Bandit Algorithm (Online Machine Learning)
Prof. Manjesh Hanawal
Industrial Engineering and Operations Research
Indian Institute of Technology, Bombay

Lecture - 32
Optimism in the Face of Uncertainty

Where were we last time? We start we completed the ETC algorithm right and then, we started discussing what are the other possibilities.

(Refer Slide Time: 00:31)

So, one option we said like is to the greedily. So, what the greedy algorithm is going to do? It is going to simply play an arm I_t which is simply arg max i of or maybe like this t minus 1. So, what are the estimates you have for each arm till round t minus 1, just play the arm with which has the highest mean value and we just said that this is going to be a bad thing right, if I am going to just select the arms greedily.

What was the example we gave? So, we say that suppose you take a Bernoulli; let us say Arm 1 is and Arm 2. So, let us say this is Bernoulli with 0.5 and Bernoulli with 0.8 and now, if we are going to do this let us say in t equals to round 1, you played let us say I 1 equals to 1 and you observed value 1.

So, initially you do not have anything right. Like what you are going to do in all these things to have at least some estimate of this, we need to have at least 1 sample right for

ith arm. So, let say initially you play each one of them once. So, that is I 2, you play 2 and you got a value of 0 for this.

So, this is t equals to 2 and after that, what is going to happen? In t equals to 3, now you start applying this. When you start applying this, which is the arm you are going to play? You are going to look at; so, this is what? This is your sample right. For arm 1, you got 1 and arm 2, you got a value of 0.

So, when you come to round 3, the estimate you have for arm 1 mu 1 hat is going to be 1 and mu 2 hat is going to be 0 right, that is the value you have so far and because of that, which is the one it is going to pick? So, it is going to pick I 1 equals to 1 and it is going to observe whatever value, let us say it observed now this time 0.

Now, in round t equals to 4; now in round t equals to 4, what is that it is going to observe? Now, it has played now arm 1 and arm 2; arm 1 twice, once it observed 1, once it observed 0. So, the average value of this is going to be half; whereas, that for arm 2 is still going to be 0.

So, which one is going to pick again? And even, let us say if it observes 0 here; then, in t equals to 4 what happens? Still the mean value observed for I_1, arm 1 is going to be some positive number right; whereas, the mean value so far observed for arm 2 is still 0 because of that you will continue to play I_1, even though it has a smaller mean than the other arm.

So, if you are going to do like this, greedily you will be stuck in a bad arm. So, now, we said that because it is very likely that in this case, the initial samples matter and if only going to start selecting the arms greedily, it is very likely that you can stuck in the bad arm. So, we have to some kind of once in a while, come out of this greedy selection. Another option you said is epsilon greedy.

So, in this, what we will do is we are going to some epsilon is given to you as an input. Now, I_t is going to be this arm whatever it is. But these we are going to do with probability some epsilon let us say and the other thing is select an arm uniformly at random.

Now, this we are going to do with probability 1 minus epsilon or maybe let us say this let us say this is 1 minus epsilon, this is epsilon. So, what you will do? You will take a coin which has a bias epsilon, you are going to toss it in each round. If you toss it let us say and it shows up head with probability 1 minus epsilon, let us say.

Then, you are going to greedily select the arm; otherwise, let us say it shows up tail with probability epsilon. Then, what are going to do is we are going to just select one of the arm to uniformly at random ok. So, because of that, so now, if you go back to this, suppose let us say when you come to round t = 5 and there is some chances that you may be end up in this situation right; instead you toss a coin in round 5 and let us say which is a with a probability epsilon, whatever it is. Then, you are in this case.

You could be either in this case or this case right line you ended up in this case, then you are going to select an arm uniformly at random and in that case, it may happen that instead of selecting 1, you may end up selecting 2 and in that case you may end up with a sample 1. So, that will possibly will take you out of keep on selecting arm I_1 itself all the time ok.

So, this epsilon greedy in a way tries to work arm; but what was the problem with epsilon greedy, is it like if you then, the question here also first is how to choose epsilon right? If you choose epsilon to be 0, this is same as greedy algorithm right; all the time, we will be selecting an arm greedily.

But suppose, let say if we make epsilon 0, if we make epsilon, then we will be always selecting an arm uniformly at random ok. If you are going to select an arm uniformly at random, it is like on an average, we will be selecting each arm equal number of times. So, you have only selected the optimal arm only 1 by K fraction of that rounds, other times you have not selected the optimal arm.

So, that is also going to be bad and the regret is going to be linear in that case. Now, how you how to choose this epsilon ok? Ultimately, you were interested in algorithm which gives sub linear regret right. Now, how to choose that epsilon ok. One has to choose this epsilon in appropriate fashion maybe like you do not want to fix it, epsilon fixed throughout; but may be like as the round t increases, maybe make it a function of t right like make it let it decrease as t increases.

So, because you know that as you have more rounds, you have more samples, you do not need to go and do the uniform selection. Then, it is enough like if you start selecting greedily, once you have enough samples and good estimates of your arm means right. So, then may be one option is to let you choose start choosing epsilon t that start decaying as t tends to as t increases.

But that the question then, again the question is how to decay make a decay as t increases ok. So, you will analyse this one of the assignment problems ok. What are the other options ok? So, as we said epsilon here is kind of deciding how much to explore and how much to exploit right. So, this when you are here, you are trying to exploit greedily and when you are in this you are exploring different arms.

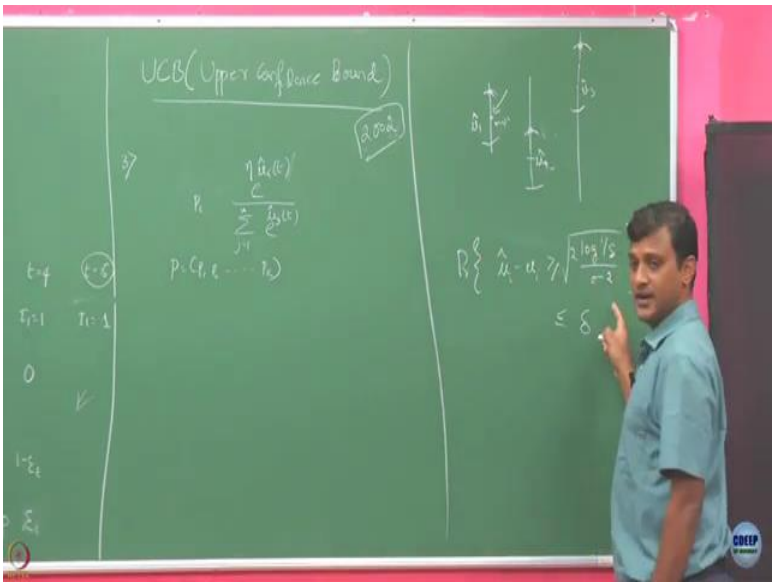
So, if you chose your epsilon t to be very small, then most of the times your just trying to exploit. On the other hand, if epsilon t is high or close to 1, you will be trying to explore either of the extremes are bad, you have to choose your epsilon t appropriately. So, ideally how we want to do?

So, when t is small that is in the initial rounds, maybe it is good to explore right because you do not know about anybody, just go and sample them uniformly each one of them. But as t increases, you have got some samples, now you have some estimates of the arms. Then, maybe you want to just restrict, you want to focus more on exploiting the things right.

So, this is like you kind of has to say like when is good to explore and then, after that at what point you should start exploiting. It is a algorithm exactly did that right. What it is it did it? Initially explored for certain number of rounds and one is it is confident or whatever like once it felt that this is enough number of samples, then it started exploiting.

Now, the question is it possible that instead of doing this exploration and exploitation separately, is it possible to combine them ok? So, we will see some algorithm based on that today, it is called Upper Confidence Bound based algorithm.

(Refer Slide Time: 12:05)



So, before we move in to this, do you think of any other possibilities for algorithm selection here. Like so far, we were always talking about selecting the arm in deterministic fashion right. You just to go and select which is the highest mean. Why not do it a bit probabilistically, the way we did it in the adversarial setting.

So, how about another option, maybe I will just write 3 here. How about saying that I am going to choose it is equals to I. So, I will define this. Some constants here. When I have these estimates, currently let us say in round t, I have this estimate. Then, I am going to define the probability for arm i to be like this and let us say this is P_i. So, this gives me a probability vector P on the k arms.

Now, select an arm according to this probability distribution. So, this is very similar to what we did it in weighted majority algorithm right. So, do you think this algorithm should perform better, if you do probabilistically like this? Now, let us come back to this upper confidence bound algorithm.

So, little bit history about this algorithm. So, we this problem of studying this Bandit problems, it has been there for quite some time like maybe in early in 1930s, there were some studies like especially in the in treatments right like when we have a set of treatments which you are applying on patients. You do not know which patient which treatment is going to be effective on the patients.

Now, we are exactly in this case right nowadays like Corona Virus, we do not know what is the treatment. But everybody is now want to identify a drug which gives you the most effective treatment, but none of us know what is the right treatment ok. Now, what how people will do? They have to apply some set of candidates, drugs will be there and have to test them and if that drug happens to cure, good.

That is a good drug for me; but that drug can also kill because if the I do not a priory that is going to be effective or not. So, it can also kill the person. So, what I want to do is if I apply a bad drug, it could effectively cause harm or maybe it is just neutral also, it may not cause harm. But what is my goal? I want to quickly identify the drug which gives maximum number of people right and how I will do this?.

Like I apply this drugs on patients and see that whether that person responds positively or negatively to this drug. Now, it is not necessary that if I apply this drug on one person and if he responds, this drug is the good one ok. So, we have to find a drug which is effective on an average on most number of people and that is exactly this problem right.

So, this I have its set of drugs, their effectiveness is like how good they are, like they being effective as certain is related to certain mean value and I want to identify a drug which has the highest effect and that I want to do as quickly as possible.

Now, such kind of problems are pretty much like these (Refer Time: 16:47) like kind of problems have been there for quite a long time and in earlier, some of these problems have been studied and some algorithms were proposed. But only like late 1980s when people thought ok, this is an interesting problem, let has been studied, but we need to understand its theoretical guarantees, what kind of things.

So, in 1980s when we talk about its lower bound, we will discuss that paper like people talked about this and see what kind of performance guarantees, we will get like how to model this problem; if I am going to model this problem, this drug testing problem as this stochastic multi armed (Refer Time: 17:29) problem; what kind of performance guarantees I get and people give algorithm ok.

But those algorithms bit complicated to analyse, but people gave some algorithms and also showed that you cannot do better than this. They established the lower bounds ok. But only in the late 2000, I think it around 2003, this algorithm is proposed called UCB,

Upper Confidence Bound which are kind of easy like this algorithm is not complicated as you will see and also it is kind of easy to analyse.

So, till that point what are the algorithm we had, they are kind of very complicated also like in terms of the algorithm and also to analyse ok. Only after this 2003, this algorithm called UCB is introduced in a way this algorithm is not so old compared to its history like the first the medical diagnosis, I just discussed those are like 1930s paper and once which gave some algorithms initially they are like 1980s paper and after that like after a long gap, this algorithm is come in 2003 in that way it is?

Student: 2002.

2002. So, in that way this is not far away right about less it will be like some still less than 20 years ok. Now, what is the idea of this algorithm? So, this algorithm is basically saying that it is kind of trying to exploit the fact that if you have the estimates and you know you if you can you have some confidence ok, this estimates if you are going to take the maximum value of these estimates.

So, this not the maximum value. Suppose, let it is just saying that you estimate a bound on your estimates such that with high probability, it contains the true value. You estimate the bound, some range and try to treat the upper value of that bound as the actual value of your mean value and now then, apply this greedy algorithm on that.

So, just how it says suppose let us say this is arm 1. Suppose, after a certain number of samples at some point, I have an estimate of that and let us say I got the estimate of arm 1 to be mu 1 hat. I can also come up with a confidence bound on this value. So, let us say this is an upper confidence bound and this is a associated lower confidence. So, this will be symmetric about this point right.

And similarly, and we know that this confidence interval is such that the true value of the mean will be somewhere in this interval, the true value the mu 1 will be in this interval with high probability. So, that is how this interval is built.

You do similarly for each arm. So, may be for arm 2, you estimated that this is my estimated value and you have some confidence interval in that. This is the upper confidence bound and let us say the arm 3 in this case, you ended up getting this estimate

and you have this confidence interval on this ok. So, you this confidence interval depend on how many samples you had right. These are the confidence intervals. The confidence interval the length depends on how many samples you had to estimate that particular arm ok.

So, if for some arm, you have lesser number of samples that its interval will be so smaller and if some arms you have more number of samples its interval is going to be larger. Now, what it is saying that fine, you have this intervals for each of these arms, treat this point as the actual the true means at that particular time and just select play the arm which has the highest value of this upper confidence bounds.

So, it is saying that at any point, whatever the observation you have made about your environment through the samples and the estimates you have, pretend that they are as good as possible to be close to the true values of the environment and pretend that like and be optimistic about these values.

And then, just take these values are the true values and play and apply the greedy thing on that ok. It is saying that whatever the observations we have current till now, treat them the upper confidence interval and upper confidence bound you have on each one of them to be the true values and just apply your greedy algorithm again.

Now, the question is why does this idea work and how to what is the right how to take this what is the right confidence intervals I should be taking right like ok. Now, I am just going to first thing is how to construct these confidence intervals. We know how to construct these confidence intervals based on our concentration bounds anything so far ok.

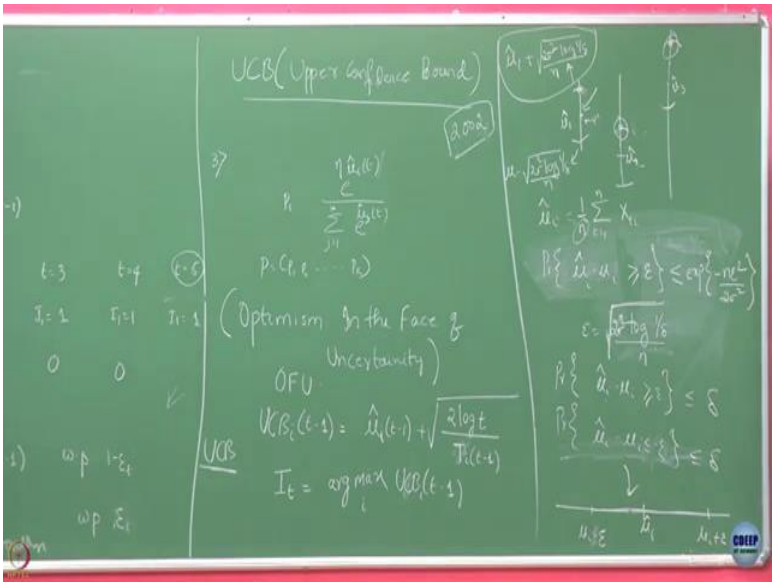
So, what we have known so far? What we know is probability that mu hat minus mu greater than. Was it 2 here to the denominator?

Student: 2 log.

2 log 1 by delta. So, if I have this, what is this probability is upper bounded as? It is delta right. So, what we are saying that is if, but we need to assume that all my random variables, the distributions when you centre them, they are all sigma Gaussian right. Then, we have this result ok. When my random variables are my distribution. So, let us

say this is for ith arm, not really. There was also m here somewhere in this ok. Let us redo this right, what we have.

(Refer Slide Time: 26:08)



So, what we said probability that mu hat minus mu. So, what is mu i hat? It is the let us say it is computed from n samples ok. So, again mu i hat is 1/nsum_{i=1}^{k}X_{ti}. Let us say my mu i hat, I was able to calculate based on my n samples drawn from that ith arm.

Now, we said that this probability is being greater than epsilon is upper bounded by exp minus n epsilon square by 2 sigma square right. Did we say this is a bound? Is this said to is there in the right place? Now, set this epsilon to be. So, if you are going to set your epsilon to be like this, why? So, if I am going to set my epsilon like this, so what is this quantity is going to be? Epsilon square right. So, two gets you are saying it should be at numerator like this ok.

If I have this, then I know that this probability that mu i hat minus mu i being greater than or equals to epsilon is upper bounded by delta. So, because of that it says that my and similarly, I can also write probability that mu i hat minus mu less than or equals to epsilon is also; so, I want this to be. So, this is a absolute value right, if I am going to take this to be this beyond that and I want this to be less than or equals to minus epsilon and this is also delta ok.

So, it is saying that if my mu hat if I calculate it somewhere here and if I take my mu i hat plus minus epsilon and mu i hat plus epsilon here. So, my true value is going to be somewhere in this interval with probability taking into both it is like some 2 delta right.

Now, this is once I have some n samples, I can come up with a interval like this, where I can say that if I estimate this mu mean to be mu i hat like this, I can guarantee that within this interval, the true mean will lie with probability that is going to be this is 1 minus 2 delta; not 2 delta, but is 1 minus 2 delta.

So, now based on that I can compute my confidence intervals here right. So, what I will do is if I have n samples, I am going to take in this case this upper guy to be mu 1 plus I am going to take this 2 sigma square log 1 by delta divided by n and similarly, this guy chi is going to be mu minus 2 sigma square log 1 by delta by n.

So, I can construct this upper and lower confidence bounds on my estimates using this results ok, based on my concentration bounds ok. Now then, I can use this quantity here, the upper confidence bound to be the current estimate whatever the value for my arm 1 and similarly, this for arm 2 and this and for arms 3 and then, just decide which one gives me the highest value and play accordingly. So, that is why this upper confidence, this kind of idea is called as what is this? Optimism in the face of uncertainty.

So, you are trying to be optimistic here right like the you are facing an uncertain environment; but whatever the current is the best values you could get, you are trying to be optimistic by taking their upper confidence value and here you are trying to be optimistic in the sense that you are trying to be like environment is going to behave as if these are the true values and then, you are going to play apply your greedy value on that. So, based on this idea, the UCB algorithm; so, it is as I said it is going to is based on optimism.

So, most of the algorithms that we are going to study for the stochastic bandits, they are going to be based on this idea of optimism in the face of uncertainty and now once, we have this what is your algorithm is going to be? Now, your algorithm is going to be in every round you are going to compute the UCB value upper confidence bound value of this arms. So, how we are going to compute?

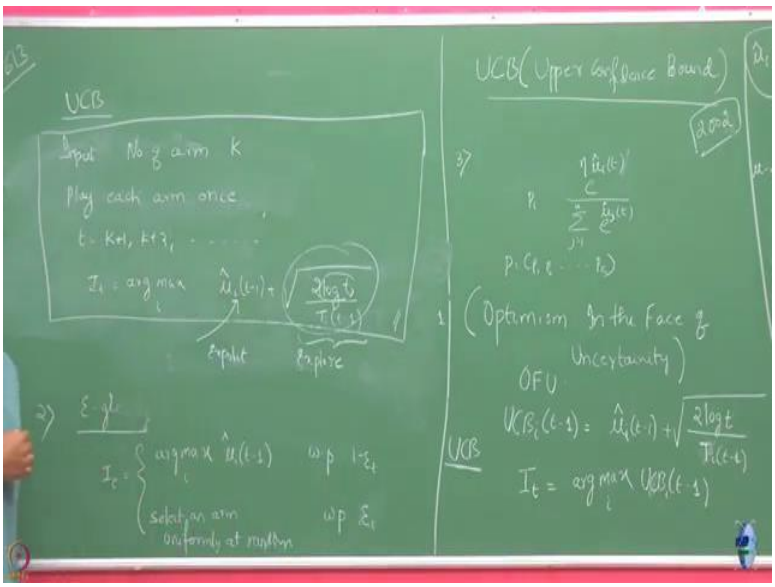
UCB of arm i in rounds let us say t minus 1, I am going to define it as its estimate plus. So, let us assume hence forth that the sigma square is 1 that is where assuming all my distribution to be 1 sub Gaussian ok. So, this is just for simplicity. So, if it is not, if it is sigma square known, you can appropriately scale your bounds ok. But for time being assumes sigma square is 1. So, I am going to take it as √(2 logt).

So, far instead of this 1 by delta here, I am going to replace it by t, you will see that why later and then, so what we do? I think we use N. What was N i? N i of t a minus 1. So, this we defined as number of samples we got for arm i till round t minus 1 right. It was Ti we defined it as.

So, here it is we assumed that this is computed based on n samples, but we do not know right how many samples we have till round t minus 1. The number of samples we have we are going to denote it as Ti of t minus 1 and now the algorithm is we are going to select it which is arg max of i UC B i of t minus 1.

So, just let me formally write this algorithm. As you can see here, this term makes sense only when there is this T i is at least 1, that is we have at least 1 sample. If we do not have any sample, this term is 0 and it can it is not defined.

(Refer Slide Time: 36:56)



So, the UCB algorithm is input is let us say number of arms that is K is given to you and for t equals to 1, 2 like this. What you are going to do?

Now, before that play each arm once. So, in the first K rounds, you are going to pull each arm once and then, for after that you will be starting with K plus 1, K plus 2 rounds. In the subsequent rounds, you are going to I_t equals to arg max of ok. So, little bit discussion about I said that when I started talking about UCB, we said like why not combine exploration and exploitation right.

Now, the question is this by doing so, is it combining exploration and exploitation? Suppose, let us say some arm so far you have not played many times, for that arm what is Ti is what? Number of arms total number of rounds total number of times you have played that arm till t minus 1 right.

If you have not played this i till round t minus 1 many times compared to the other arms, theta is going to be smaller right and because of that compare to the other arms, the guy which has smaller value of Ti is going to make this quantity large for that arm ok. So, because of that whatever the value of the mu i that arm has, if this Ti is going to be smaller, this quantity is going to be larger because of that the sum is going to be larger ok.

So, what does this mean? If you have not played this arm i many times that is you have did not observe many samples of it, you want to. Then, this value could be larger and because of that you may be this algorithm may be forced to choose that arm to play in that round ok.

If you if it if that is the case if this guy and if it so happens that all the arms has been played enough number of times ok, then what comes to dominate maybe in that case all everybody has almost the same value of this, what now differs for them is the mean values. The in the and in that case, the guy which has the highest value of the estimated guy maybe picked.

So, this algorithm is kind of doing the same thing right, this is like you say that this part is saying that exploit because the one which have the highest value of these estimates maybe you want to choose that. But this part here is saying no, you have to explore also because if some guy has been not played many times, I am going to give make him larger and because of that the sum will be larger ok.

And it is not that this guy is you have to keep on increasing this all the time, the number of samples has to be keep on increased. There you see that there is also a log t term there in the numerator ok, this is also slowly increasing right. As t increases, this is also t t increasing ok.

So, it is not necessary that if you have not played, anyway if you have not played this arm many times, it is going to make this guy larger and even let us say if you have for some reason you did not play it and or if you played it enough number of times, this is not going to contribute much to that. But still there is a log t quantity here which keeps on increasing

So, because of this if because this term log t is continuously increasing, if you do not play a certain arm many times that this denominator becomes small. Then, also this term may dominate these terms right and you will be playing that arm and ok, now consider the case that you got good estimate for all the arms.

Now, the question is whether this algorithm will start stop exploring at any time? Like what happened in ETC algorithm right it takes it observe the samples till sometimes and after that it was only getting stuck to the work. Will it happen in this algorithm? Will it so happen that after sometime, this exploration part will not have that much say in the sum, only these guys have a say that is you have got enough number of samples for all the arms.

So, then this way if you have enough number of the arm for all the arms right, then this T i becomes large for all the arms and then it will make this quantity small right. Then, only the say comes from this exploitation part; but that is not going to happen right. Because there is a log t term here in which this numerator is also continuously increasing.

So, even because of that even if you go further down into the time, even if you have many samples for that particular arm i, it may happen that this term may dominate this term and you may be still going and observing samples from the arms which may not be optimal.

So, I am saying that because of this kind of specific thing we have, it is not the UCB will never stop playing such suboptimal arms. Even if it has identified let us say the

exploration never stops. This terms keep on contributing to the sum and algorithm will keep on it will never get stuck to 1 arm. But deliberately this increase is slow enough which is not like it is increasing fast, it is increasing very slowly right, this log t is increases very slowly right.

So, because of that when you have many T i's many many samples for each of the arms that because this is in the denominator, it is going to make this small this term smaller, but there is a log t term, there it is going to make this larger at a very slow rate. Because it is increasing logarithmically intake.

So, because of that the algorithm will never get stuck into 1 arm, it keep on exploring the other arms; but the exploration may be at a very slow rate ok. Because you do not want to explore very too frequently right because in that case you will be maybe playing suboptimal arms too many times fine. So, this is the algorithm, it has these properties. Now, the question is what performance guarantees it gives?