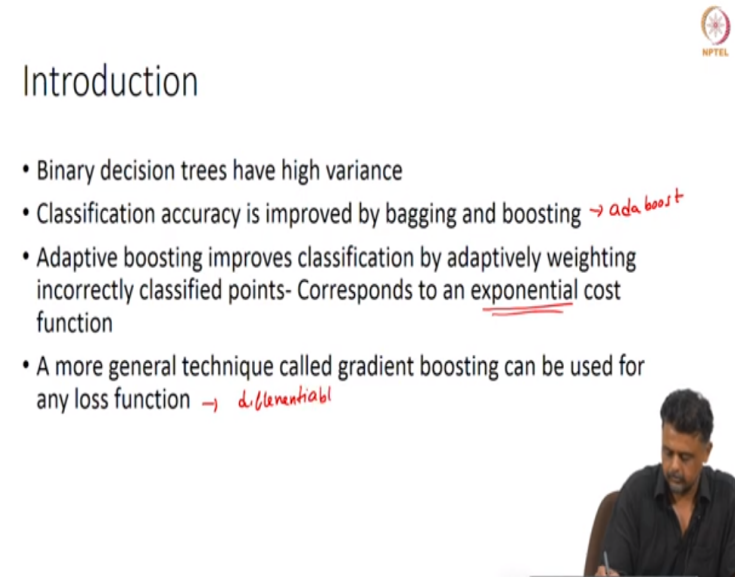
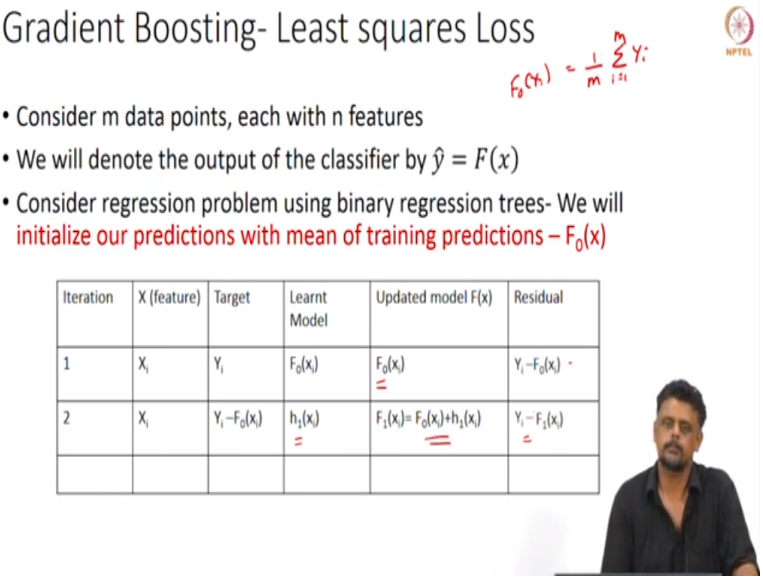
Machine Learning for Engineering and Science Application
Professor Ganapathy Krishnamurthi
Department of Engineering Design
Indian Institute of Technology Madras
Gradient Boosting

(Refer Slide Time: 00:14)



Hello and welcome back so in this video we will look at gradient boosting, now we seen that binary trees have high variance and we have improved classification accuracy by bagging and adap boosting adap boost is what we have look at the way adaptive boosting improves classification by adaptive reweighting incorrectly classified points in every iteration, we show that we will see we try to explain by actually corresponds to an exponential cost function look at which is gradient boosting where in exponential cost function, so in this lecture we will look at the more general technique called gradient boosting can be used any loss function in the sense any differentiable loss function.

(Refer Slide Time: 01:04)



So we will consider M data point training data point each with N features, M data point sand N feature we will denote the output of the (())(01:18) classifier but we are going to consider only a regression problem so we look at binary regression tree, let we will consider a binary regression tree the output in the regression tree is Y hat corresponding to the input of X, I have not integrate a subscript here for every data point.

So we have look at this table here we are looking at iteration wherein we are in going to the successfully update our model, machine learning model F by learning from residual, from in this case consider in put feature XI so I runs from one through M corresponding target variable the ground truth is Y subscript I and we will learn we will fir this using binary regression tree and we will called that we will denote that by F1 of XI so every XI the output is F1 of XI, F1 is our model address point.

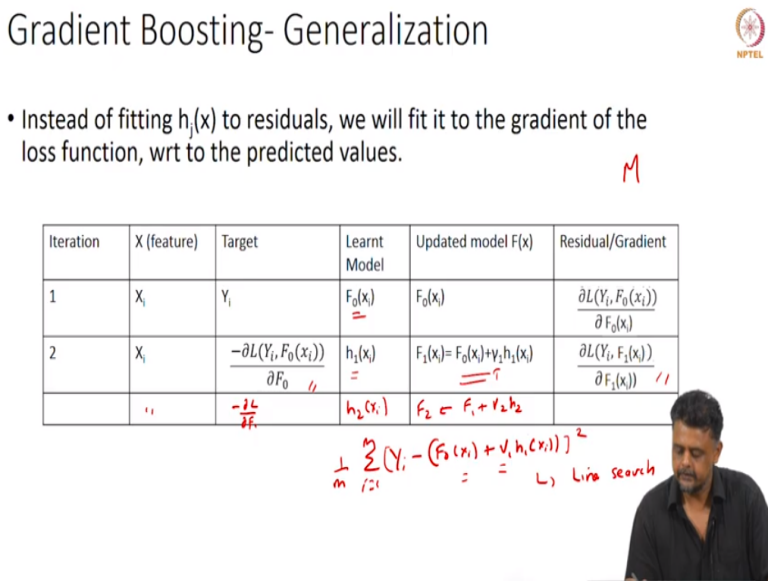
So this is our final model this point F1 of IX then we will calculate the residual which is basically the different between the ground truth and the output of the regression tree, YI minus Fi of XI and we take the second iteration we once again consider the input features for retaining data but then we will fit it not to YI but rather to the residual and the model that we use to fit XI to the residual we will called H1 of XI so then at this iteration the updated model F2 will be the previous model F1 plus the model that we use to fit residual okay, and we call we denote that final model as F2 and then we once again calculate the residual so we can see how this goes the

third iteration we will once again consider the input feature are data point XI we will fit it to F2 we will call this model H2 then we will update out model plus F2.

So this is one form of gradient boosting and this actually corresponds to we will see that this actually corresponds to using a loss function a least squares loss function so I am just going to be without any subscribe are anything I am just going to be write it as F of X square, so the form of the loss function we will see that we will see in the fused lights that the case, so just to bring in line with how it usually (())(03:55) in literature we will instead of in the first at there we will saw that the first iteration we are trained the idea you should trained your a data using a binary regression tree rather in this case we will initialize the model with by considering the means of productions.

So F zero of XI for all X for all XI nothing but one over the number of data points, so it is just a means of responses, so every responses initialize to the mean of the responses in the training data it is just a guess in this case and then we proceed as before and then we call that model F zero of XI, zero iteration that is what we start of we start of with guess and we say that, we will be calculate the residual base on the guess and in the first from the first iteration onward we will fit it to a regression tree and the update our model once again calculate our residual so on and so forth, so this is how we will process, how do we generalize this okay, this is like a mention earlier again I have done the rate of explanation but it mention between case where your last function actually least squares lost function.

(Refer Slide Time: 05:18)



So then how do we generalize to any lost function we will look at it now so we have it features target we learns initial model is mean of all the responses and then what we do is we actually calculate the gradient of the lost function with respect to your guess from the previous iteration that is gradient of the lost function with respect to the prediction is what we calculate and that is our target, the second iteration as usual we updated remember as we see here we have updated using this again the learn model H1 which we have fit to the gradient to the lost function with respect to the prediction from the previous iteration and of course we have this height to hyper parameter to new one this again we have to do one more optimization.

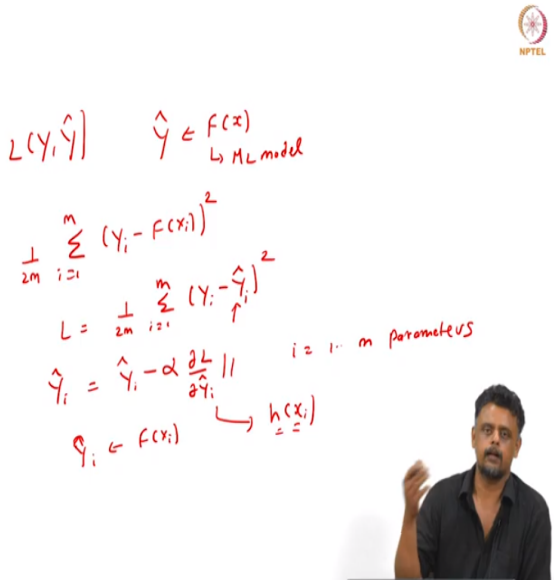
So in this case I will just write it down the one more optimization I am talking about this you have to solve this problem, say we can still use squares then use to a (())(06:34) this is done using another one called them line search there just simple (())(06:39) which have do not use write now so think is another optimization you have problem you have to solve right now in order to estimate this new one recent we have do that recall is that we have actually fit X side the gradient of the lost function is respect to the prediction from the previous iteration, so you can think of this as a correction that we do to improve the prediction impact.

So than once you have done this then we calculate again the gradient of the lost function with respect to the estimates that we have done and move on to the next iteration, so here (()(07:11) XI then it will be delta L with respect to delta F1 right I am leaving out all the variable but you

can fit them and then we fit them H2 then we will update F2 as F1 plus V2H2 and we once again estimate V2 in a similar process we will keep doing this till MI equation capital MI equation.

Let us say M is determine by cross validations, it is simply put you have a felt out data set which you will test and after and find out the accuracy at which you will you get the best find out the iteration at which you get the best accuracy that is one we have doing it, so in general cross validation is the best method to estimate the capital M, so this is the so why it is this work so where we doing this and so just to give you some idea why it is works you all seen gradient descent, the algorithm gradient descent.

(Refer Slide Time: 08:13)



So let us consider our any lost function this is a ground truth this is a prediction remember we have also said the prediction we can write it as F of some, X is our training data F is the ML model this case it can be a binary regression tree a least square lost function over I equal to one to M will have YI minus F of XI, squared one upon two M, let us if we treat and just for sake of making it easier to write and also easier to comprehend I am just going to write YI minus YI hat square this time to avoid too many subscription (())(09:09).

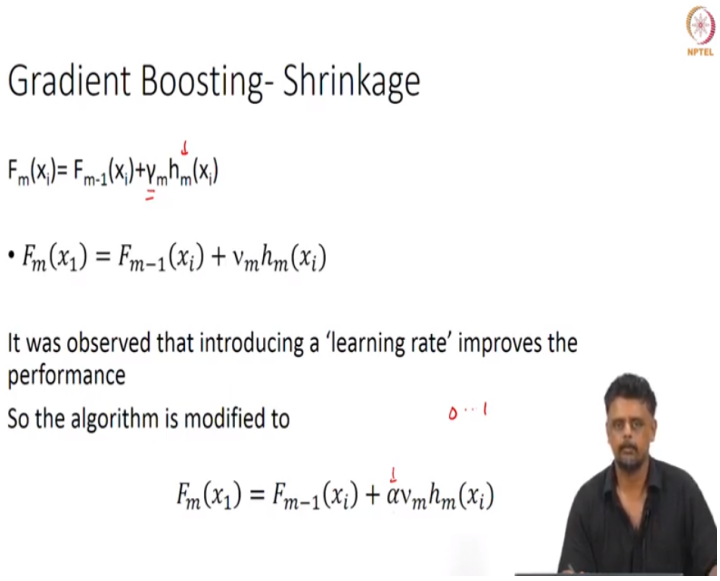
So let us see if you cancel this lost function we can said this as a function of this predictions explicitly, so L depends on YI hat your prediction and let us say our point and what we do when we are trying to do this regression problem is that we wants bring YI hat as closed to YI as possible, so you can think of YI hat S unknown parameters to be estimated, so given this lost

function YI hat is parameter that we want to estimate, so if you use gradient descent to estimate this YI hat given this lost function how do we do that we would say YI hat we update to is YI hat minus some learning rate so do not confused this learning rate with the new one we saw earlier I am just going to use just to keep this different just trying to established some methodology it times alpha.

So there are I equal to one to M parameters and if treat the YI hat remember YI hat but nothing but the what you got F of XI, and let us say if you doing a iterative process and after every iteration YI hat have some updated and how do we update it by estimating delta L by delta YI hat and how do we estimate delta L by delta Y hat, we estimate delta L by delta y I hat by fitting it to a machine learning model H of X sign this is what we do, so why do we you can say we can just now sincerely differentiable with just calculate that the problem with doing and then directly just keep updating does not make sense the problem with that is remember that we have a finite that training data's we do not know we do not have all X side limited we do not have entire space of X side.

So sense that training data has limited we only have small number of points if we say each X has 20 feature and we only have thousand or s ten thousand data points so think about it, so we would not be spanning the entire space so in fact it is get think of it three dimension also it will be very easy for to see that they do not be enough there are not a enough point so we are finite number of points and that trying find of let us say regularized this gradient by fitting it to the machine learning model H of X side, so once we estimate the gradient we update the parameter right that is exactly what we did in set here by a for estimating the gradient we are actually fitting it to machine learning model H of X side, so this is where this you can think of it this way this is where gradient boosting comes.

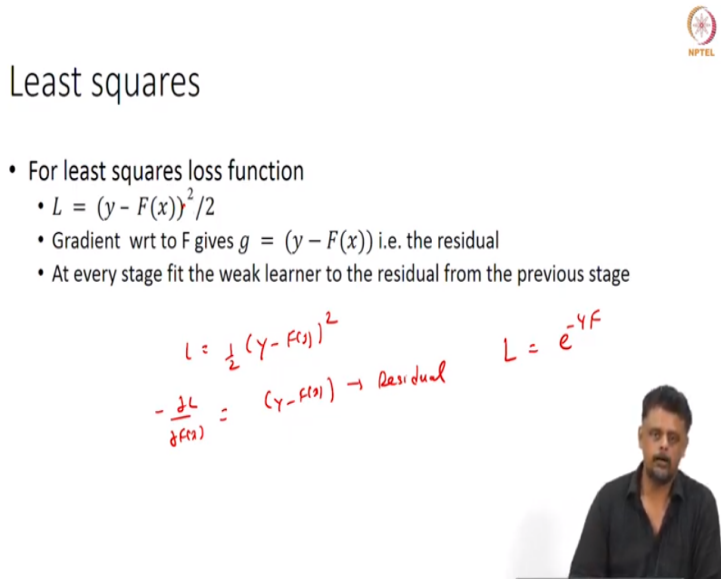
(Refer Slide Time: 12:04)



So to the summarized we start of initial guess and at every iteration we would update your initial guess by fitting H M to the gradient to the lost function with respect to the estimates your prediction from the your previous iteration and then also figuring this hyper parameter to get better estimate because the number you are fitting to the gradient, so then once if you fit the gradient to take that a machine learning model have a multiplicative factor here and try to estimate that factor.

So you can think of it also as doing a linear combination of multiple machine learning model in most of the case it just linear combination of multiple regression tree, now it is also observe that see because if you do that in very quickly in may be in few iteration you were tend to start to over fit this can happens so in order to improve conversion and also not to over fit a learning rate is typically is to do alpha this is between zero and one, I think 0.1 to 0.3 also some something of that sort you also have to set in order to get better conversion so this is called shrinkage learning rate that you typically use, so you will estimate U but you will fixed figure alpha like a hyper parameter, so this is the general process for gradient boosting.

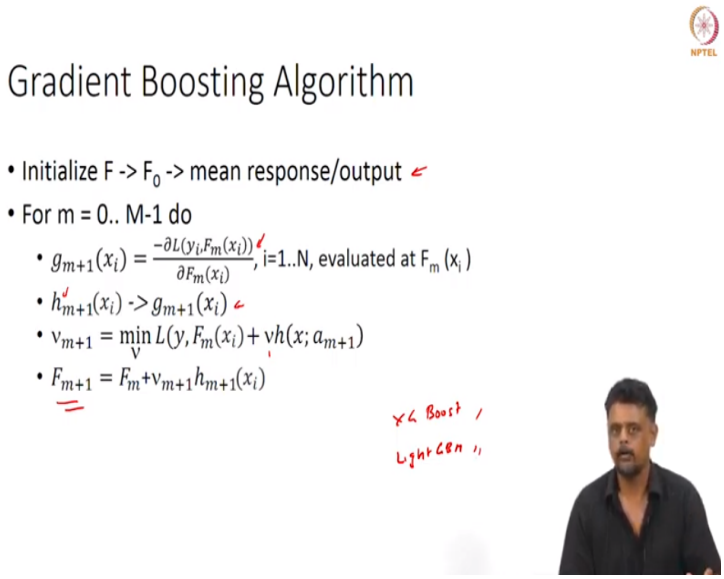
(Refer Slide Time: 13:29)



So let us if we consider let us say non least square model so this case an absolute deviation so Y minus F of X, I am going to leave out this submission and all that just make it little bit clear and this is your last function and if you want to take the derivative with respect to F of X give that as an exercise you will get sine of that okay that so that is it, so then iteration is start fixing it is not fitting your machine learning model just sign in, so your feature would be the same input feature and that of point target would be either plus or minus, so plus one or minus one would be a target that is how you are fit to a machine learning model and then of course you update your model and you can also use a learning rate to by improve conversion same then.

So if you consider your Y minus F of X square this is your last function you can calculate delta L by delta F of F of X is nothing but so we get negative is nothing but of that Y minus F of X if you have fact of two here that two will go away, this nothing but your residual remember that, so we also saw that you know add a boost the number add a boost, so add a boost correspond to E raise to minus four F, this is a last function for add a boost, so you can actually derived the update equation so add a boost starting from this last function so that is one way of doing it.

(Refer Slide Time: 15:14)



So in see that for a variety of lost (())(15:17) also something called Huber lost which you have not covered you can look that up once you can use Huber lost also as a last function you can do it, so remember that I told you we will go through this slide shortly that we are actually doing this using binary descent trees so but remember that descent tree as it so own, one binary descent tree but binary regression tree so if you use binary descent tree we again we will see that the we use this same procedure for there also so do not get just saying okay this is work certain regression but it will work for pretty much even for problems that you are doing classification I will just mention that briefly when we go toward the end.

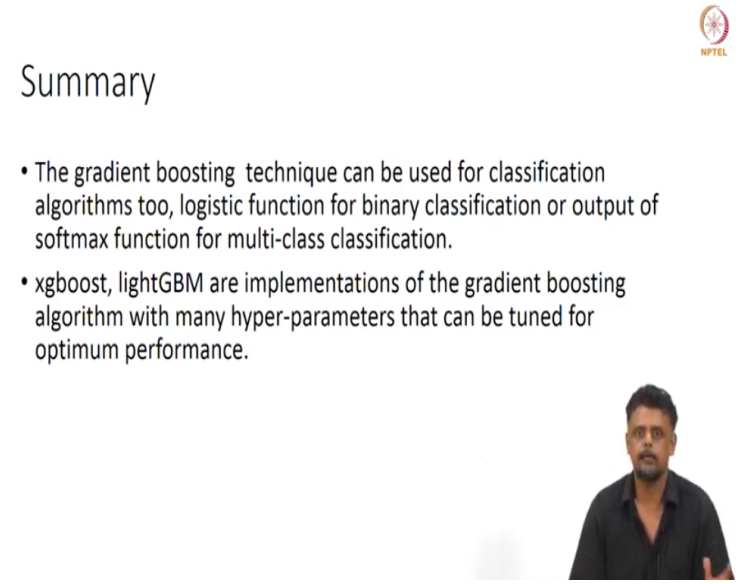
So let us look at the end of algorithm and summary so we initialize to F not which is not mean response, mean response every tend data you take that and for every X side that is output that is initialization and for substitute iteration you calculate the gradient of the lost function with respect to the value from the previous iteration you have predicted values from previous iteration fit a machine learning model which is a binary regression tree to that gradient estimate new, by solving a line search problem update your model keep doing that so fixed at number of iteration as indicate at class validation study and that is your gradient boosting algorithm.

So algorithm like SG boost I think I am also now a something called lite GBM, GBM are nothing but a gradient boosting machine once just one terminology there are just basically implementation of this technique, now they of course very efficient implement exchange G boost

like G very efficient implementation because they actually fit a binary regression tree effectively and enough there are lot of parameter that you can put in to find tuned that you see more flexibility depending on your problem but this is a fundamental implementation you calculate gradient of you lost function so you treat you are estimated parameter so you are estimated output.

So I know parameter estimated output or predicted output as parameters in you lost function so you doing gradient descent to estimate those parameter or your predicted output so that is the idea behind all gradient boosting algorithm, so this is a fundamental idea all gradient boosting algorithm.

(Refer Slide Time: 17:42)



So in summary so can we use classification algorithm also we have seen it now for regression but can also be used for classification, so when we are doing classification you used a logistic function there is away to do that a paper I am mention the beginning of the lecture actually describe how to do that if you use the logistic function now from classification problem our output of soft-max function for multi-class classification.

So remember that when are doing this way then the you will be actually instead of the output is just the probability where you rather than the class itself you will be working with the probability as a real number then you try to move the a probability closer to the closer to one in this let us say what do you like to do as I mentioned earlier XG boost, light GBM or the implementation of

boosting in algorithm with many hyper-parameter that can be tuned in for a particular problem and they tuned for optimum performance thank you.