CS/CNS/EE 253: Advanced Topics in Machine Learning
Topic: Dealing with Partial Feedback #2 **Lecturer:** Daniel Golovin
Scribe: Chris Berlind **Date:** Feb 1, 2010

8.1 Review

In the previous lecture we began looking at algorithms for dealing with sequential decision problems in the bandit (or partial) feedback model. In this model, there are K "arms" indexed by $1, 2, \ldots, K$, each with an associated payoff function $r_i(t)$ which is unknown. In each round t, an arm i is chosen and the reward $r_i(t) \in [0, 1]$ is gained. Only $r_i(t)$ is revealed to the algorithm at the end of round t, where i is the arm chosen in that round; it is kept ignorant of $r_j(t)$ for all other arms $j \neq i$. The goal is to find an algorithm specifying how to choose an arm in each round that will maximize the total reward over all rounds. We began our study of this model with an assumption of stochastic rewards, as opposed to the harder adversarial rewards case. Thus we assume there is an underlying distribution $\mathcal{R}_i$ for each arm i, and each $r_i(t)$ is drawn from $\mathcal{R}_i$ independently of all other rewards (both of arm i during rounds other than t, and of other arms during round t). Note we assume the rewards are bounded; specifically, $r_i(t) \in [0, 1]$ for all i and t.

We first explored the ϵ_t-Greedy algorithm in which with probability ϵ_t an arm is chosen uniformly at random, and with probability $1 - \epsilon_t$ the arm with the highest observed average reward is chosen. For the right choice of ϵ_t, this algorithm has expected regret logarithmic in T.

We can improve upon this algorithm by taking better advantage of the information we have available to us. In addition to the average payoff for each arm, we also know how many times we have played each arm. This allows us to estimate confidence bounds for each arm which leads to the Upper Confidence Bound (UCB) algorithm explained in detail in the last lecture. The UCB1 algorithm also has expected regret logarithmic in T.

8.2 Exp3

The regret bounds for the ϵ_t-Greedy and UCB1 algorithms were proved under the assumption of stochastic payoff functions. When the payoff functions are non-stochastic (e.g. adversarial) these algorithms do not fair so well. Because UCB1 is entirely deterministic, an adversary could predict its play and choose payoffs to force UCB1 into making bad decisions. This flaw motivates the introduction of a new bandit algorithm, Exp3 [1] which is useful in the non-stochastic payoff case.

In these notes, we will develop a variant of Exp3, and give a regret bound for it. The algorithm and analysis here are non-standard, and are provided to expose the role of unbiased estimates and their variances in the developing effective no-regret algorithms in the non-stochastic payoff case.

8.2.1 Hedge & the Power of Unbiased Estimates

Back in Lecture 2, the Hedge algorithm was introduced to deal with sequential decision-making under the full information model. The reward-maximizing version of the Hedge algorithm is defined

as

HEDGE(ϵ)
1 $w_i(1) = 1 \qquad i = 1, \ldots, K$
2 **for** $t = 1$ **to** T
3 Play $X_t = i$ w.p. $\frac{w_i(t)}{\sum_j w_j(t)}$
4 $w_i(t+1) = w_i(t)(1+\epsilon)^{r_i(t)} \qquad i = 1, \ldots, K$

At every timestep t, each arm i has weight $w_i(t) = (1+\epsilon)^{\sum_{t' \le t} r_i(t')}$ and an arm is chosen with probability proportional to the weights. We let X_t denote the arm chosen in round t.

In this algorithm, Hedge always sees the true payoff $r_i(t)$ in each round. Fix some real number $b \ge 1$. Suppose each $r_i(t)$ in Hedge is replaced with a random variable $R_i(t)$ such that $R_i(t)$ is always in $[0,1]$ and $\mathbf{E}[R_i(t)] = r_i(t)/b$. We imagine Hedge gets actual reward $r_i(t)$ if it picks i but only gets to see feedback $R_j(t)$ for each j rather than the true rewards $r_j(t)$. We can find a lower bound for the expected payoff $\mathbf{E}\left[\sum_t b \cdot R_{X_t}(t)\right] = \mathbf{E}\left[\sum_t r_{X_t}(t)\right]$ as follows. First note that the upper bound on Hedge's expected regret on the payoffs $R_i(t)$ ensures

$$\mathbf{E}\left[\sum_{t=1}^{T} R_{X_t}(t)\right] \ge \mathbf{E}\left[\max_i \sum_{t=1}^{T} R_i(t)\left(1 - \frac{\epsilon}{2}\right) - \frac{\ln K}{\epsilon}\right]$$

Also note that for any set of random variables $R_1, R_2, \ldots, R_n$

$$\mathbf{E}\left[\max_i R_i\right] \ge \max_i \mathbf{E}[R_i]$$

One way to see this is to let $j = \operatorname{argmax}_i \mathbf{E}[R_i]$ and note that $\max_i\{R_i\} \ge R_j$, always. Hence $\mathbf{E}[\max_i R_i] \ge \mathbf{E}[R_j] = \max_i \mathbf{E}[R_i]$. Using these two inequalities together with $\mathbf{E}[R_i(t)] = r_i(t)/b$ we infer the following bound. Below, expectation is taken with respect to both the randomness of $R_i(t)$ and with respect to the randomness we used for Hedge.

$$\begin{aligned}
\mathbf{E}\left[\sum_{t=1}^{T} r_{X_t}(t)\right] &= \mathbf{E}\left[\sum_{t=1}^{T} b \cdot R_{X_t}(t)\right] = b \cdot \mathbf{E}\left[\sum_{t=1}^{T} R_{X_t}(t)\right] \\
&\ge b \cdot \mathbf{E}\left[\max_i \sum_{t=1}^{T} R_i(t)\left(1 - \frac{\epsilon}{2}\right) - \frac{\ln K}{\epsilon}\right] \\
&= \max_i \; b \cdot \mathbf{E}\left[\sum_{t=1}^{T} R_i(t)\left(1 - \frac{\epsilon}{2}\right)\right] - \frac{b \ln K}{\epsilon} \\
&= \max_i \sum_{t=1}^{T} r_i(t)\left(1 - \frac{\epsilon}{2}\right) - \frac{b \ln K}{\epsilon}
\end{aligned}$$

Hence

$$\mathbf{E}\left[\sum_{t=1}^{T} r_{X_t}(t)\right] \;\ge\; \max_i \sum_{t=1}^{T} r_i(t)\left(1 - \frac{\epsilon}{2}\right) - \frac{b \ln K}{\epsilon} \tag{8.2.1}$$

This indicates that even though Hedge is not seeing the correct payoffs, it still has *nearly* the same regret bound due to the linearity of expectation. The only difference is that the $\frac{\ln K}{\epsilon}$ term in the regret increases to $\frac{b \ln K}{\epsilon}$. This will turn out to be a very useful property.

8.2.2 A Variation on the Exp3 Algorithm

The idea here is to observe a random variable and feed it to Hedge, since the above analysis shows this will not hurt our performance. Define

$$R'_i(t) = \begin{cases} 0 & \text{if } i \text{ is not played in round } t \\ \frac{r_i(t)}{p_i(t)} & \text{otherwise} \end{cases}$$

where $p_i(t) = \Pr[X_t = i]$. Then $\mathbf{E}[R'_i(t)] = r_i(t)$. To use the above ideas we need to scale these random rewards so that they always fall in $[0, 1]$. Since $r_i(t) \in [0, 1]$ by assumption, the required scaling factor is $b = \min_{i,t} p_i(t)$. This suggests that using Hedge directly in the bandit model would result in a poor bound on the expected regret because some arms might see their selection probability $p_i(t)$ tend to zero, which will cause b to tend to ∞, rendering our bound in equation (8.2.1) useless. Intuitively this makes sense. Since we are working in the adversarial payoffs model, and lousy historical performance is no guarantee on lousy future performance, we cannot ignore any arm for too long. We must continuously explore the space of arms in case one of the previously bad arms turns out to be the best one overall in hindsight. Alternately, we can view the problem as controlling the variance of our estimate for the average reward (averaged over all rounds so far) for a given arm. Even if our estimate is unbiased (so that the mean is correct), there is a price we pay for its variance. To enforce the constraint that we continuously explore all arms (and keep these variances under control), we put a lower bound of γ/K on the probabilities $p_i(t)$. This ensures that $b = K/\gamma$ suffices. The result is a modified form of Hedge. In this algorithm, a variation on Exp3, each timestep plays according to the Hedge algorithm with reward

$$R_i(t) := R'_i(t)/b = \gamma R'_i(t)/K$$

with probability $1 - \epsilon$ and plays an arm uniformly at random otherwise. Formally, it is defined as follows:

EXP3-VARIANT(ϵ, γ)

for $t = 1$ **to** T

  $p_i(t) = (1 - \gamma)\frac{w_i(t)}{\sum_j w_j(t)} + \frac{\gamma}{K}$ $\quad i = 1, \ldots, K$

  Play $X_t = i$ w.p. $p_i(t)$

  Let $R_i(t) = \begin{cases} \frac{\gamma}{K}\frac{r_i(t)}{p_i(t)} & \text{if } X_t = i \\ 0 & \text{otherwise} \end{cases}$

  $w_i(t + 1) = w_i(t)(1 + \epsilon)^{R_i(t)}$ $\quad i = 1, \ldots, K$

Let $\mathsf{OPT}(S) := \max_i \sum_{t \in S} r_i(t)$ be the reward of the best fixed arm in hindsight over rounds in S, and let $\mathsf{OPT}_T := \mathsf{OPT}(\{1, 2, \ldots, T\})$ Using Equation (8.2.1), we get the following bound on

expected reward bound, where X_t is what we played on round t.

$$\mathbf{E}\left[\sum_{t=1}^{T} r_{X_t}(t)\right] \;\geq\; \mathbf{E}\left[\max_i \sum_{t\in\text{EXPLOIT}} r_i(t)\left(1-\frac{\epsilon}{2}\right)\right] - \frac{K\ln K}{\gamma\epsilon}$$

Here, EXPLOIT is the (random) set of rounds on which the algorithm exploited previous knowledge rather than explored[1]. It is not too hard to see that $\mathbf{E}[\mathsf{OPT}(\text{EXPLOIT})] \geq (1-\gamma)\mathsf{OPT}_T$. In effect, giving up the reward for each round with probability γ to explore should only cause us to lose a γ fraction of the static optimum OPT_T). Thus we get the following regret bound.

Theorem 8.2.1 *The algorithm above obtains expected reward at least* $\mathbf{E}[\mathsf{OPT}(EXPLOIT)]\cdot\left(1-\frac{\epsilon}{2}\right) - \frac{K\ln K}{\gamma\epsilon}$ *and so has expected regret at most* $\left(\frac{\epsilon}{2}+\gamma\right)\mathsf{OPT}_T + \frac{K\ln K}{\gamma\epsilon}$.

Noting $\mathsf{OPT}_T \leq T$ and balancing terms, we can optimize the bound by setting $\epsilon, \gamma = \Theta((K\ln K)^{1/3}T^{-1/3})$ for a regret bound of $O(T^{2/3}(K\log K)^{1/3})$.

Compared to the $O(K\log T)$ regret bounds in the stochatic reward setting, this is much worse. Ignoring the dependence on K, it means the average regret shrinks as $O(T^{-1/3})$ instead of $O(\frac{\log T}{T})$. This algorithm and analysis are not the best possible; As we discuss below, Exp3 achieves a $O(\sqrt{TK\log K})$ regret bound, and a lower bound of $\Omega(\sqrt{TK})$ is known for the adversarial payoff case.

8.2.3 The Original Exp3 Algorithm

The original Exp3 algorithm has only one parameter, γ, and is obtained by setting $\epsilon = e-1$ in our variant, i.e., Exp3(γ) $\equiv$ Exp3-Variant($e-1,\gamma$). Here is the psuedocode.

EXP3(γ)
1 **for** $t = 1$ **to** T
2 $\quad p_i(t) = (1-\gamma)\frac{w_i(t)}{\sum_j w_j(t)} + \frac{\gamma}{K} \qquad i = 1,\ldots,K$
3 $\quad$ Play $X_t = i$ w.p. $p_i(t)$
4 $\quad$ Let $R_i(t) = \begin{cases} \frac{\gamma}{K}\frac{r_i(t)}{p_i(t)} & \text{if } X_t = i \\ 0 & \text{otherwise} \end{cases}$
5 $\quad w_i(t+1) = w_i(t)\exp(R_i(t)) \qquad i = 1,\ldots,K$

Auer et al. [1] then prove the following regret bound for Exp3.

Theorem 8.2.2 *The expected regret of* EXP3(γ) *after T rounds is at most*

$$(e-1)\,\gamma\,\mathrm{OPT}_T - \frac{K\ln K}{\gamma}$$

where OPT_T *is the static optimum for the first T rounds.*

[1]To decide if a round t was an "exploitation" or an "exploration" round, let i be the arm chosen in round t, and flip a coin with bias $\gamma\cdot(Kp_i(t))^{-1}$. If it comes up heads, its an exploration round. Otherwise its an exploitation round. Proving $\mathbf{E}[\mathsf{OPT}(\text{EXPLOIT})] \geq (1-\gamma)\mathsf{OPT}_T$ is easy if you note that this can be done after all the rounds have been played.

With the optimum choice of ϵ it is possible to achieve a regret bound of $O(\sqrt{\mathrm{OPT}_T\, K \ln K})$.

8.3 Gradient Descent without the Gradient

Unbiased estimates are used in other algorithms in the bandit feedback model as well. For example, Flaxman et al.[2] have shown that it is possible to perform gradient descent in the bandit setting by getting an unbaised estimate of an n-dimensional gradient[2] from an observed (scalar) reward! See their paper and references therein for more on this topic.

References

[1] Peter Auer, Nicolo Cesa-Bianchi, Yaov Freund, and Robert E. Schapire. The non-stochastic multi-armed bandit problem. *SIAM journal on computing*, 32:48–77, 2002.

[2] Abraham D. Flaxman, Adam Tauman Kalai, and H. Brendan McMahan. Online convex optimization in the bandit setting: gradient descent without a gradient. In *SODA '05: Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 385–394. Society for Industrial and Applied Mathematics, 2005.

[2] They estimate the gradient of a smoothed version of the objective function, rather than the gradient of the objective function itself.