t(\pi(x_i)) \tag{8.10}$$

This happens to be a well-studied problem called "cost-sensitive multi-class classification":

Problem: Cost-sensitive multi-class classification

Given: policy class Π; N data points $(x_t; \tilde{r}_t(a) : a \in \mathcal{A})$, $t \in [N]$.
Find: policy $\pi \in \Pi$ with a largest realized policy value (8.10).

For some policy class, this problem is NP-hard, practically efficient algorithms exist for several important policy classes such as linear classifiers, decision trees and neural nets.

From now, we assume there exists some *classification oracle* $\mathcal{O}$ for the problem above

Now, we can construct a simple explore-first algorithm that builds on a classification oracle.

We define the fake feedback as follows:

$$\tilde{r}_t(a) = \begin{cases} r_t K & \text{if } a = a_t \\ 0, & \text{otherwise.} \end{cases} \tag{8.11}$$

Algorithm 8.3: Explore-first with a classification oracle

Parameter: exploration duration N, classification oracle $\mathcal{O}$

1. Explore uniformly for the first N rounds:
   in each round, pick an arm u.a.r.
2. Call the classification oracle with data points
   $(x_t, \tilde{r}_t(a) \in \mathcal{A})$, $t \in [N]$ as per Equation (8.11).
3. Exploitation: in each subsequent round,
   use the policy π_0 returned by the oracle.

Theorem 8.5. Let $\mathcal{O}$ be an exact classification oracle for some policy class Π. Algorithm 8.3 parameterized with oracle $\mathcal{O}$ and $N = T^{2/3}(K \log(|\Pi|T))^{\frac{1}{3}}$ has regret

$$\mathbb{E}[R_\Pi(T)] = O(T^{2/3})(K \log(|\Pi|T))^{\frac{1}{3}}.$$

Remark: ALG 8.3 works for any sampling scheme and corresponding unbiased estimator with $\tilde{r}_t(a) \leq K$

Proof:

$\tilde{r}(\pi)$ is an unbiased estimator of $\mu(\pi)$

$$\begin{aligned}
\mathbb{E}[\tilde{r}_t(a) \mid x_t] &= \mu(a \mid x_t) && \textit{(for each action } a \in \mathcal{A}\textit{)} \\
\mathbb{E}[\tilde{r}_t(\pi(x_t)) \mid x_t] &= \mu(\pi(x) \mid x_t) && \textit{(plug in } a = \pi(x_t)\textit{)} \\
\mathop{\mathbb{E}}_{x_t \sim D}[\tilde{r}_t(\pi(x_t))] &= \mathop{\mathbb{E}}_{x_t \sim D}[\mu(\pi(x_t)) \mid x_t] \\
& && \textit{(take expectation over both } \tilde{r}_t \textit{ and } x_t\textit{)} \\
&= \mu(\pi),
\end{aligned}$$

Now, let us use this estimate to set up a "clean event":

$$\{\mid \tilde{r}(\pi) - \mu(\pi) \mid \leq \texttt{conf}(N) \text{ for all policies } \pi \in \Pi\},$$

where the confidence term is $\texttt{conf}(N) = O\left(\sqrt{\frac{K \log(T|\Pi|)}{N}}\right)$.

Then, we have

$$\mu(\pi^*) - \mu(\pi_0) \leq 2\mathsf{conf}(N) \qquad w.h.p.$$

Total regret:

$$\begin{aligned}
\mathbb{E}[R_\Pi(T)] &\leq N + 2T\mathsf{conf}(N) = N + O(\sqrt{K \log(T|\Pi|)/N}) \\
&= O(T^{2/3}(K \log(|\Pi|T))^{1/3})
\end{aligned}$$

with choosing $N = O(T^{2/3}(K \log(|\Pi|T))^{1/3})$

Q.E.D

Oracle-efficient algorithm with optimal regret. A near-optimal regret bound can in fact be achieved with an *oracle-efficient* algorithm: one that makes only a small number of oracle calls. Specifically, one can achieve regret $O(\sqrt{KT \log(T|\Pi|)})$ with only $\tilde{O}(\sqrt{KT/\log|\Pi|})$ oracle calls across all T rounds. This sophisticated result can be found in Agarwal *et al.* (2014). Its exposition is beyond the scope of this book.

Learning from contextual bandit data

Problem: Policy evaluation and training

Input: N data points (x_t, p_t, a_t, r_t), $t \in [N]$.
Policy evaluation: estimate policy value $\mu(\pi)$ given a policy π.
Policy training: find policy $\pi \in \Pi$ that maximizes $\mu(\pi)$ over a given policy class Π.

Problem protocol: Contextual bandit data collection

For each round $t \in [N]$:

1. algorithm observes a "context" x_t,
2. algorithm picks a sampling distribution p_t over arms,
3. arm a_t is drawn independently from distribution p_t,
4. rewards $\tilde{r}_t(a)$ are realized for all arms $a \in \mathcal{A}$,
4. reward $r_t = \tilde{r}_t(a_t) \in [0,1]$ is recorded.

Policy Evaluation

Inverse Propensity scoring (unbiased estimator)

$$\texttt{IPS}(\pi) = \frac{1}{N} \sum_{t \in [N]:\ \pi(x_t) = a_t} \frac{r_t}{p_t(a_t)}. \tag{8.12}$$

Typo?

$\mu(\pi) = \mathbb{E}_{x \sim \mathcal{D}}[\mu(\pi(x)|x)]$

Lemma 8.6. $\mathbb{E}[\texttt{IPS}(\pi)] = \mu(\pi)$ for each policy π. Letting $p_0 = \min_{t,a} p_t(a)$, for each $\delta > 0$, with probability at least $1 - \delta$

$$|\texttt{IPS}(\pi) - \mu(\pi)| \leq O\left(\sqrt{\tfrac{1}{p_0}\ \log(\tfrac{1}{\delta})/N}\right). \tag{8.13}$$

Bernstein's

If we want:

$$\Pr\left[\ |\texttt{IPS}(\pi) - \mu(\pi)| \leq \epsilon \quad \text{for each policy } \pi\ \right] > 1 - \delta.$$ 1/poly(T)

How large should N be, as a function of M and the parameters? Taking a union bound over (8.13), we see that it suffices to take

$$N \sim \frac{\sqrt{\log(M/\delta)}}{p_0 \cdot \epsilon^2}.$$

where $M = |\Pi|$

Lemma 8.7. Assume rewards $\tilde{r}_t(\cdot)$ are chosen by a deterministic oblivious adversary. Then we have $\mathbb{E}[\texttt{IPS}(\pi)] = \tilde{r}(\pi)$ for each policy π.

Letting $p_0 = \min_{t,a} p_t(a)$, for each $\delta > 0$, with probability at least $1 - \delta$ we have:

$$|\texttt{IPS}(\pi) - \tilde{r}(\pi)| \leq O\left(\sqrt{\tfrac{1}{p_0}\ \log(\tfrac{1}{\delta})/N}\right). \tag{8.14}$$

This lemma implies Lemma 8.6. It easily follows from a concentration inequality, see Exercise 8.3.

Policy training can be implemented similarly: we define the fake reward (abusing notation)

$$\tilde{r}_t(a) = \mathbb{I}[a = a_t]\frac{r_t}{p_t(a_t)}$$

Then, we call the oracle $\mathcal{O}$

Corollary 8.8. Consider the setting in Lemma 8.7. Fix policy class Π and let $\pi_0 \in \text{argmax}_{\pi\in\Pi} \texttt{IPS}(\pi)$. Then for each $\delta > 0$, with probability at least $\delta > 0$ we have

$$\max_{\pi\in\Pi} \tilde{r}(\pi) - \tilde{r}(\pi_0) \leq O\left(\sqrt{\frac{1}{p_0}\log(\frac{|\Pi|}{\delta})/N}\right). \tag{8.15}$$

8.6 Contextual bandits in practice: challenges and system design