

NPTEL

NPTEL ONLINE CERTIFICATION COURSE

Introduction to Machine Learning

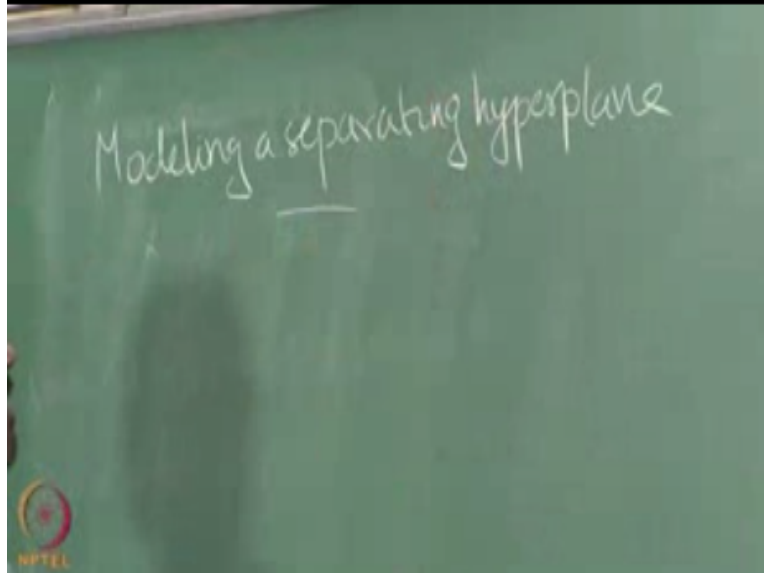
Lecture 26

**Prof. Balaraman Ravindran
Computer Science and Engineering
Indian Institute of Technology Madras**

**Separating Hyperplane Approaches
Perceptron Learning**

So I said there are two ways in which we can build linear classifiers so one was the discriminant function approach right, where by comparing discriminant functions we decide what is the separating hyperplane right and what is other approach modeling the separating hyperplane directly.

(Refer Slide Time: 00:38)



So there are many ways in which this can be done and we look at two in particular one because it leads us into neural networks later the other because it is the most popular way of building classifiers nowadays okay.

(Refer Slide Time: 01:13)



Before we go on let us just have a little further battles again okay, so that is a separating hyperplane right and right so this constitutes of all points x such that this equation is satisfied okay so this is the definition of a separating hyperplane so I am going to call this perfect sequence that defines our hyperplane v equate f of x to 0 that defines our hyperlink and if f of x is whatever if it is greater than 0 it implies in this case G of x is 0.

(Refer Slide Time: 03:00)

$$\{ \alpha \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 = 0 \} = L$$

$$(\hat{\beta})^T (x_1 - x_2) = 0$$

$$\Rightarrow \hat{\beta}^* = \frac{\hat{\beta}}{\|\hat{\beta}\|}$$

The chalkboard also features an NPTEL logo in the bottom left corner.

So some property is here if x_1 and x_2 both belong to the line okay and this is going to call this here if x_1 and x_2 both belong to L then we know that $\hat{\beta}^T (x_1 - x_2)$ will be 0 right, so what does it mean the $\hat{\beta}$ is actually a normal to L right so $\hat{\beta}$ is normal to L right so I am going to assume a specific actually okay $\hat{\beta}^*$ is normal to L right so $\hat{\beta}^*$ is perpendicular to L okay so $\hat{\beta}^*$ is a normal is unit direction I mean unit distance dimension.

(Refer Slide Time: 04:00)

$$(1) \beta^T(x_1 - x_2) = 0$$

$$\Rightarrow \beta^* = \frac{\beta}{\|\beta\|}$$

$$(2) \beta^T x_0 = -\beta_0 \quad \forall x_0 \in L$$

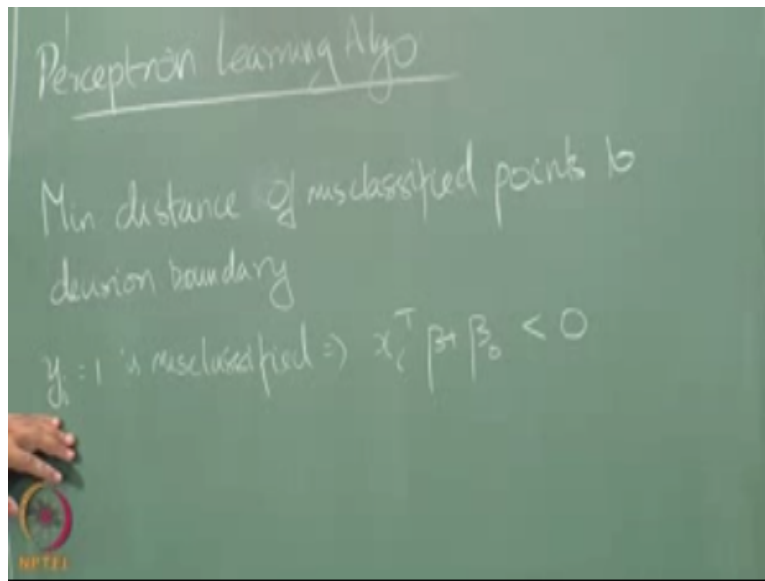
Hey this also we know so $\beta^T x$ not equal to $-\beta$ not right so I mean that essentially gives you this is $1 - \beta$ or not as $+\beta$ not so it will be 0 okay, this is β remember the β_0 is left out here okay so $\beta^T x_1$ will be $-\beta_0$ $\beta^T x_2$ will be $-\beta_0$ again so this expression will become 0 that is why x_1, x_2 belong to L if people have not, so what is that this $\beta^{*T} x - x_0$ where x_0 belongs to L x does not belong to L what is that β^* not is a direction that is perpendicular to L , so this is essentially the distance of x from L right there actually the signed distance of x from L depending on which side of L it is this is going to be different.

But because I am multiplying with β^* it gives me the projection on the perpendicular so this gives me the distance to some x , right and this will be the direction of β^* right so this is essentially projecting that on to this direction so this will give me the okay it is a β^* we know is β by norm β right so I replace that so I get $\beta^T x + \beta_0$ because x_0 times $\beta - \beta_0$ so I get $+\beta_0$ so and this expression is actually f of x right so this is f of x divided by norm β and β is what if $-$ of x right if it take the derivative of f of x with respect to x excuse me β right.

So this is three so it is norm of F dash of x right so this is f of x divided by the norm of F dash of X so what did we say this expression was the same distance to the hyperplane so f of x gives me a quantity that is proportional to the sign distance to the hyperplane right and if I find a hyperplane says that I normalize my meter to be one f of x gives me the sign distance to the hyperplane it is not proportional it actually gives me the sign distance to the hyperplane okay, that makes sense yes I just want you to keep this in mind.

Because whenever I say I am going to minimize f of x it essentially means I am going to try and minimize the distance of the data point to the hyperplane all right I am going to say were maximizing f of x I am going to maximize the distance of the data point to the hyperplane right, so just to keep that clear I am just introducing these notations beginning.

(Refer Slide Time: 08:50)

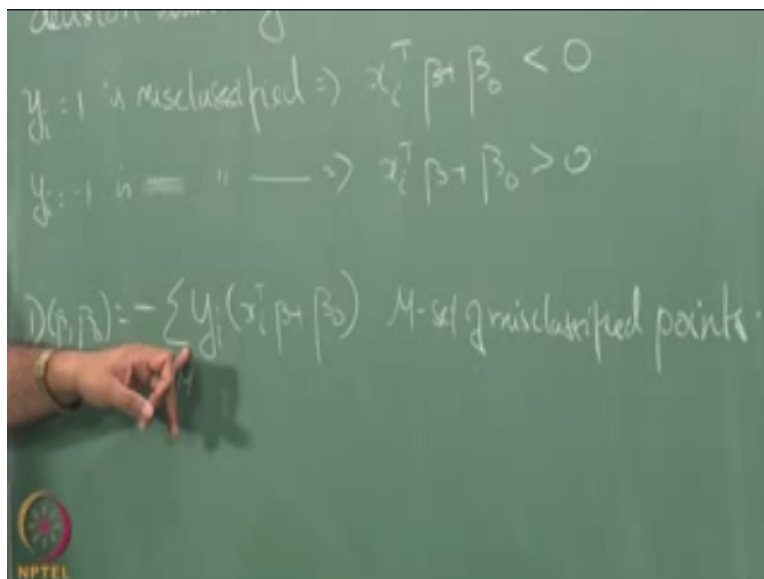


So the first thing we look at it is a perceptron learning algorithm so it is got a hurry tradition right people are familiar with have heard the term here networks artificial neural networks yes no people have heard the term artificial neural networks, so you know that we are going through the third boom of artificial neural networks okay, so in the back in the 50s and 60s there was this initial boom of artificial neural networks right everybody was so taken up with artificial neural network they said oh! here is something that can solve the a human learning problem.

And he talked about this already right, so that was started by the perceptron learning algorithm right and what about the second boom people know what the second boom was okay now we will come to that later, so the second boom was started by the back propagation algorithm which took care of some of the problems with perceptrons and then that also died away and the third boom has been shattered by what people call deep learning now right and the hype is scary but apparently not as scary as it was I mean I recently read some newspaper clippings from the 60s it is really scary okay.

So the idea behind the perceptron learning algorithm is very simple okay so I have this decision boundary right I want to make sure that any point that I actually misclassified at any point that I miss classify is as close as possible to the decision boundary right, in fact if I put it on the other side I am happy right I am doing the signed distances right, so if I put it on the right side I am happy put it on the wrong side and I try to keep it as close to the hyperplane as possible so that early there is some kind of a satisfaction that we are not getting something very egregiously wrong right, so I am going to minimize right, so we will assume that for data points which is such that $x^T \beta + \beta_0$ is greater than 0 and output us class plus 1 right, whenever it is less than 0 I will output the class as - 1.

(Refer Slide Time: 13:30)



So if the true class level is $p + 1$ but if the f of x is $<$ than 0 I mean if effect is less than 0 that means I will be outputting minus 1 right but the true class is $+ 1$ so that means this has been misclassified right. So likewise if right now that make sense if you understand this point now right so if the true class was $+1$ but f of x is < 0 then x will get misclassified right the true class was $- 1$ and f of $x > 0$ then x will get misclassified, so these are the two conditions under which you will have a point being misclassified.

So if I take this quantity right so what it little me for misclassified points it will be negative for misclassified points, it will be positive for correctly classified points right take that and I minimize it for all so take this for all the misclassified points right and I try to minimize this, so it is sometimes called the perceptron criterion right of the perceptron objective function okay right does it make sense what we are trying to do here louder, okay I will try to write begin but the idea was that that gives you zoomed in thing if I am very small then I will write bigger yeah okay so the point here is that I will not rewrite this from subsequently I write bigger okay point here is that if you take the misclassified data points okay.

So this is going to be a negative quantity right and I want this to be as close to 0 as possible right in fact I want m to be empty right I want m to be empty, so that is that is the goal this is the way to minimize this is to make sure that m is empty as long as um is non-empty hey I still not achieved the optimum okay when can m become empty so if my data is actually linearly separable right if you remember I drew some data points sampling from Gaussians there right I drew a minus somewhere here and then plus s there and so on so for that kind of a data will never be linearly separable.

I can never draw a straight line separating those data points right so here things are nicely linearly separable right all the X 's are one side all the zeros are on the other side right here the data was nicely separable so the perceptron objective function works well if the data is linearly separable if it is not linearly separable.

(Refer Slide Time: 17:28)

$$D(\beta) = -\sum_{i \in M} y_i (x_i^T \beta + \beta_0) \quad \text{M-set of misclassified points}$$

$$\frac{\partial D}{\partial \beta} = -\sum_{i \in M} y_i x_i, \quad \frac{\partial D}{\partial \beta_0} = -\sum_{i \in M} y_i$$

We will run into problems trick so now what we do is just use gradient descent so I will find by so what will that be okay four sizes are fine right so this is the derivatives and then so what I do now is essentially the technique that is very popular nowadays but it was not really called by that name in the olden days it is called stochastic gradient descent.

So people know what gradient descent is yes what is gradient descent yeah usually find the direction of steep descent and go in the opposite direction how far do you go in the opposite direction proportional to the what proportional to the gradient, gradient is very large in what you do right so it depends right so if you know the gradient properly I can compute the gradient set it equal to zero solve for it right and then just jump there right I do not have to go in small steps if I actually know where the gradient becomes zero I can just set my solution to that point right.

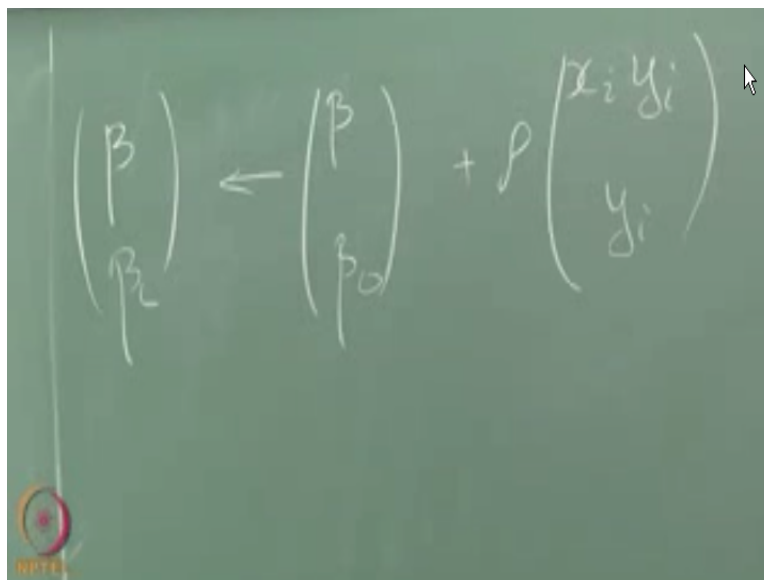
So the problem comes when things are little iffy why as soon as I move my beta in some direction what will happen in the perceptron case may move my β right say if I move in the direction of the gradient what will happen set M changes right so immediately my definition of gradient changes right so they move in a change β slightly I have to re compute the gradient right so the gradient I am computing is valid for wherever I am sitting in the β space when I move from there then I have to re compute the gradient.

So I cannot just move in the direction all the way right I can't really move a long distance pointed to by that particular point in space a particular gradient so I have to leak the iteratively re

compute the gradient so in such cases what we do is usually take a step in the direction indicated with a gradient and hope that you are moving in the generally expected direction right so if you take the expectation of the gradient you are moving in the right direction.

So the stochastic gradient descent has a lot of neuron cysts and other things to it so we will not get into that whenever we used to catch the gradient I will point out to you okay what are the things to be concerned about but will not get into the details of that right possibly anyone doing the optimization course and look yeah you might I am not sure whether they are going to get to stochastic gradient descent but they might covered it in the that course.

(Refer Slide Time: 20:36)


$$\begin{pmatrix} \beta \\ \beta_2 \end{pmatrix} \leftarrow \begin{pmatrix} \beta \\ \beta_0 \end{pmatrix} + \rho \begin{pmatrix} x_i y_i \\ y_i \end{pmatrix}$$

But not, not here right so what we are going to do is essentially take β so the gradient is $x_i y_i$ right but since I am anywhere doing stochastic gradient right as soon as I find one misclassified data point I am going to find the estimate of the gradient and move in the direction I am not going to wait till I find all the misclassified data points under that particular β right for the current β I find one misclassified data point and then I will change my β in the direction of the gradient right.

So what does it look like so I just take the misclassified data point multiplied by the decided output and add it to the weight vector right I have written this in a really funny spacing rate that is because I can usually what should go here now I can convert this into matrix notation right so

every time I encounter a misclassified data point I change my meter right so we are doing it in this particular form because this is the perceptron learning algorithm right.

So if you want to find all the misclassified data points and then compute the cumulative gradient and then change your β within that direction you are welcome to okay that is a perfectly valid way of doing it okay the reason we are doing it once to one data point at a time is saying this is exactly how the perceptron learning algorithm was derived yeah just wait okay so the row has to be really small if you are operating in a very, very stochastic environment okay.

Because so in effect what you are doing when you are doing stochastic gradient descent is that you are making many, many different estimates of the gradient in a local region right and you are trying to move in the expected direction of the average direction of the gradient so if the row is very large okay it turns out that you just make one estimate of the gradient and then move out of the region right.

So you will take a large step so we will go somewhere else the gradient will be completely different right or this could have been the wrong estimate for the gradient direction so to make sure that you are following a reasonable expected gradient row has to be really small okay having said that in the perceptron learning algorithm sitting low to one actually works okay and that is how the original perceptron algorithm was also stated right row was one.

And for normally for stochastic gradient descent one is very, very, very large okay you typically think of the order of 10^{-3} 10^{-4} and things like that for step sizes okay but it turns out in this case it is fine right row equal to 1 is fine and it will work it will converge you can show convergence with ρ equal to 1 but if in typically in stochastic gradient descent you want ρ to be small because the idea is what I told you just like we talked about taking a replacing expectation with an average over a region and into nearest neighbors.

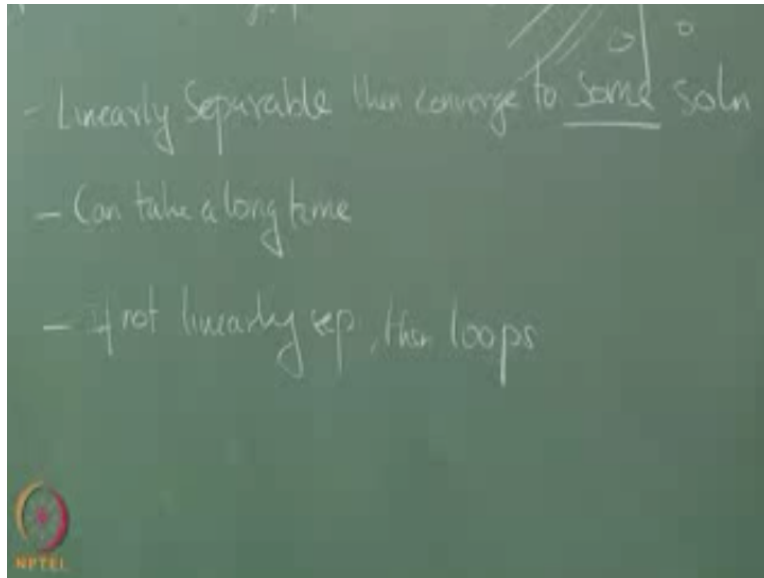
So something like that okay at a very, very gross level so you want to be able to take an average of the gradient in the local region but if you move very fast then you might be in a completely wrong direction okay so therefore you go slowly okay so this is the perceptron learning algorithm yeah β where is β_L sorry about that and that is not βJ either right.

(Refer Slide Time: 25:07)

A chalkboard with handwritten mathematical equations. The main equation is
$$\begin{pmatrix} \beta \\ \beta_0 \end{pmatrix} \leftarrow \begin{pmatrix} \beta \\ \beta_0 \end{pmatrix} + \rho \begin{pmatrix} x_i y_i \\ y_i \end{pmatrix}$$
 Below the equation, it says "- Linearly Separable." There is a small logo in the bottom left corner of the chalkboard image.

So if the data is linearly separable so what you mean by linearly separable that that exists a linear separating hyper plane that will make no mistakes okay so if the data is linearly separable then the perceptron algorithm perceptron learning algorithm will converge to some solution so what do I mean by some solution so here we saw okay let me do it again right so let us say these are the data points that are given to you.

(Refer Slide Time: 25:52)



There are many, many solutions that work right all right I mean assuming all of those are straight lines so you have many, many different straight lines that you can draw that separates these two classes right infinite number of them actually correct so the perceptron learning algorithm will converge to one of these three or one of these many infinitely many number of separating hyper planes which one would it converge to depends on the starting point when it is hard to say a priori right given a starting point can you tell me what is the Apple trend is going to converge to you can just have to run the what is the problem yeah, yeah sorry yeah.

You are going to say something we will come to that okay that is a way of defining one of these infinite number of hyper planes as the most desired hyper plane okay so if you think there is a way to pick one of these yes we will come to that the second problem is it can take a long time and take a long time especially if the gap between the two classes is very small and if the gap between the two classes is very small then it can take a long time okay so partly it is a function of setting row 21 but then it is, it is, it is hard I mean so see setting row to 1 essentially makes your thing oscillate a little bit right so.

So you will have something that makes a mistake here then you will go back you will make a mistake here and then you have to keep going back and forth multiple times before you converge to the right answer but then setting row to something small okay will also make you move only small steps at a time so that also may take a long time so there is always a trade-off between how large you make row and how small you make row.

And so one way of fixing this is what Row is one thing but can increase the gap between the data points of transformations use basis expansions that we talked about earlier so instead of doing it in the original next space say do it in the x^2 space or three x space or whatever I mean you can think of some way of scaling the data or transforming the data so that the gap between the classes widen and that for it converges quicker right.

But the third problem is the harder one the data is not linearly separable what will happen to the perceptron algorithm right there will be no hyper plane where M is empty so as long as M is not empty I am going to keep adding something to the β again and again right as long as M is not empty I will keep adding it up was every iteration I will be changing the β right it is not linearly separable.

Then it enters then it loops basically then it looks but the problem is sometimes the loops can be very, very large and if you have a very large data sets it might actually be setting up a very large loop so it may not be easy to detect also right so normally it takes a long time to converge that in that if it is looping so you have no way you do not know if that the algorithm is taking a long time to converge or whether it is simply looping so how could you indirect loops.

So you think that every time it has to be a clear decrease if it is going to converge yes right you think so towards convergence yes but when it is taking this whole very long time to actually work through the thing right so if there is no guarantee that cardinality of m monotonically decrease also right so it is not very easy its cardinality MF is not the only thing that will give you so it is yeah has no efficient way to get around that okay so the problem is and yeah.

There is so people discovered a lot of drawbacks to perceptrons and the biggest drawback was they discovered that some very, very simple problems that you want to solve are not linearly separable right so XR so that is XR right so XR is not linearly separable I simply cannot even solve simple problems like X on what a useless thing is I do not care how well you make the baby speak you cannot solve XR .

So I am going to forget about people remember I talked even the example how they trained a perceptron to reproduce speech and ask the training progress it just sounded like a baby learning to speak and people just went over it any way so we'll stop here it looks like I am actually out of time so the next class we will try to look up her I will try to explore a way of fixing this right so

what is the problem there not that it is linearly separable but the fact that it could converge to any solution that is linearly separable we start by trying to define a single optimal solution.

IIT Madras Production

Funded by
Department of Higher Education
Ministry of Human Resource Development
Government of India

www.nptel.ac.in

Copyrights Reserved