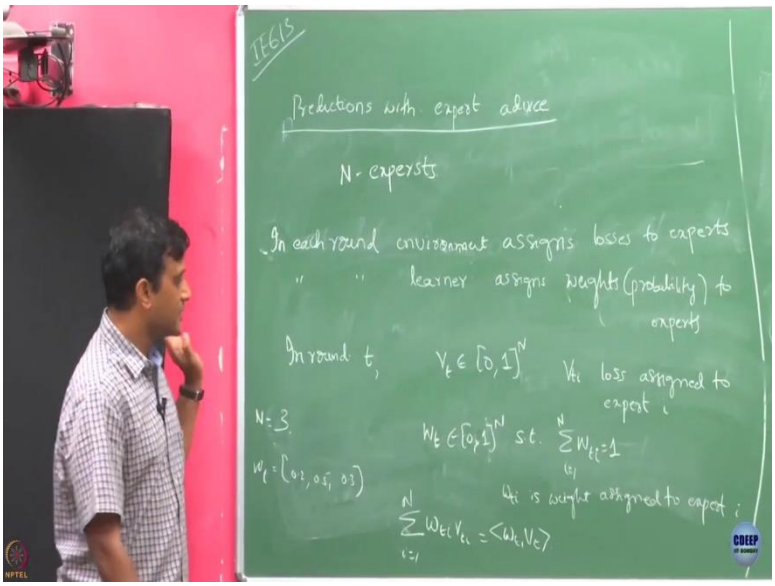
Bandit Algorithm (Online Machine Learning)
Prof. Manjesh Hanawal
Industrial Engineering and Operations Research
Indian Institute of Technology, Bombay

Lecture - 10
Weighting Majority

So, fine our now goal is to show there exist an algorithm such that my hypothesis class becomes learnable right. Now, what is that algorithm? Ok. So, before we understand that algorithm, we are going to make a slight detour and going to understand a set up called prediction with expert advice and study a algorithm called Weighted Majority and then show that; that weighted majority algorithm is the one; actually that makes my hypothesis class learnable ok. So, first try to understand what is that prediction with expert advice setting ok.

(Refer Slide Time: 01:01)



Suppose, let us say you are consulting some N number of experts and their experts in the problem you are interested, but even among the experts; the level of expertness can differ right. Now, what you would want to do is; you want to among these experts, you want to identify the best expert ok.

How? It works, let us say think of this experts as my hypothesis class, right. Let us say each hypothesis class is like an expert and as we are doing so far, we have our goal is to identify and hypothesis or an expert which minimizes my total number of losses. So in this

setting, we are going to treat my; we are going to consider that there are some; let us say N number of experts, in each round when an instance comes, I am going to ask each of this experts, tell me what is your prediction on this? Tell me what is your prediction on that?

So, each of them can give prediction, but I have to decide which experts prediction I will go with ok. So, I may go with one, I can see like out of this experts I can select one expert and maybe give as my output. But we know that if you are doing that we know that the adversary can make you incur loss in every round right; if (Refer Time: 03:02).

So, what we are going to do that; whatever the prediction we got from experts, we are going to assign probability to each one of them and going to declare one of the; so we are going to select one of these experts according to some probability and then going to declare that as my prediction. It is not that deterministically I will take one expert and declare that as a prediction, I am going to select these experts according to some distribution and then I am going to declare whatever the suggested by that expert as the prediction, but now as I said all these experts may not be of the same expert level right.

So, at some point you want to start; start selecting the expert; a good experts with higher probability. So, what you can basically do is then assign some weights to these experts and then kind of update your weights on these experts based on whether the predictions are correct or not as you are going to observe. So, let us try to make this a bit more formal, but now we are no more going to say only prediction language; we are going to say that there are N experts ok, let us say N experts and in each round; the environment is going to ok.

So, the interaction between you and the environment is happening in this fashion; in each round the environment is going to assign loss value to each of the experts. You are going to assign some weights or probability to each of these experts and then you are going to select this expert based on your probability distribution on them and then you are going to declare that as your label ok. Let us say in round t; let us say V_t is a vector of dimension N.

So, all of you understand this notation? 0, 1 to the power N means; this is Cartesian product N times or this interval 0, 1. So, in this V_{ti} here means loss assigned to expert i. So, in round t; the environment has assigned losses to all the vectors, sorry all the experts and now; the learner is going to choose his vector W_t which is basically a probability vector.

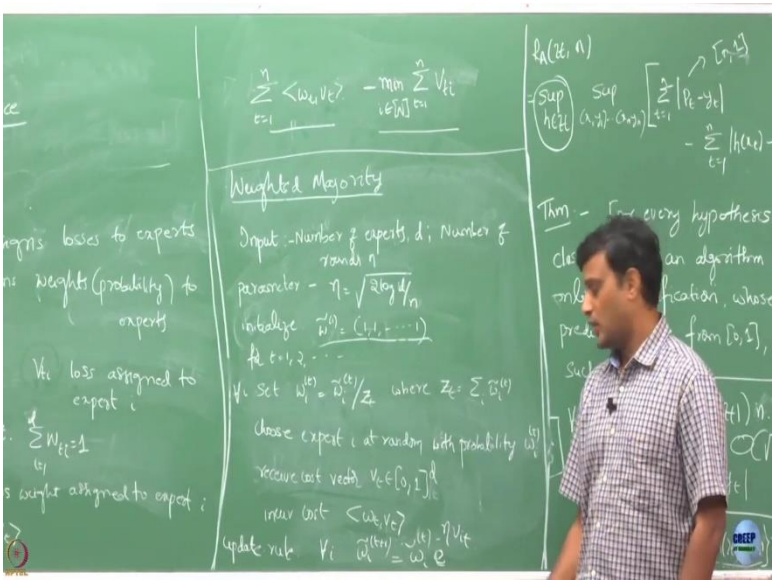
And then he is going to select one of these expert according to this probability distribution, you understand this point?

So, let us say for simplicity like; let us say N = 3 and then in this case W_t could be; let us say 0.2, 0.5 and 0.3. The learner assigned these states to the experts; then it means that he is going to select one of the expert according to this distribution; that means, he is going to select the first one with probability 3, maybe second one with probability 5 and the third one with probability 3.

If this is a case, if environment is going to assign this loss to the experts and you have been selecting one of them with this probability; what is the expected loss you are going to incur? So, if you are going to select i th expert yeah. So, if you are going to select i th expert; your loss will be V_{ti}, but you are going to select them in a random fashion according to this distribution W_t. What will be the expected loss? The expected loss will be then $\sum_{\{i=1\}}^{n} W_{ti}V_{ti}$ right; that I am going to incur in round t. Is this clear? And what is this? This is nothing, but the inner product of W_t and V_t.

So, right now I am just talking in the language of loss which is in the interval [0, 1] ok. So, if you are going to pick i th expert in that round, the nature of the adversary nature, possibly adversarial has assigned this much of loss to him and you are going to incur that. And since you are randomized picking this experts; I am interested in knowing the expected loss and this is that quantity ok. If that is the case, I want to minimize my expected loss incurred over N rounds ok.

(Refer Slide Time: 10:32)



So, what is the expected loss incurred over n rounds? That is going to be $\sum_{\{t=1\}}^{n} < W_t, V_t >$

Now, we are taking the expectation; we are not interested in one realization right, we are trying to; so we had; we have modified our regret and we are interested now in expected regret right; that is because of that we will be interested in the expected loss here ok.

So fine, you are right like; I will select only one expert in a round finally and I am going to incur the loss with respect to right, but because you are selecting that with some probability; it could have been other also right with some probability. So, what you want to guarantee how your algorithm performs on an average, not on a particular one realization ok; that is why we are interested in the expected loss here. Now, this I want to compare with; what I get over if I select a particular expert.

So, let us say if I am going to select a particular; let us say I; I happen to choose i th expert, what is the loss I would have incurred?

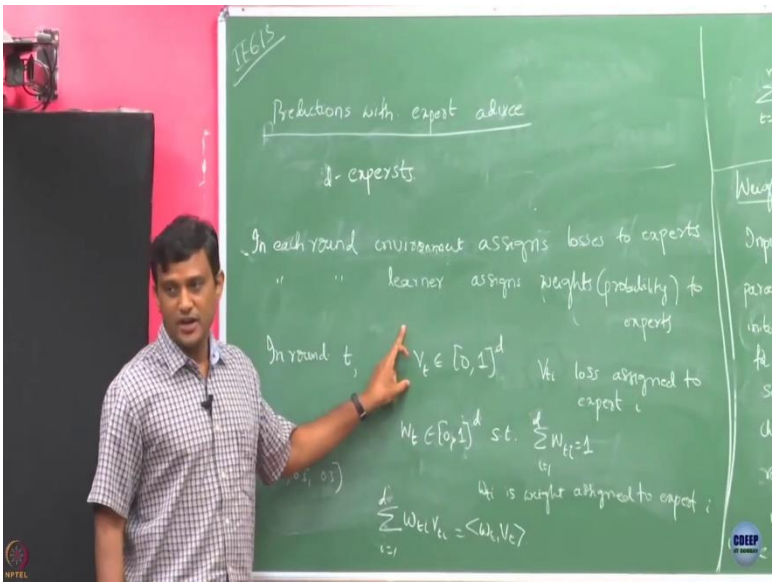
Over t rounds; so let us say my strategies deterministic. In each round, I only select i th user; sorry i th expert, what would be my total loss? $\sum V_{ti}$ right. So, this is what I am doing; basically I am comparing the performance of expected loss with n rounds to the performance I get if I have to place a single experts in all the rounds ok.

But what I may be interested in is the best I could get from the experts. So, this is V_t sequences; let us say fix, some V_t sequences is there that is the loss generated by the environment. By applying your algorithm or like by applying by selecting the weights and choosing a experts according to the weight; this is the loss I incurred now, but I am comparing this against the best loss I would ha`ve incurred; who is the best expert among all this loss sequence that I have observed; who would have given me the smallest loss?

And that is going to be my benchmark; I am going to compare my performance against this and the goal in this prediction with expert advice is to minimize this quantity is that clear? Ok. Now, let us see what is a good algorithm to minimize this. Ok. So, we are now going to discuss a classical algorithm called weighted majority. So, notice that when I say the environment is generating this loss vectors V_t; it could be arbitrary process, I did not say any put did any restriction on this right it; all I am saying is environment is going to assign some loss ok.

Let us now try to work out this. So, maybe I think we were going to use; so in this course, we will try to maintain the consistence level; whenever I use a capital letter; that is for random variable. If something is like a fixed deterministic quantity, I am going to always use a small letter for that ok.

(Refer Slide Time: 15:42)



So, because of that I am going to use a slightly different notation for this; I am going to use the d here because this number of experts is fixed a priori; so that is a deterministic quantity. Ok fine, so this is the; let us discuss this algorithm.

So for this algorithm, the input is the number of experts d and also this algorithm needs you to tell a priori how many rounds you want to run this ok. Like, we will see later how this can be relaxed where we do not need to tell a priori how many rounds we have to run, but we can stop at any time you want to stop and still get the same performance guarantee this, we are going to derive for this setup ok.

Let us assume for time being like we are; we have been told what is the number of rounds, it is going to run. And now we are going to based on this quantities d and n; I am going to set a parameter η which is insert in this fashion; $\sqrt{\left\{\frac{2\log d}{n}\right\}}$. Now, this algorithm maintains weights; what is the, in this setup I do not have any control over the way the environment or the adversary is choosing this loss vectors right, but what I have control over is; how I am going to choose this weight vectors, this is under learners control right.

So, depending on the way the learner chooses this W_t; we will have different different algorithm. This weighted majority is nothing, but a specific way of choosing this weight vector W_t; sorry this W_t in each round. So, let us discuss how this algorithm chooses this weights in each round ok. So, this algorithm begins by; so there are two quantities here, one is W_t ok; maybe I should write it like this and this is for all; I ok. So, there are two quantities here; one is W_i^t and then another is $\widetilde{W_i^t}$ ok

So, this $\widetilde{W_i^t}$, initially I will start with taking (1, 1, 1,…..,1); that means, I am giving equal weightage to all the experts ok. Now, when I start; I am going to convert this weights into probabilities in this fashion. So, I am going to take this Z_t to be the sum of all these W_i^t's and divide this $\widetilde{W_i^t}$ by Z_t. Now, can you see that this W_i^t over i's make a probability distribution; is that clear to you? So, if you add all this W_i^t; this quantity is actually 1 ok; so, this W_i^t makes a probability distribution.

Now, you choose an expert i at random with probability W_i^t right; in every round you have anyway come up assigned a probability vector on the experts you choose one of them with that probability. After you do that, you get to see the loss vector from the environment for

this and after this; anyway this is the cost you are going to incur. We have already discussed this is the expected loss we are going to incur right.

And now this is the main part of the algorithm, how I am going to update this weights? Ok. The way I am going to update this weights is W_i^t to be whatever the previous weight I have in my previous round; I am going to multiply it by $e^{-\eta V_{it}}$. η is what I have set here is the initial parameter. So, notice that if my loss is high, what is happening to the weights in the update? They are coming down? Right. If my loss is small, then it will not be coming down that much ok.

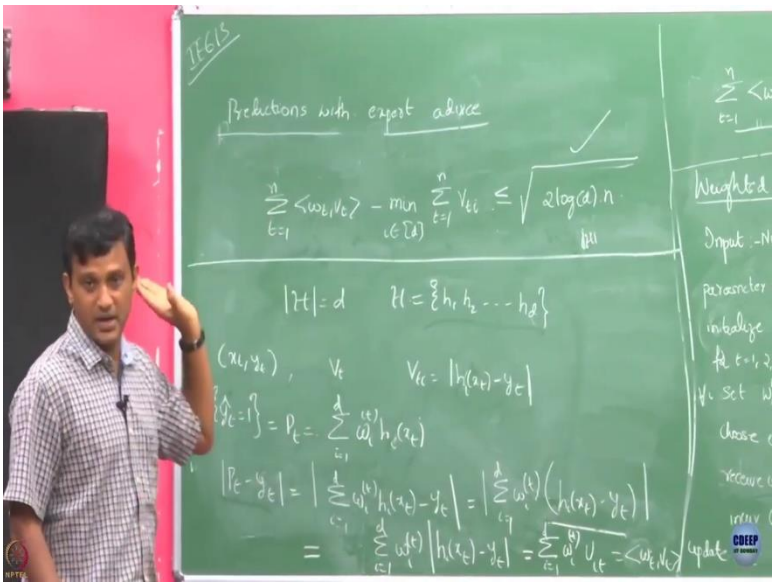
The ones for which losses are high; they will definitely come down significantly. So, in a way what this algorithm is doing is; it is kind of keeping track of the weights for all the experts and in every round; it is updating these weights, depending on whether that expert give me high loss or low loss; if the loss is high, I am going to decrease its weight significantly ok.

So notice that like when I am going to get this entire vector V_t; so I know what is the loss for each of the expert. So, that is why I know this enter V_{it} and I could do this. So, now again when you come back; now again these are weight, this need not be probability vector, but you again make it a probability vector by dividing it by the sum of all the components and then I again you are going to play an expert according to the probability distribution ok.

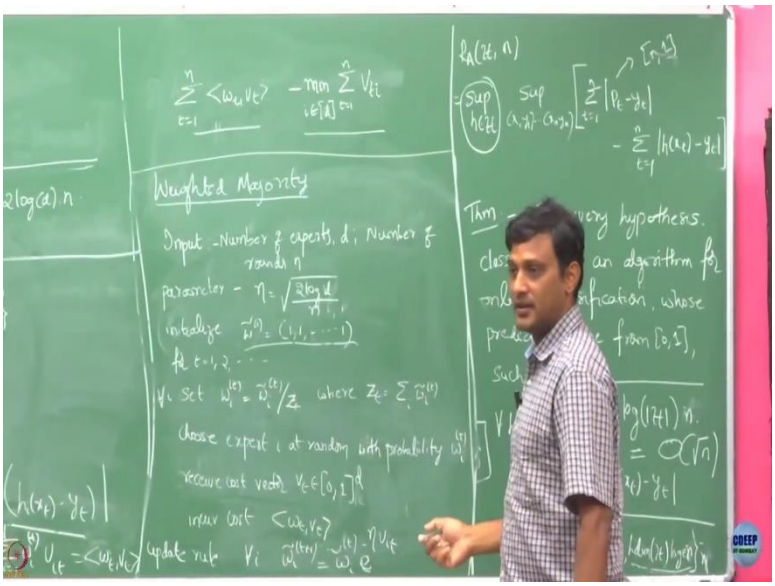
So, do you think this algorithm should give a good performance; if you are going to update its weights like this ok? May be, we will show that this is indeed does the job what we wanted to do. Let us write down; what is the performance of this algorithm and this is one of the kind of celebrated algorithm in all our adversarial setting; you will see that all the kind of later the bounds we are going to derive for various setting will be based on mostly this idea. The idea is simple like you increase the weights, if the loss is small; you decrease the weights, if the loss is high and accordingly you build a distribution and accordingly you select the experts.

In a way, that makes sense right like if I am observing high loss for some experiment; I want to give him less weight. If somebody is giving me a small loss, I want to play him more maybe like that is a guy who is always keep give me lesser loss ok.

(Refer Slide Time: 25:18)



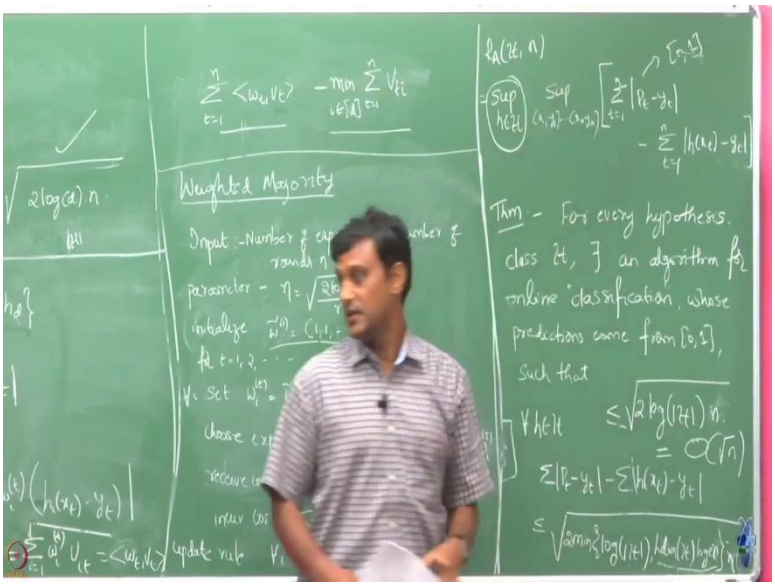
(Refer Slide Time: 25:37)



Now, what is the guarantee for this? We are going to show that; we are going to show that this algorithm has an upper bound of $\sqrt{\{2 \log d\, n\}}$. You noticed that this bound is for the case where this W_t is are chosen according to this algorithm. If you are choosing this W_t according to some other logic, this bound need not hold yeah.

Now, I want to go back and see; how I can use this result, for time being let us assume this result is true ok. Yes, we are going to show it in the next class, but if this is true; can I show this; this result? Ok.

(Refer Slide Time: 26:57)



Now, how I can map this result to this result. So, in this case I am going to focus on the case where my hypothesis class is finite to argue. So, let us say this hypothesis class H has d elements in it. Whatever the sign, let us call it d and let us say and enumerate them $H = \{h_1, h_2, \dots, h_d\}$. I am going to now treat each of these hypothesis class as an expert and so what is happening in each round? According to this expert setting; the environment is going to assign losses to each one of this; right that was V_t.

Now in this setup the environment is only choosing x_t, y_t and round t right. So, now, I have to map whatever the x_t, y_t that has been chosen in round t to V_t that has been chosen in my expert setting. How I am going to do is; I am going to define $V_{ti} = |h_i(x_t) - y_t|$. Can I do this? So, I told you write like this V_t is arbitrarily generated. So, if x_t and y_t is the sample that is generated in round t; then I can treat this as my loss in round t right for the i th expert ok.

So, I have this vector now V_t; now if these are the experts, what I am saying is when a point comes x_t; I can feed it to all of them and get to see what is the prediction right. And then I can take; I can take their linear combination as my actual output, you remember here P_t is something between [0, 1]; it is not necessarily 0 or 1, it can be anything between 0, 1.

So, in this case; I will take, I will define P_t to be simply; can I define it like this and is it still true that $\Pr\{\hat{y_t} = 1\} = P_t = \sum_{\{i=1\}}^{d} W_i^t h_i(x_t)$. So, $h_i(x_t)$ is the prediction given by i th hypotheses on point x_t. So, some of them may say 0 and some of them may say 1, right.

So, for 0; those who say 0, they are not contributing anything to the summation; only those who say 1, they are going to contribute to the summation. So, it is basically what I am doing is; I am adding the weights of all the hypotheses that is saying 1, right. That means, with that probability that is the sum of all this W_t's; where this guy is one that is going to tell me what is the probability that, I am going to predict it as label one in that round t.

So, that is why I can write this P_t as this; is that clear? Ok. Now, let us express my $|P_t - y_t|$. So, I need this quantity right basically I am trying to express this quantities in terms of my W_t's and V_t's ok. P_t, I have already this quantity and now this is y_t; is this clear? So, now we will do some series of manipulation simple one; just try to follow them, notice that this W_i^t is a probability vector right and this y_t is a constant here in that round t.

So, what I could do in this case is; so can I manipulate it like this and now I want to do some what further manipulation of this; yeah.

This V_{ti} is the mod of this quantity, but here it is we are not at in that shape; we have to convert it to that shape that will be our next task. So, how can I write this; how can I bring the mod inside? Ok. So, notice that let me first write this, our claim is this is true. So, we are saying that even instead of looking the mod outside, if you look mod inside; for like this, they are same ok.

Now, let us understand why this is true; is this true? Yeah, mod of the whole term need not equals to the mod of the sum right. But, if we can show all the quantities inside the summation; they are all are positive or all negative, then this is true right. Suppose, if all of them are positive; no problem, if all of are the negative?

Student: No problem.

No problem, only issue comes if some of them are positive and some of them are negative ok.

Now, let us take; so my y_t is what? Is a binary variable value right; y_t can be 0 or 1. Suppose, y_t is 0; no problem right, y_t is 0 means $h_i(x_t)$. either 0 or 1, this W_i^t are always positive; so every term inside is positive, when y_t equals to 0; so, then this result holds

Now, we are almost done right; now what is this quantity equals to? This is equals to; equals to this and this is equals to this. Now, what we are basically showed is; this quantity

here, the first term here is nothing, but we can map it to the first term here. And the second term here is the other term right because this V_{ti} nothing, but this $V_{ti} = |h_i(x_t) - y_t|$ and now because now we are saying this bound is independent of what is the sequence and what are the loss vector we have observed. Actually, I could as well tear this to be the infimum over h here, right.

Because this is the independent of what is the sequence. So, because of that can I claim that this algorithm here is actually, if I apply on this; it will give me this regret bound because I have already shown you right like if I apply this algorithm on this experts set advice setting; I know already I am getting this bump, but we have just argued that this expert predictions with experts setting is nothing, but this setting ok. So, because of this I mean whatever the setting for which we define this regret bound right and this expression here is nothing, but this expression by appropriately defining your P_t and your V_{ti}.

So, because of that we have this bond and what is d here? d is nothing, but cardinality of h and that is what we wanted to show and this is what is η?

Student: (Refer Time: 37:37).

Yeah, yes; yeah this is your internal parameter; for your algorithm, use whatever internal parameter you want to use. What I care is finally, based on your input parameters what is your outcome?

Student: (Refer Time: 38:03).

Now we are connecting this setting prediction with expert advice to whatever the classification of online classification problem we dealt with. And we are just said that this is nothing, but this setting my mapping is expert advice setting to my binary classification; online binary classification problem.

So, our claim is whatever we have this bound; we can achieve what we wanted to basically show there exist an algorithm, right. Now, the thing is this is that algorithm; this is that algorithm which will give you this form. This is online right; yeah, once you have this; this is online learnable because this is sub linear the regret we are getting ok.

So, in the next class; what we will do is, we will just basically go through the proof of this part. And we will not prove for the case where this is infinity, when the cardinality of H is

infinity that proof is bit more involved; we will just skip it, but we will complete this ok, let us stop here.