EE194/290S: ML for Sequential Decision under Uncertainty **Fall 2018**

Lecture 16: Introductory Bandits 1

Lecturer: Anant Sahai/Vidya Muthukumar *Scribes: Rishi Puri*

16.1 Basic Intro

Lemma 16.1 *The multi-arm bandit problem is used to describe systems such as this: You are in a casino filled with multiple slot machines all with different functions determining their payouts and probabilities. There are multiple levers, or arms(hence multi-arm), you can pull but at each turn you must choose to pull only one. In doing so you incur some loss/gain and learn something about just that arm. At the same time you pay an opportunity cost of what you could have learned about other arms.(As opposed to the case of sequence prediction, where we learned about loss incurred over all possible options) This dichotomy gives rise to a an exploration/exploitation trade off. This gives rise to the regret benchmark of* $R = \sum_{n=1}^{T} l_t(a_t) - \min_a(\sum_{n=1}^{T} l_t(a))$*, where the first summation represents the loss incurred from the sequence of actions actually chosen and the second is the loss that would have happened if we had chosen to pull only the best arm in hindsight.*

There are 3 problems encompassed in the multi-arm bandit situation:
1)Limited Information
2)Expensive Information
3)Balancing Between Exploring Exploiting

Proof: We will build up to a solution to the multi-arm bandit problem by tackling each of these 3 problems sequentially:

First we tackle the problem of limited information. To do so let us create a toy problem to help us understand.

Problem: we have $x_i \epsilon \{0,1\}, \vec{l_i} = [l_i(0), l_i(1)], \forall i = 1, 2, ..., T$
assume that at each turn you receive $\vec{l_i}$ with probability ε and you receive an erasure with probability $p = 1 - \varepsilon$.
This situation gives rise to a simple solution: conduct MWU except only update your weights when you receive $\vec{l_i}$ and do not update your weights when you receive an erasure. We just need to decide how to pick our learning rate η to achieve sub linear regret.
Recall that if we have full information and play for T rounds then we should pick our $\eta = \sqrt{\frac{2*\ln 2}{T}}$ in order to achieve $O(\sqrt{T})$ regret

So with limited information since we recieve on average $T\varepsilon$ losses for which we make the same number of weight updates. This leads us to set our $\eta = \sqrt{\frac{2*\ln 2}{T\varepsilon}}$ in order to achieve $O(\frac{1}{\varepsilon}\sqrt{T\varepsilon})$ regret In classic MWU we let $L_t(j) = \sum_{i=1}^{t} l_i(j) \Rightarrow w_t(j)\ \alpha\ e^{-\eta * L_{t-1}(j)}$ In the limited information version $\widetilde{L}_t(j) = \sum_{i=1}^{t} \widetilde{l}_i(j)$ where

$$\widetilde{l}_i(j) = \begin{cases} 0 & \text{if time i erased} \\ \widetilde{l}_i(j)/\varepsilon & \text{otherwise} \end{cases}$$

(where the $\frac{1}{\varepsilon}$ factor comes from compensating in expectation for all of the missed losses.)

$\rightarrow \mathrm{w}_t(j)\ \alpha\ e^{-\widetilde{\eta} * \widetilde{L}_{t-1}(j)}$. To find the best $\widetilde{\eta}$ the weights in both interpretations should be updated the same so $\eta * L_{t-1}(j) = \widetilde{\eta} * \widetilde{L}_{t-1}(j) \rightarrow \widetilde{\eta} = \sqrt{\frac{\varepsilon * 2 * ln(2)}{T}}$. Now we have a multiplicative weights

update system that updates on every turn to get the same result with a different learning rate.
Now to move onto the idea of expensive information we must first realize that if we had a system where each $l_i(j)$ can be erased independently with probability $1-\varepsilon$ we can use the $\widetilde{L}$ interpretation to achieve same results.

The idea of Expensive Information: we get to choose ε but we will pay $T\varepsilon * C$ (where C is a constant) of extra loss (which is the opportunity cost of exploration). Tying this into the prior interpretation we had $R \leq \alpha * \sqrt{\frac{T}{\varepsilon}}$
and we now pay an exploration cost in reality that we don't pay in the hindsight regret benchmark $\rightarrow R \leq \alpha * \sqrt{\frac{T}{\varepsilon}} + T\varepsilon * C$ In order to find ε that gives worst regret bound we use a mini-max hueristic and set $\alpha * \sqrt{\frac{T}{\varepsilon}} = T\varepsilon * C \rightarrow \varepsilon = \frac{\alpha^{2/3}}{C^{2/3}} * T^{-1/3}$ and if you plug this into regret we get a bounding on how well we will do regardless of what ε we choose $R \leq 2 * T^{2/3} * C^{1/3} * \alpha^{2/3}$

With this information we finally tackle the third problem and we do this in the setting of the actual multi arm bandit to give a general solution. Before we can do this we see that in the prior setting every arm had an independent probability of getting pulled, but in the actual multi arm bandit setting, this independence does not hold, as the occurence of one arm being pulled implies all others are not. However by using the union bound we know that the probability of more than one arm being pulled is $\leq \varepsilon^2$ and since epsilon is very small, this will have almost no effect on our expected regret.
Main Idea: Split time steps randomly into explore and exploit slots so that we have, in expectation, $T\varepsilon$ observations of actions and their losses and $T(1-\varepsilon)$ exploit slots where we throw away info received.
Let g be the number of arms. Then with probability $g\varepsilon$ you explore and you exploit the rest of the time. Start with initially uniform weights and at each time step either
Explore: Uniformly select one arm to pull at random and update that arms loss. Re-evaluate and normalize exponential weights.
or
Exploit: randomly sample an arm based on the weights and throw away losses.

Now that we have achieved our goal let us summarize our approach to achieving it. We first introduced the idea of limited information in which at every turn we lose all information with probability ε and showed how much regret our MWU on the scaled loss estimators achieved. In a small transition we noted that losing the entire vector of losses each turn behaves the same in our MWU algorithm as if each entry to the loss vector can independently be erased. Now that we had cemented this idea we added the next major point of paying an opportunity cost of information each time you choose to exploit. This gives rise to the inherent tradeoff in this setting: explore vs exploit because the more we exploit, the higher cost of expensive information but the more optimally we are playing based on the knowledge gained from exploring up to this point.

Excercises: Play with the IPYNB to see the effect of updating losses in both explore AND exploit slots. Play with the IPYNB to see what happens when we don't explore at all.