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

Lecture - 31
Regret Analysis and ETC

(Refer Slide Time: 00:17)

So, we have this bound, right, ok. Now, let us see my input parameter to the algorithm is what? We said that is going to be the m, right. Where is m? What is m? m is the number of place of each of the arm before I commit to somebody, ok.

Now, how should we select this m? Suppose, if I select m to be small. Is it good? Why? Right. So, if your m is going to be small, we are not going to have enough samples, right. So, because of that the estimates may not be good and with few number of samples and the estimate with them if you commit to one arm maybe that may not be the best one you are going to get, right.

Because you have only few arms maybe the estimates may not be good and because of that the one which is the best one may not come up with empirically best. Somebody who is not actually best may end up empirically best and you may end up playing that for the rest of the duration. So, it is going to be bad.

Why not then choose m large? Suppose, we set m to be large there we have large, large number of samples we have a very good estimate after that we will commit to a good arm, but before that we have wasted so much of time in collecting the samples, right, when you are collecting the samples you may be you are also collecting bad samples. So, you have been wasting lot of your time to identify that. So, both of them are not good, right.

Then, how to choose m? That is exactly captured in this bound also, right. Suppose, you choose m to be small the quantity $m(\sum_{i=1}^{k}\Delta_i)$ is going to be small. So, this is what? This is basically the loss you incur in the initial exploration phase. Now, this part comes from the exploration part and this part $(n-mk)(\sum_{i=1}^{k}\Delta_i\exp(-m\Delta_i^2/4))$ from come the commit part.

If you choose m to be small $m(\sum_{i=1}^{k}\Delta_i)$ is going to be small. But what about $(n-mk)(\sum_{i=1}^{k}\Delta_i\exp(-m\Delta_i^2/4))$? This can be potentially large, right because it is already saying that this probability if you have small number of samples this probability can be large that you will end up making a mistake. So, because of that the regret you are going to incur the expected regret you are going to incur in the commit phase can be larger.

On the other hand, if you want to make $(n-mk)(\sum_{i=1}^{k}\Delta_i\exp(-m\Delta_i^2/4))$ is small the only way you can make $(n-mk)(\sum_{i=1}^{k}\Delta_i\exp(-m\Delta_i^2/4))$ is small is make this m large, if you are going to make this m large then $m(\sum_{i=1}^{k}\Delta_i)$ going to take a hit; the regret you are going to incur in the exploration phase. So, you see that already one we have to balance this how to explore, how much we have to put our resource in exploration and how much we have to put in my expectation.

So, then how we are going to choose this m? m is an input, right. You see that if I increase m this guy is going to take a hit, if I going to decrease m this guy is going to take a hit. So, why do not why treat this m even though m is an integer, I take this bound as a function in m and try to optimize it over m and try to find a value that minimizes this upper bound, right. So, can we do that just. So, here m is a linear quantity here in this, right and here this is exponential in m. Anyway this is going to be convex, but this is quantity is decreasing in m.

So, see that this is going to be let us say convex function in m, you need to verify that and then let us say it is going to achieve minima for some particular m. Can we find that m by just differentiating and equating it to 0? What is that (Refer Time: 05:09) value?

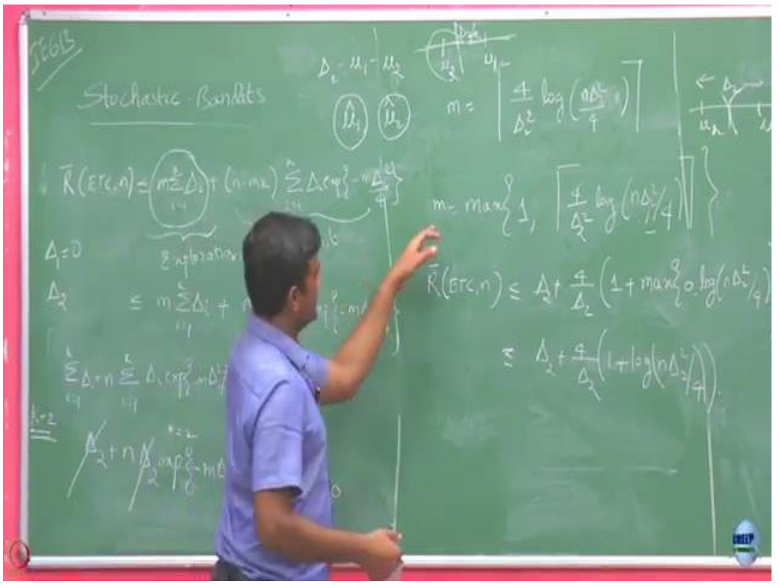
So, if you just treat it as variable in m differentiate and find it what is the m you are going to get that maximizes this. So, ok, before that, so this is going to make a slightly complicated, right; this term $(n-mk)(\sum_{i=1}^{k}\Delta_i \exp(-m\Delta_i^2/4))$, because this I know is go into I mean we if right now we I say you verify whether this is a convex function, but we do not need to even go there. Let us say we know this guy is convex, right; the only the exponential minus of m. But this product with this I do not know right now whether this is going to be a convex function.

So, let us simplify this. Instead of that I am going to make it an upper bound, I will just ignore this term and just take an n here, simplify this. Now, the second term the only m dependency comes in this part, right. I have removed this m and I got an upper bound. Now, I know that this quantity here the second term is also convex in m, right because exponential minus m it is a convex in m, if you treat m as a continuous. So, this is convex this is linear. So, this guy is now convex function.

Just differentiate it and find what is the m that minimizes this. So, if you just differentiate it we will going to get in m; so, differentiate and equal to 0, ok. So, let us make one more simplification. Let us set this k to be simply 2. Let us consume there are only two arms, ok. So, then the expression here becomes what? Now, because I have only two arms the first quantity Δ_1 is going to be what? By our definition what is Δ_1? It is going to be 0, right. And I will have Δ_2.

So, in this summation here it is going to be Δ_2 plus n times Δ_2 times exponential minus. And also, in the summation I only need to worry about the second term because the first terms make this entire thing 0. So, I have now we get this simplification. And if I now want to equate it to 0, this can get rid off and then if I am going to optimize find solution for this, then m is going to be $4/\Delta_2^2 \log(n\Delta_2^2/4)$, ok. So, this is at whatever the value is.

(Refer Slide Time: 09:03)



But, now coming to m, m I want it to be an integer, right. Even though I differentiated it rating m as if its m is a continuous variable, but m is an integer. So, what I will do is whatever the value of this I am going to take the ceil of this. So, even that value happens to be a fraction then it is going to give me some integer value, ok.

And also is it possible that this quantity inside this ceil can be negative, right. So, what I am doing is this; what is inside n? Delta 2 whole square by 4, right. Suppose this delta 2 is so small such that this quantity here happens to be less than 1. If this quantity happens to be less than 1 then what is you are going to get ? Then log of that quantity is going to give you negative value, right.

So, negative value, means the ceil is going to be what? ceil of a negative value some integer, right, but some integer value. But I want m to be at least 1, right because m is the number of samples. So, what I will do is I am going to redefine this m to be max of 1 times. So, for the two arm case what is the best way to set this m? It is like this. If you go into set it like this, ok.

Now, let us plug it back in this; if you are going to plug it back the value of m in this expression, you will end up with. So, I am only doing it for the case when k equals to 2. Already written there. So, if you simplify this I am directly writing after simplification you can verify this. What is that?

Student: (Refer Time: 12:53).

4 by. So, we are getting reciprocal of this. Yeah, I think it is (Refer Time: 13:07) reciprocal, I made a mistake. No, I think the inside is n delta 2 by square.

Student: (Refer Time: 13:19).

It is ok, right. This is what we have written. So, this is what the finally of, after putting back this a value of m in this expression for the case k equals to 2 you are going to get. But I think there has to be some correction here this quantity. So, finally, if you are going to plug it back the value of m like this you will end up this quantity and you will see that.

How is this regret depends on n? So, let us say; so if I just simplify this upper bound ignoring all these terms I am just going to keep the ones which are relevant. So, this is the exact expression, but I will just simplify this. So, you will see that now this regret is logarithmic in n, right. Yeah.

Student: (Refer Time: 14:59).

Yes.

Student: (Refer Time: 15:10) So, we do not know (Refer Time: 15:12).

We do not know.

Student: (Refer Time: 15:15) something (Refer Time: 15:17).

We will come to that. So, I am just saying that somehow, if you are going to set m like this you are going to get a regret bound which is in this fashion, if you just plug in, ok. So, now, just look into that.

Now, this regret bound how it is like? In n this is logarithmic in n, right. Now, coming to what he is saying fine, if you are going to set an m like this, good, you have ended up with a logarithmic regret like this which is definitely sub linear. But to certain m like this what you need to know? Delta 2, right. So, what is delta 2?

That is the gap between mu 1 and mu 2. Even though my current algorithm does not know which one is the optimal, arm 1 is optimal or arm 2 is optimal, but what is need to know the separation between them, that is delta 2, right. So, if it knows separation between them, it now can settle exactly like it can know how much to explore and then when to start, when to commit, right.

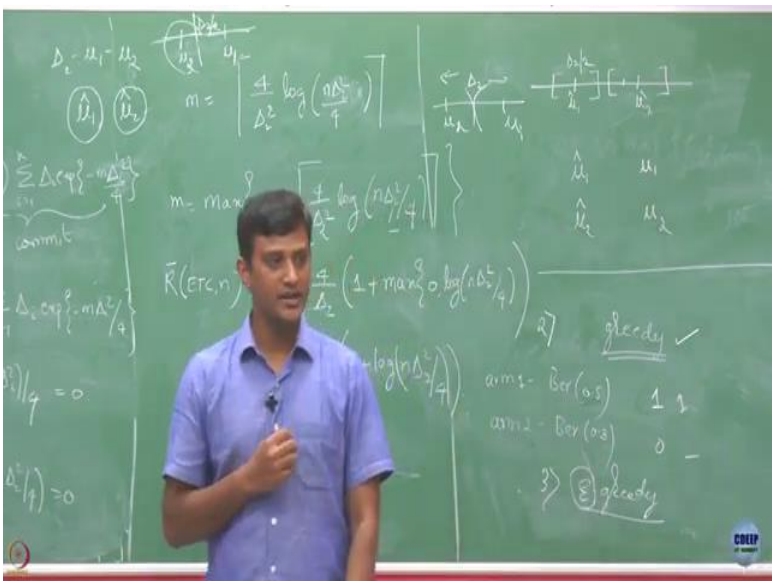
So, then it is natural way, right. Does it make sense like why? If I tell the algorithm, what is the gap between the first arm and the second arm that is the separation between those two, it kind of decide, right when how much samples it need to explore. This is because it has to estimate the parameters and if it already knows that the separation between them is delta 2, all it needs to ensure is the accuracy with which it is going to estimates the parameters that accuracy have that error happens to be less than delta, right.

So, again, what we are saying this delta 1 is mu 1 (Refer Time: 17:52). So, what my algorithm is doing is? My algorithm is estimating mu 1 hat and mu 2 hat, ok. Suppose, it is able to estimate now I know that somewhere mu 1 is here and mu 2 is here sorry, mu 1 we are assuming to be larger, right. So, mu 1 is here and mu 2 is here and the gap between them is we are going to call it as delta 2, right.

Suppose, my algorithm estimates the mean value such that the whatever the estimated value I have that the true value of this mu 2 will be contained within that delta 2 around this within delta 2 approximation. So, instead of delta let us make it delta by 2. So, that it will be whatever my estimated value is going to be below this and whatever my estimated value of mu is above this quantity plus delta 2 by 2, right.

So, if this I can estimate my mu 1 hat and mu 2 hat within delta 2 by 2 approximation of that is true value, then I know that when I am going to compare this with this, I am not going to make mistake, ok.

(Refer Slide Time: 20:00)



So, let me rephrase this. What my algorithm is doing? My algorithm is estimating mu 1 hat and also it is estimating mu 2 hat and it is finding which one is the bigger one of these two, right. Now, I know that the true value of them that the true value of this mu 1 and mu 2, the difference between them is delta 2, ok.

Suppose, if I have this mu 1 hat here and I let me estimate this guy within delta 2 by 2. So, now, I know that my true mean is somewhere in this interval with high confidence, right, like I know like with some confidence it is going to be. And then, I have another mu 1 mu 2 hat which is also like this and I know that my mu 2 the true value of mu 2 is again going to be are within this interval with high confidence.

Now, if I am going to compare these two quantities their separation will not be more than there will be at least separated by distance of delta, right and that is also the actual separation between mu 1 and mu 2. So, because of that if I can know the true difference between mu 1 and mu 2, I can a priori decide to what accuracy I should be estimating my mu 1 and mu 2.

And now here we are saying, if the true gap is between mu 1 and mu 2. So, again coming back to this, I know that this gap is delta 2. If I can ensure that my estimate of this mu 1 two somewhere falls below this and my estimate of mu 1 remains above this, then when I am going to compare the estimates their ordering will not change, right. So, because of that the one with the highest mean will be declared as the highest mean. If I am going to estimate those values within this delta by 2 error part.

So, that is why once I know my delta 2 and it is saying that if I am going to set my m to be this many number of samples, I am going to make a very small error only this much of error in fact. What is that exponential part; where we remove? Yeah; this much part of the error in separating these two through their estimates, ok.

So, naturally as long as I tell you the actual values differ by certain amount that will kind of already give you a hint to what accuracy I should be estimating each of these arms, right. And that is what it is saying like, ok. If you are going to set m like this I am going to set my number of samples like this. With this I will get a error bound which is of this format which happens to be leading to regret which is logarithmic in n here, ok.

So, good we have logarithmic regret here in n. But the bad part is I need to tell you what is this delta 2, right, to set m in this fashion that is I need to tell with the difference between arm m sorry; if it in the two case itself like arm 1 and 2 what is the gap between them and I need to tell. But we may not know this a priory, right. All we are assuming that each of these arm has certain distributions. We do not know those means, so we also do not know what is the gap between them.

We are interested in algorithms where we even do not want we do not have this knowledge of this. Even without this can we get something like this, some regret bounds which are logarithmic in m, ok. So, what other options we have there? We have this explore then commit algorithm which said that yes if you tell me the gap between the best arm and the suboptimal arm, I will somehow give you suboptimal regret, but I need extra information.

What is that extra information I need to tell you? This delta 2. If you tell you delta 2 I know exactly how many rounds I need to explore and after that I will do a commit, that commit will almost identify the, right arm with high probability. So, then this exponent commit needs to tell you the gap between mu 1 and mu 2, right. I do not want that; what other options I have, ok.

Other possibility is called epsilon greedy or may be just greedy. So, what is greedy? You just go and sample each arm once and after that you just or may be instead of once maybe sample each one of them certain number of rounds and after that commit to the one which has gives you the best empirical arm; is that; that is greedy. After that in every round you just you the one which gives you the empirically best you play that and you see whatever you get in the next round you update all of them means and play that. That is a greedy version of this, right.

Instead of committing to one arm after certain number of rounds to explore initially and after that you start playing an arm greedily. In every round you pick the one which has the highest empirical mean. Will that be good? Why. So, why not if ETC- explore and commit algorithm has bought you this level with some knowledge of delta 2 we can get a logarithmic regret, why not if you continue to do greedy instead of committing why not that is a bad idea, ok.

So, if I start doing greedy in every round instead of exploring I just get sample from each of them once and then start doing greedily, is that a good thing or bad thing to do? Ok. You have understood my question? You are going to sample each arm once and after that you start selecting the arm greedily in every round.

Student: (Refer Time: 28:25).

Yeah, you take anything you want and see in which case it is good and is there a case where it is going to be bad.

Student: (Refer Time: 28:34) practice (Refer Time: 28:36).

Yeah. So, along that lines suppose let us say I can take two case, arm 1 is simply let us say Bernoulli some value 0.5, let us say it is just half and let us say arm 2 is Bernoulli 0.8, ok. Which arm has highest mean in this? Arm 2, right. Its mean is going to be pointed. Suppose, you pick one arm one sample from this and another sample from this, and in that case it may happen that you may get sample one from this and sample 0 from this, right, it is possible, ok. Now, because of that when you want to play in the second round which arm you will going to pick?

Student: 1.

You are going to pick 1. And it may happen that again you get sample 1 from that, ok. And now and in the third round what you are going to do? Ok. Just forget this. So, I am going to write let us say in the first round you got sample 1 from this and 0 from this, then what you are going to do in the second round?

Student: (Refer Time: 30:01).

You are again going to choose arm 1, right. And what you are going to do; and you did not get any sample from this. So, now you have only one sample from this which is 0, while from the other samples you have got from arm you have at least one. Whatever the average you are going to get you are average will be greater than 0, right.

So, you will be always playing then arm 1, you will never get to arm 2 even though it has highest mean. So, because of that if you are going to do greedily you are never going to get the optimal, it is pretty much possible that we will miss out the optimal 1 and because of that you are regret could be linear.

So, the other option is you do not play the empirically best in every time. You know that you could be missing the other ones. So, every time before you, every time with some probability you go and explore, because of that you may end up in this case may be also playing this one and get a sample from this and because of that your average about that could improve.

Then, the question is how to set this epsilon, right, ok. So, how to set this and how this algorithm going to work, you will do it in an assignment question. And, there we will ask you to discuss how to choose this epsilon and there are some regret bounds like this for it, ok.

Let us stop here.