EE675A - IITK, 2022-23-II

Lecture 4: UCB and Thompson Sampling Algorithm

18th January 2023

Lecturer: Subrahmanya Swamy Peruru *Scribe: Akansh Agrawal, Aditya Ranjan*

We have so far discussed about the Successive Elimination Algorithm and performed the regret analysis for two cases : Instance Dependent and Instance Independent. In this lecture we will discuss yet another algorithm - UCB algorithm[2] and derive the regret for this case as well. Next, we will cover the Thompson Algorithm[2] which is a probability distribution based algorithm.

1 UCB Algorithm

Let's understand the following conversation to first get the gist of the algorithm.
$Teacher$: Consider a bandit in an imaginary world, who attacks the diamond trader having a treasure box with k-arms. The trader being from IIT is smart and designed the mechanism in his box that each arm when played yields the reward with some unknown distribution. Moreover, the maximum times the box can be played is bounded by some constant number. So can you guess what is the strategy bandit would apply apart from using successive elimination algorithm ?
$Student1$: Yeah..he will definitely ask the trader to tell the optimal arm to maximize the return.
$Student2$: But does the trader himself know which is the optimal arm ?
$Teacher$: The trader indeed knows but he may lie. IITians can be liars, can't they?
$Everyone$: Laughs...
$Student1$: So maybe some exploration is required at initial stages since there is uncertainty about the action-value estimate here .
$Teacher$: Right, but can we do something rigorous ?
$Student2$: May be ϵ-greedy would work.
$Teacher$: The drawback ϵ-greedy possesses is the random selection of an arm in between. Can we do better than this approach by giving weightage to the arms which has the potential to be the optimal arm, using what we learnt from Successive Elimination Algorithm?
$Students$: Hmmm...
$Teacher$: Alright, you guys can only think when it's about marks . Consider yourself as a dacoit and treasure to be the marks earned.
$Student1$: May be we can use Upper Confidence Bound to play the arm after doing the initial exploration.
$Teacher$: Bingo ! See it works. The marks drive you all crazy... So keeping the jokes aside, let's try to understand the algorithm our friend just proposed.

Upper-Confidence-Bound (UCB) Algorithm utilises the fact to adaptively select the arm by exploiting their potential of being optimal arm by considering their estimate and upper confidence

bound. The algorithm is formally stated as follow :

Algorithm 1 UCB

1. Play each arm $\in \mathcal{A}$ once.
2. For each round t, Play with arm a(t) $\in \mathcal{A}$ such that a(t) = $argmax_a UCB_t(a)$,

where $UCB_t(a) = \bar{\mu}_t(a) + \epsilon_t(a)$ and $\epsilon_t(a) = \sqrt{\frac{2logT}{n_t(a)}}$

1.1 UCB Regret Analysis

Essentially an arm 'a' will be played in round t after exploration if it's UCB is equal or greater than of the UCB of the optimal arm estimated so far.

$$UCB_t(a) \geq UCB_t(a^*) \tag{1}$$

$$\bar{\mu}_t(a) + \epsilon_t(a) \geq \bar{\mu}_t(a^*) + \epsilon_t(a^*) \tag{2}$$

$$\bar{\mu}_t(a^*) - \bar{\mu}_t(a) \leq \epsilon_t(a) - \epsilon_t(a^*) \tag{3}$$

Also we have ,

$$\Delta(a) = \mu(a^*) - \mu(a) \tag{4}$$

$$\Delta(a) \leq \bar{\mu}_t(a^*) + \epsilon_t(a^*) - (\bar{\mu}_t(a) - \epsilon_t(a)) \tag{5}$$

$$\Delta(a) \leq 2\epsilon_t(a)) \tag{6}$$

$$\Delta(a) \leq 2\sqrt{\frac{2\log T}{n_t(a)}} \tag{7}$$

Note that this bound we achieved in equation (7) is similar to the one we have in Successive Elimination Algorithm as well and therefore we will get similar bound on Regret for this case as well. Let's quickly derive it as well.

1.2 Derivation for the Instance-dependent bound for UCB

$$\Delta(a) = \mu(a^*) - \mu(a) \leq 2\epsilon_t. \tag{8}$$

$$\Delta(a) \leq \mathcal{O}(\epsilon_T) = \mathcal{O}(\sqrt{\log T / n_T(a)}) \tag{9}$$

$$n_T(a) \leq \mathcal{O}(\log T / (\Delta(a))^2) \tag{10}$$

Therefore the expected total regret contributed by an arm a is :

$$R(T;a) = \Delta(a).n_T(a) \tag{11}$$

$$R(T;a) \leq \Delta(a).\mathcal{O}(\log T/(\Delta(a))^2) \tag{12}$$

$$R(T;a) \leq \mathcal{O}\left(\frac{\log T}{\Delta(a)}\right) \tag{13}$$

The total regret, $R(T)$, is given as :

$$R(T) = \sum_{a \in \mathcal{A}} R(T;a) \tag{14}$$

$$R(T) = \sum_{a \in \mathcal{A}} \mathcal{O}(\log T/\Delta(a)) \tag{15}$$

$$R(T) \leq \mathcal{O}(\log T) \sum_{a \in \mathcal{A}} \frac{1}{\Delta(a)} \tag{16}$$

$$R(T) \leq \mathcal{O}\left(\frac{K \log T}{\Delta}\right) \tag{17}$$

where: $\Delta = \min_a \Delta(a)$.

As we saw in the Successive Elimination as well that this bound gives us a tight bound only when Δ is large enough otherwise this bound will become loose and therefore instance independent bound is required .

1.3 Derivation for the Instance-Independent bound for UCB

We saw above,

$$R(T;a) \leq \mathcal{O}\left(\frac{\log T}{\Delta(a)}\right) \tag{18}$$

$$R(T) = \sum_{a \in \mathcal{A}} R(T;a). \tag{19}$$

We'll can bound $\Delta(a)$, by some confidence interval ϵ and analyze the regret for the 2 cases:

If $\Delta(a) < \epsilon$,

$$R(T;a) \leq \epsilon\, n_T(a)$$

If $\Delta(a) > \epsilon$,

$$R(T;a) \leq \mathcal{O}\left(\frac{\log T}{\Delta(a)}\right) \leq \mathcal{O}\left(\frac{\log T}{\epsilon}\right)$$

Therefore we have,

$$R(T) = \sum_{a \in A} R(T;a) \tag{20}$$

$$R(T) \leq \mathcal{O}(\epsilon.T + \frac{K}{\epsilon}\log T) \tag{21}$$

Since this is same as we did in Successive Elimination, so we get $\epsilon^* = \sqrt{\frac{K}{T}\log T}$ on minimizing the RHS, which on substituting gives :

$$R(T) \leq \mathcal{O}\sqrt{KT\log T}. \tag{22}$$

Therefore we get the same results for UCB as we did in the Successive Elimination Algorithm. For more info please follow the lecture notes from previous lecture as well.

2 Maintain Probability Distributions

In all our previous discussions regarding the Bandit problem, we tried to get some estimate about the true means $\mu(a)$'s of arms. In the Explore-Then-Commit and ϵ-Greedy algorithms, we estimated the means using a point estimate: the sample mean $\overline{\mu}(a)$. To capture more information, in the Adaptive and the UCB algorithms, we maintained confidence intervals for $\mu(a)$: $(LCB(a), UCB(a))$.

We can go even further: keep track of the *entire* probability distributions of $\mu(a)$'s! We will have some prior distributions initially, and every time we play an arm a, using Bayes' rule, we update the distribution of $\mu(a)$ to its posterior distribution, which is just our belief of the new distribution of $\mu(a)$ after we get a new observation (the reward obtained).

Let us illustrate how this works using a coin-toss game.

2.1 Coin Toss game using Bayesian Inference[1]

Suppose we are given a biased coin such that $\mathbb{P}(\text{Head}) = p$, where the probability p is unknown to us, and we wish to estimate it. A frequentist person will approach this problem as follows: toss the coin n times, if Head appears H_n times, then $p \approx \frac{H_n}{n}$. In fact, in the limit $n \to \infty$, the ratio $\frac{H_n}{n}$ will converge to p.

Here is the Bayesian approach to the problem. First of all, have some prior idea about the parameter p. What we mean by this is: have a prior distribution about the value of p. For example, if initially we have no idea about p's value then we can choose it to be $\text{Uniform}(0,1)$, i.e. all values

in $[0, 1]$ equally probable. The prior distribution could, in general, be some distribution in the range $[0, 1]$ (as we know p cannot lie outside that), for example, $B(\alpha, \beta) = c(\alpha, \beta)\theta^{\alpha-1}(1-\theta)^{\beta-1}, 0 \leq \theta \leq 1$ (Beta distribution). Here $c(\alpha, \beta)$ is a constant depending on α, β which is needed to make the total probability sum up to 1.

Given our prior distribution, we now toss the coin once. Let the result be some $r \in \{0, 1\}$ (1 for Heads, 0 for Tails). Bayes' rule states that for events A, B:

$$\mathbb{P}(A|B) = \frac{\mathbb{P}(B|A)\mathbb{P}(A)}{\mathbb{P}(B)} = \frac{\mathbb{P}(B|A)\mathbb{P}(A)}{\sum_{A'} \mathbb{P}(B|A')\mathbb{P}(A')}$$

If we take event A as the distribution of p (i.e., event $p = \theta$) and event B as our observation (result of the coin toss), we can write this as:

$$\mathbb{P}(p = \theta|\,\text{Observation}\, r) = \frac{\mathbb{P}(\text{Observation}\, r|p = \theta)}{\mathbb{P}(\text{Observation}\, r)} = \frac{\mathbb{P}(\text{Observation}\, r|p = \theta)}{\int_0^1 \mathbb{P}(\text{Observation}\, r|p = \theta')\mathbb{P}(p = \theta')\mathrm{d}\theta'}$$

(Here we abuse the notation a bit, $\mathbb{P}(p = \theta)$ can mean the continuous probability distribution function of p at θ. Also the summation is replaced by an integral due to the continuous nature of p's distribution.)

We can see that $\mathbb{P}(\text{Observation}\, r|p = \theta)$ is easy to find, it is θ if $r = 1$ (Heads), and $(1 - \theta)$ if $r = 0$ (Tails), i.e. $\mathbb{P}(\text{Observation}\, r|p = \theta) = \theta^r(1-\theta)^{1-r}$. Thus, if we have our prior distribution $\mathbb{P}(p = \theta)$, we can find out the posterior distribution given the observation: $\mathbb{P}(p = \theta|\,\text{Observation}\, r)$.

If we have a Beta prior (even $\text{Uniform}(0, 1) = B(1, 1)$) $B(\alpha, \beta)$, then our posterior is:

$$\mathbb{P}(p = \theta|\,\text{Observation}\, r) = \frac{\theta^r(1-\theta)^{1-r} \cdot c(\alpha, \beta)\theta^{\alpha-1}(1-\theta)^{\beta-1}}{\int_0^1 \theta'^r(1-\theta')^{1-r} \cdot c(\alpha, \beta)\theta'^{\alpha-1}(1-\theta')^{\beta-1}\mathrm{d}\theta'}$$

Note that the denominator is just a constant (this is true in Bayesian inference in general, and not just this coin toss example), so we can write our posterior in a proportional manner (neglecting the constants).

$$\mathbb{P}(p = \theta|\,\text{Observation}\, r) \propto \theta^{\alpha+r-1}(1-\theta)^{\beta+(1-r)-1}$$

i.e., our posterior distribution is $B(\alpha + r, \beta + (1 - r))$ ($B(\alpha + 1, \beta)$ in case of Heads, $B(\alpha, \beta + 1)$ in case of Tails). In the case of Beta priors and Bernoulli coin tosses, each time we toss, our distribution belief for p changes from the previous one in the above manner.

If our initial prior is $B(\alpha, \beta)$, we toss n times, out of which there are H_n heads and T_n tails, then our final posterior distribution of p is $B(\alpha + H_n, \beta + T_n)$. Given this final posterior distribution of p based on our observations, we can now get an estimate of p as per our needs, for e.g. the posterior mean, posterior median etc.

2.2 Thompson Sampling Algorithm

Now, let us consider the Bandit Problem with Bernoulli Rewards, each arm gives a reward of 1 with some probability (fixed for that arm), and a reward of 0 with the rest probability.

$$\left.\begin{aligned}\mathbb{P}(R(a)=1)&=\mu(a)\\ \mathbb{P}(R(a)=0)&=1-\mu(a)\end{aligned}\right\}\text{ for each arm } a\in\mathcal{A}\ (0\leq\mu(a)\leq 1)$$

Note: It can be seen that $\mathbb{E}(R(a))=\mu(a)\cdot 1+(1-\mu(a))\cdot 0=\mu(a)$, which is why we retain the notation of the Bernoulli probabilities as $\mu(a)$.

To begin with, we have may some initial idea about the behaviour of each arm, i.e. some prior distribution for each parameter $\mu(a)$, which we take to be a Beta distribution $B(\alpha_0(a),\beta_0(a))$. Even if we have no prior information on $\mu(a)$, we can take it to be the $\mathrm{Uniform}(0,1)$ distribution, i.e. $B(1,1)$ $(\alpha_0(a)=1,\beta_0(a)=1)$.

Every time (say at round t) we choose an arm a (we will see the selection policy soon) and play it, we get a reward $R_t(a)=r$. Based on the reward, we update $\mu(a)$'s distribution to the posterior distribution, which can be found using Bayes' rule.

$$\mathbb{P}(\mu(a)=\theta|R(a)=r)\propto\mathbb{P}(R_t(a)=r|\mu(a)=\theta)\cdot\mathbb{P}(\mu(a)=\theta)$$

Note that $\mathbb{P}(R_t(a)=r|\mu(a)=\theta)=\theta^r(1-\theta)^{1-r}$, for $r\in\{0,1\}$ (Bernoulli distribution). If our prior was $B(\alpha,\beta)$, like we saw for our coin toss example, our posterior distribution is $\mathbb{P}(\mu(a)=\theta)\propto\theta^r(1-\theta)^{1-r}\cdot\theta^{\alpha-1}(1-\theta)^{\beta-1}=\theta^{\alpha+r-1}(1-\theta)^{\beta+1-r-1}$, i.e., $B(\alpha+r,\beta+(1-r))$. So based on the reward r, the posterior of $\mu(a)$ for the chosen arm a is updated as above.

How to choose the arm a? The heuristic is: for any $a\in\mathcal{A}$, play arm a with the probability that it has the highest $\mu(a)$, i.e. $\mathbb{P}(\mu(a)=\max_{a'\in A}\mu(a'))$. This probability is known to us based on our *own* beliefs about the distributions of each $\mu(a)$.

For example, if there were only two arms a,b, then we choose arm a with probability $\mathbb{P}(\mu(a)>\mu(b))$, and arm b with probability $\mathbb{P}(\mu(b)>\mu(a))$. Here the probability distributions for $\mu(a)$ & $\mu(b)$ are our most up-to-date distribution beliefs for both.

Now, these probabilities themselves are hard to calculate analytically. Hence, we use a simple sampling algorithm which will do our job. The idea is as follows: for each arm a, sample a point $\widetilde{\theta}(a)$ from the distribution of $\mu(a)$, $B(\alpha(a),\beta(a))$. Then, select the arm with the highest $\widetilde{\theta}(a)$. As $\widetilde{\theta}(a)$'s and $\mu(a)$'s come from the same distribution, this will indeed select a particular arm a with the required probability. This is known as the Thompson Sampling algorithm.

Thus, the final algorithm is presented as follows:

Algorithm 2 Thompson Sampling

At $t = 0$, have prior Beta distributions for $\mu(a)$ for each arm $a \in \mathcal{A}$: $B(\alpha_0(a), \beta_0(a))$
for ($t = 1$ to T) **do**
    For each arm $a \in \mathcal{A}$, sample $\widetilde{\theta}_t(a) \sim B(\alpha_{t-1}(a), \beta_{t-1}(a))$
    Play arm $a(t) = \arg\max_a \{\widetilde{\theta}_t(a)\}$
    Update the posterior of the arm $a(t)$ based on the reward $r_t \in \{0, 1\}$ received in round t: $\alpha_t(a(t)) = \alpha_{t-1}(a(t)) + r_t, \beta_t(a(t)) = \beta_{t-1}(a(t)) + (1 - r_t)$.
end for

References

[1] S. Agrawal. Lecture 4: Introduction to thompson sampling, multi-armed bandits and reinforcement learning.

[2] A. Slivkins. *Introduction to Multi-Armed Bandits*, volume 7. 2022.