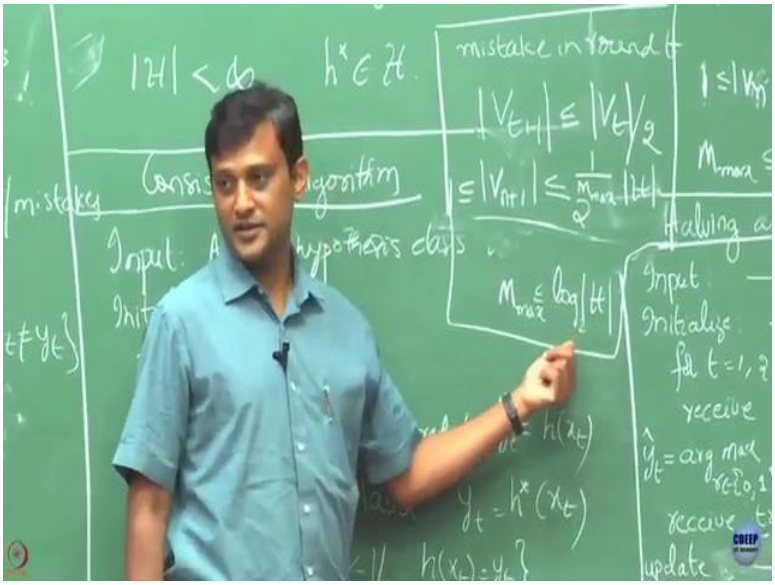
Bandit Algorithm (Online Machine Learning)
Prof. Manjesh Hanawal
Industrial Engineering and Operations Research
Indian Institute of Technology, Bombay

Lecture – 06
Online Learnability

So, now the question is, fine. I went from cardinality of log(x) to log of cardinality of h. Maybe I can do further, maybe like type just gave you two algorithm, right. Then, if then we may ask the question is it possible like I will get something better.

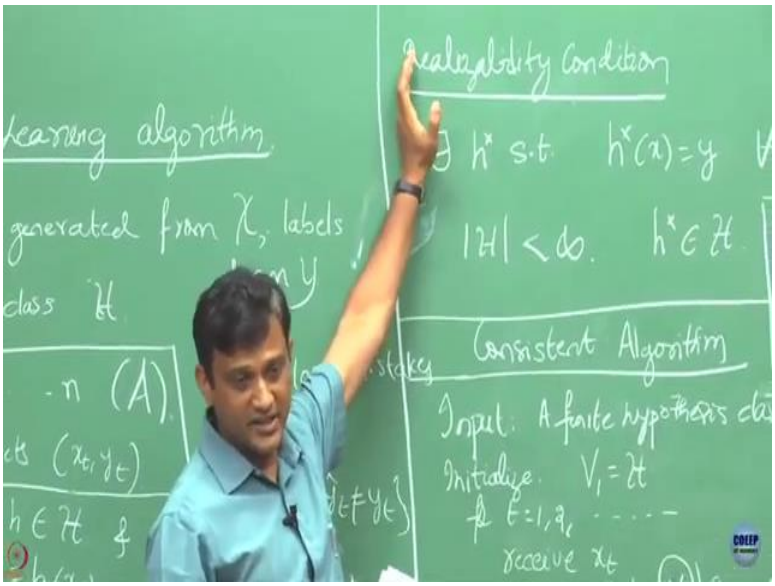
(Refer Slide Time: 00:45)



Maybe something better could be just like I can say, instead of $\log_2 |H|$ is it like $\frac{1}{2}\log_2 |H|$ if I can get that algorithm is still better, right. That is giving that is making smaller mistake.

Now, the natural question when we have this, this algorithm better, this algorithms is used to be still better, still this algorithm to be still better. So, how much better we can do? Is there a limit to that? So, that limit will be decided by what? Of course, but who is you are the one who is picking. There is an environment and you are learner, right. You are learning an environment.

(Refer Slide Time: 01:30)



If the environment become more and more, more and tougher, right. So, as I said earlier if I have not put this realizability condition, there is no way you could have gotten all these things, right. You would environment could have made you incur loss in every round. So, of course what kind of bounds we are going to get it depends on how tough our environment is. We had restricted our environments capability by putting this realizability condition here, ok fine.

Even once we have put this realizability condition and we are restricted the power of environment, we missed one to say, under this scenario what is the best I can do, what is the best bound I can achieve, is it that $\log_2 |H|$ is the best I can do or something I can go beyond this, ok.

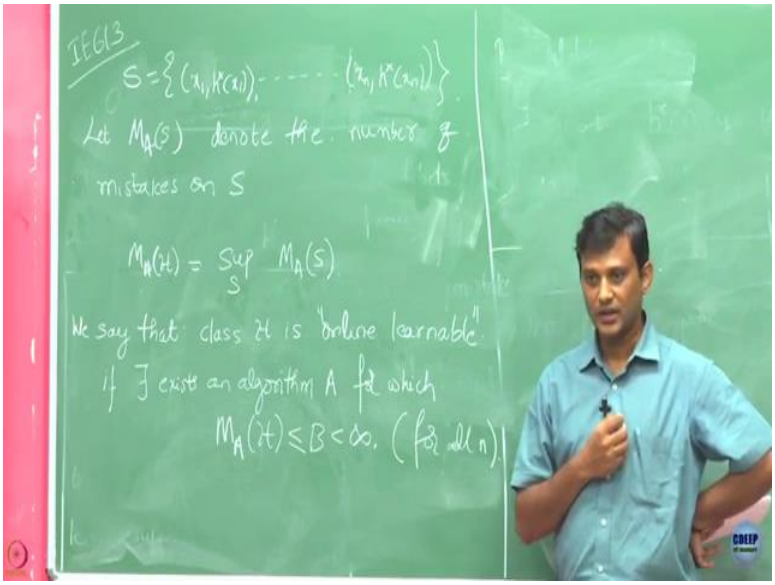
So, to do this let us introduce some notion and notice that this bounds I am giving you is irrespective of how the environment is choosing x_t s, ok. Even though I have said that there is a fixed h^* that is going to assign label I have not made any assumption that how this h^* itself is chosen. The h^* chosen could be very complicated and because of that you would may not easy for you to figure out what is that h^*, ok.

So, to understand that what is the best we can do, maybe if we can give a lower bound on this. If we can say that, no matter what algorithm you are going to use you will be going to incur at least this much of loss that means, that is like a dead end for me, right. Like no matter how intelligent I am how best my algorithm is I will be going to suffer this much

of loss. Then, we will see that if at all we can derive that then we will see that whether whatever the upper bound I got was that how good it was compared to the lower bound, ok.

So, to do that we will just formally introduce the notion of mistake bounds and online learnability, ok.

(Refer Slide Time: 03:55)



Let us say this is one sequence from the environment you faced, where in round in the first round environment generated this pair, x_1 and the associated label $h^*(x_1)$ like that and in the nth round this was the one.

So, notice that this is not the same sequence you are going to face every time your in algorithm, if you are going to restart your algorithm the sequence seen by you or algorithm could be altogether different, ok. So, I am just saying let us say this is one sequence that your algorithm faced.

And let me denote $M_A(S)$ denote the maximum number of mistakes sorry, the number of mistakes your algorithm made on this sequence S, ok. So, for all my talk all our discussions were going to fix this small n. We are going to assume that my algorithm is for run for some fixed number of rounds that is going to be n. What my interest is the number of mistakes I am going to make.

And now I do not want to I will be not interested in giving a bound on a particular sequence. I would like to give a bound which is irrespective of what sequence I am going to see, right because I do not know what is the sequence my algorithm is going to face, ok. So, then I am going to denote. So, this is the mistake my by my seen made by my algorithm A on a sequence S.

Now, I am saying that I am putting it against all possible sequences and see what is the maximum number of mistakes it is going to make on any of this sequences. So, whenever I am going to take the worst case scenario here, right. I am looking at what is the toughest sequence you faced so far.

So, toughest sequence is the one on which you made the largest number of mistakes. So, that I am going to denote by $M_A(H)$. So, H is my hypothesis class, A is my algorithm. So, this is the number of maximum number of mistakes made by my algorithm A while learning my hypothesis class H.

Let me rewind this I said that taken S which has this n points in this and I am going to call this $M_A(S)$ to be the number of mistakes made on sequence S. And now I am going to take this supreme or S to be the maximum number of mistakes I made in any sequence.

And here I do not need to restrict my S to be of size n, this could be of arbitrary length. You may have run it for n equals to 100 or you may be you may have run it for n equals 1000 or may be n equals to 1 million.

Now, we are going to say that this hypothesis class is online learnable, if there exists some algorithm A. I do not know what algorithm is this for which I should be able to bound this guy. What is this guy? This is the maximum number of mistakes I am going to make on any sequence, if I can bound it by some B which is a constant and finite if that is the case then I am going to say online learnable, ok.

Now, my two algorithms like consistence algorithm and my halving algorithm were they online using them if I have a finite hypothesis class H, can that final hypothesis class be made online learnable using this algorithms, ok. So, like put alternatively if I have a hypothesis class which is finite in size, is it online learnable according to this definition? Let us assume this, this right now we are in that realizability assumption. Is this online learnable according to this definition?

Student: Yes.

So, what was B for my consistence algorithm?

Student: Cardinality of H (Refer Time: 10:59) H (Refer Time: 11:00).

If H is finite then cardinality H is finite and I can take that value has to be my B, right. So, I have a bound which is independent of what was my sequence length and for the halving algorithm, I got $\log_2 |v|$ which I can take it to be B. So, that is why it is taken.

So, as long as I have a hypothesis class which is finite in size and good. According to this definition I am learnable. It is not like if I continue to use that algorithm forever I will make an large number of errors, at some point you stop making errors because your number of mistakes is bounded by a finite number. After that you are going to make no more mistake.

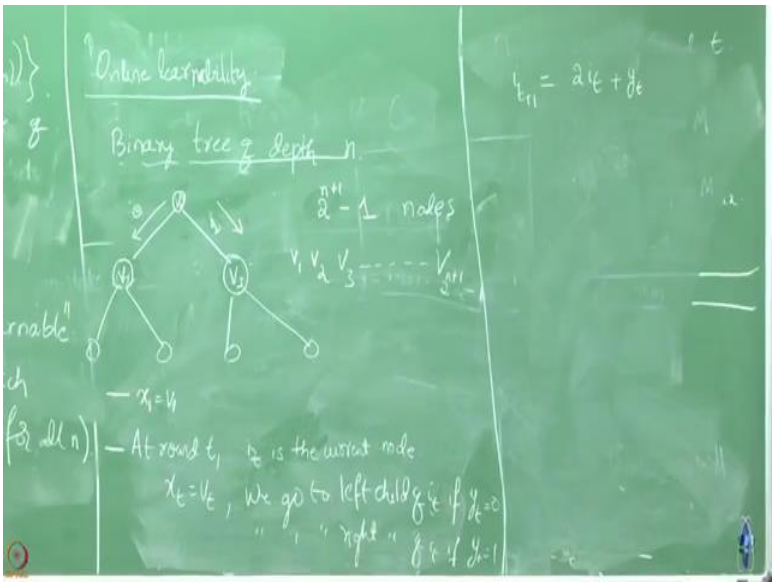
Or just think of like if I have an algorithm you start applying on it, and you keep on applying it forever, after some point that algorithm will not make mistake, right because you have ensured that its mistake bound is finite.

So, now you have to now answer the question what is the best we can do, right. Is this whatever B we got earlier like |H| and $\log_2 |H|$ is that something better than we can do better or if that is the case what is the what is that we should aim for.

So, for that we now look a lower bound on the number of minimum number of mistakes that the environment can force on me because the environment is the one against whom we are learning, right. If the environment becomes complicated maybe it can force more errors on me.

But what is the maximum number of errors it can force on me? And can I also ensure that whatever it can force maximum number of errors I will not make more errors than that. What are been enforced on me which is unavoidable, ok, fine, that much errors I will make, but no more than that, ok.

(Refer Slide Time: 13:23)



So, then we will talk about what we call as. So, any questions so far about this algorithms and the notion of whether hypothesis class is online learnable?. So, see how we are stating this results under what assumption like, if we do not have this realizability assumption these are all incorrect. We will, in the next class we will relax that assumption like, if we do not have that realizability assumption what is that we should look for, ok.

So, to understand this what is the lower bound we are going to get on this mistake, we have to understand what is the power the environment has. So, the power environment has we should think it in an adversarial manner; so, what is the best mistakes that the environment can force on it, right.

So, I mean sometimes we sit in some sports, right tennis and all. If you are playing against a very tough player he can force a lot of error on you, right. Like I mean you can make self errors, but the opponent is very strong he can hit it in some spots which will make you error. So, it depends on what is the power of your enemy.

Now, here let us treat the environment to be in a similar fashion let us say environments aim is to just make sure that you make lot of errors; he want to win against you; how much errors he can force on you, ok. So, in that way we are always going to treat this interaction between environment and the learner as a game. One guy is you are trying to play trying to learn your opponent and up by making minimum number of mistakes and maybe like opponent let us say he is trying to enforce a more number of mistakes on you, ok.

Now, the question is you have ensured that through this bounds; however, tough is your enemy or however, tough your opponent I will not going to make more than this mistakes. You have ensured this. But now let us ask from the opponent's point of view or from your enemy point of view how much mistakes he can at least enforce on you that will be his question, right.

So, for that we are now going to understand what is the power of environment and what is the power of your opponent. And remember we are going to look at the power of your opponent or environment, still under the restriction that he has to follow the realizability assumption. We have I will set up a rule for this game which is realizability assumption. He has also has to show his power under this rule.

So, to understand this we will consider little bit of this tree structure. So, consider a binary tree of depth n, ok. So, if I have a binary tree of depth n, how many nodes it is going to have? 2^{n+1}-1 nodes(if depth starts from zero) , right. So, let us consider for simplicity. Let us take the case of simply 3 nodes; when sorry; depth to, I have a depth to 2 here and I have 7 nodes here, ok.

Let us call this to be v_1, v_2, v_3. Now, let us say I will come up with, I am going to look at for these points I mean these v_1, v_2, v_3 they are basically sample points which we denoted as x_t earlier, ok. These are the ones. So, v_1 is the first point, v_2 is the second point, v_3 is the third point like that.

So, let us say the adversary tries to follow, do the apply the following strategy. Here initially throw you v_1. Let us forget what the learner is doing, let us only think of from the adversary point of view or the environ point of view. He will start with v_1 this is the root node here and then let us say he is going to apply assign some label to this.

If he applies label like 0 label, he is going to move towards its right child and in the next round he is going to use this has his sample. If he gives label 1, he will move to this point and then he is going to use this as the next sample like this. He keep on doing that.

At this point after he, suppose in this point if he has assigned label 1 to this, let us say he moved here and at this point he assigns label 0 he will move towards his left side and if we which is, this is; when I see this let us say this is my, right and this is my left. So, when

is here when he applies 0 label he will going move towards, right and then he is going to take left he is going to assign label 1 and like that he keeps branching like this, ok.

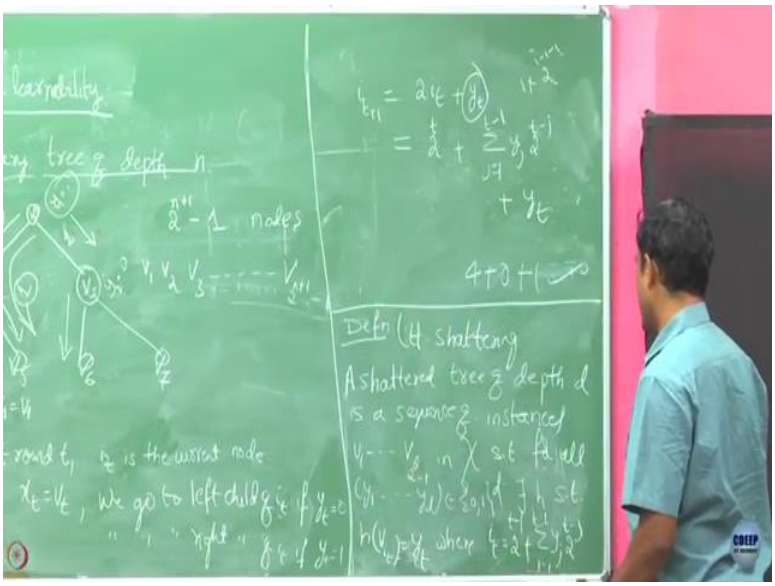
So, now if we have 2^{n+1} -1 nodes(these are the nodes that we can label if depth starts from zero), I am going to enumerate them as $\{v_1, v_2, v_3, \ldots, v_{2^{n+1}-1}\}$. So, these are the node choice he has and using these node choice he has constructed a binary tree like this, ok. So, now, as I said his strategies going to be, he is going to choose x_1 to be v_1 to begin with, ok. Now, right away let us at round t, let us do at round t. Let us say at round t is at i_t is his current node, ok. At something like let us say i_t is this current node.

Now, from this node if he is going to assign a label 0, he is going to go to the next child node which is towards his right and if he is going to assign value label 1 to the point at i_t, he is going to move towards his left, ok. So, let us say how to write. So, let us say if we choose x_t equals to v_t, I think I have to use this slightly different label to be consistent, ok. I will I will just flip the labels here.

We go to left child if this is going to and then v go to right child if $y_t = 1$. So, let us try to define just this strategy of when whatever this construct like this and then we will see that how the adversaries going to use this to inflect what kind of damage he can inflect on the learner, ok.

If this is the case at $i^{t\text{-th}}$ round he is going to either right or left depending on whether your label is going to be $y_t = 0$ or $y_t = 1$. And I have labeled my nodes in this fashion v_1, v_2, v_3, if I have further down I will label it as v_4, v_5, v_6, v_7 then v_8 v_9 like this, ok.

(Refer Slide Time: 23:27)



Then, if it is this case what is i_{t+1} is going to be? It is going to be $2_{it}+y_t$. So, I am going to one stage down, going to next stage. Can I write this in a iteration fashion the node numberings? So, let us say let us take the simple case, let us say I am in this node, this is my i_t. Let us say I have made a label 1 and I came here. What is this value is going to be?

Student: 6.

So, it is going to be 6, right. So, 3, it is 3 to the 6.

Student: Sir, (Refer Time: 24:19) label 0, 2 times 3 plus 0th (Refer Time: 24:22).

Right. So, when I said it is going to come this is going to be 0, right y_t is 0 in this case, it is going to be 3 into 2, 6, we are going to get. So, like that I can iteratively write this. So, this is i_{t+1} written expressed in terms of i_t. I can do the same thing, right I can replace i_t here by i_{t-1}. If I keep on doing this repetition what I am going to get, you can verify this. So, he can write i_{t+1} in terms of the labels. So, finally, what I have? i_{t+1} expressed in terms of the labels I have seen till time t - 1, right.

No, I am saying that let us say my environment is just going to use this strategy. He is going to start with this node. He will assign a label to that based on whether he assigned a 0 or 1, he will go to the next one and he can continue to do this. No, this is j running from 1 to t - 1.

. So, how many of you are already know the notion of VC dimension? You know VC dimension. Which course?

Anybody else. So, in machine learning especially in the supervise learning setting those of you know what is VC dimension. I mean those who do not know about VC dimension kind of VC dimension tells you what is the complexity of your hypothesis class how tough it is and based on that we determine what is how many samples we need to get. So, that we can guarantee certain number of error on your risk basically.

We are going to use a similar notion of something VC dimension, we will come to that. But as always like before we decide define the notion of VC dimension there is something called shattering, right. So, we will just introduce the analog motion of that then I am we will just discuss that in the next class. So, those who do not know that notion of VC dimension.

So, just read it before you come to the next class. In the supervise learning set up when you look into basically if you look into the learning theory of supervise setting, you will definitely come across this term shattering and VC dimension.

So, as I said we constructed basically this decision, binary decision tree using this instances, right. So, let us say you have the sequence of instances which forms a decision tree of depth d. We are going to say that this decision tree here whatever we have is going to be shattered if there exists a hypothesis class H such that you give any kind of sequence of labels.

So, you have this points, you give any arbitrary labels set of labels of d, upto d then v should be able to find a hypothesis class h such that all this points here will be getting a label y_t according to this strategy that we adopted. That is the i_t^{th} node will be selected in this fashion, and when we select that i_t^{th} node the corresponding node on that will get the label that you want.

So, basically we are saying that this decision tree like this is going to be shattered if I can come up with a hypothesis such that you for any given you give me this sequence of labels, labels series, right. So, this labels you are going to see. I will reproduce that label by taking this path. You understand this, ok.

So, we will bit discuss this more this notion of shattering. It will be easier for you to digest this concept if you look into what we what is VC dimension in supervise learning, ok. We will revisit in the next class. Please do read that book notion of VC dimension.