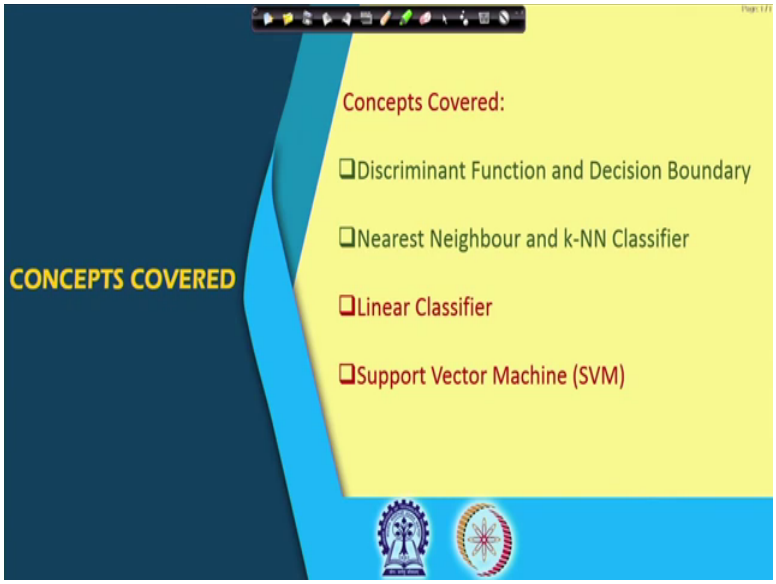
Deep Learning
Prof. Prabir Kumar Biswas
Department Of Electronics and Electrical Communication Engineering
Indian Institute of Technology, Kharagpur

Lecture –10
Linear Classifier – II

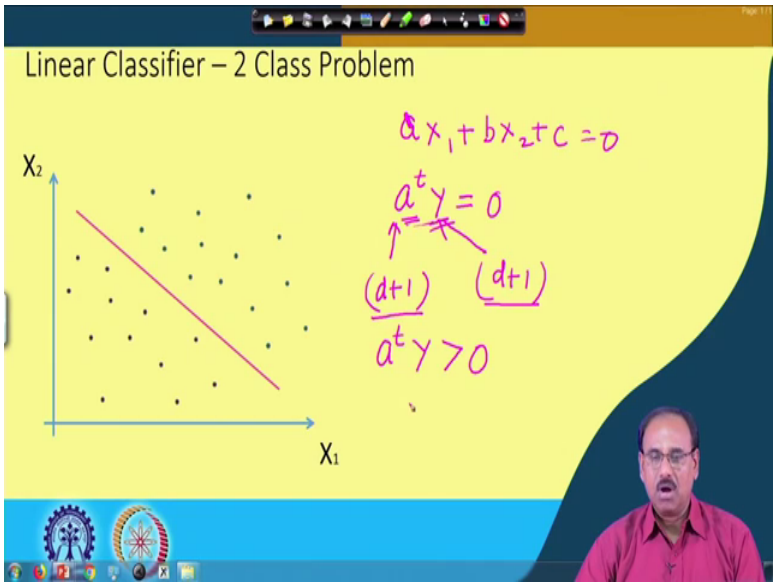
Hello welcome to the NPTEL online certification course on Deep Learning.

(Refer Slide Time: 00:33)



In our previous class, we have talked about we have recapitulated the discriminant function and the decision boundary. We have talked about the nearest neighbour and k-NN or k nearest neighbuor classifier and we had started our discussion on linear classifier. So, today let us continue our discussion on linear classifier and then we will move on to support vector machine.

(Refer Slide Time: 01:03)



So, you see here that the linear classifier that we are discussing over here is for a 2 class problem that is we have samples belonging to 2 different classes say omega 1 and omega 2. And I assume that the samples are linearly separable and in which case given these two training samples from classes omega 1 and omega 2, I can separate these samples using a linear boundary. So, the linear boundary in a two dimensional case will be a straight line having an equation of the form a X 1 plus b X 2 plus c is equal to 0.

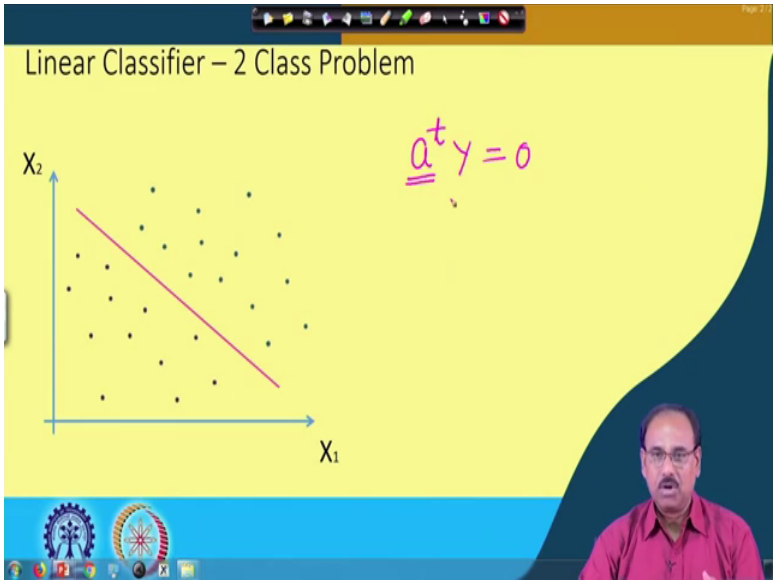
Whereas, in a multi dimensional case, this linear boundary which will be nothing, but on hyperplane will be of the form a transpose Y is equal to 0 where this a is a d plus 1 dimensional feature vector Y is also a d plus 1 dimensional feature vector. So, as we have said in our previous lecture that this a includes the vector which is perpendicular to the plane separating the feature vectors belonging to 2 classes and also the bias term and Y is obtained as appending 1 to the feature vectors of dimension d. So, that is how we get d plus 1 dimensional vector for a and also d plus 1 dimensional vector representing Y that is the feature vectors.

And we have also assumed that for all the feature vectors taken from class omega 1, Y remains as it is; whereas, for all the feature vectors which be taken from class omega 2 Y is negated. This is for the design purpose. Why we have done it? Because while designing the separator a transpose Y equal to 0 for correct or incorrect classification, I

have an uniform decision rule that I should have always a transpose Y greater than 0 if our training vector Y is correctly classified.

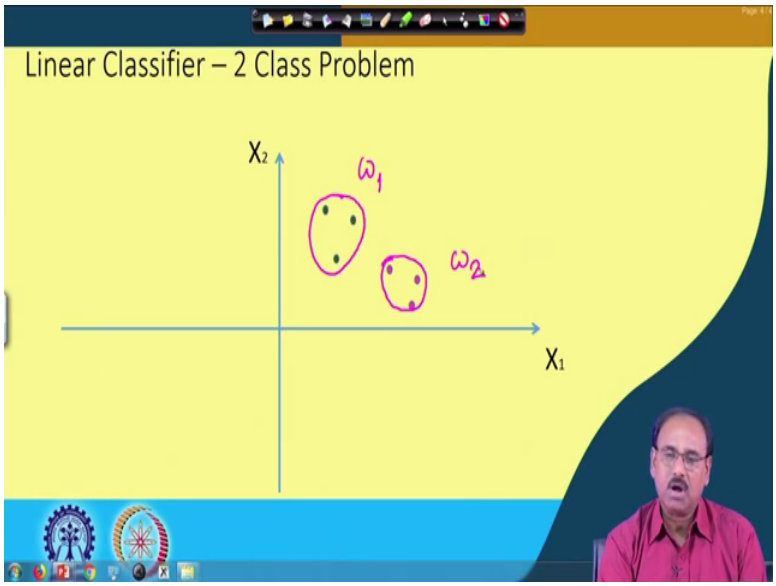
So, our original classification rule was if Y is taken from class omega 1 that then a transpose Y will be greater than 0; if Y is taken from class omega 2, then a transpose Y should be less than 0. So, what we have done is we have simply negated Y for all the samples taken from class omega 2. So, for after negation whether the same samples are taken from class omega 1 or class omega 2, I should always have a transpose Y greater than 0. So, this is simply for the convenience of designing the decision boundary.

(Refer Slide Time: 04:18)



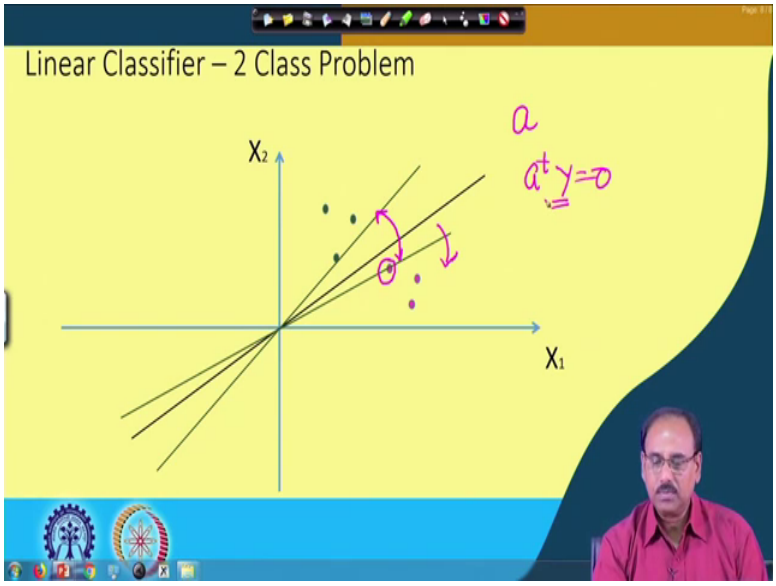
So, given this, now let us see that we have seen that as we have our decision boundary given by a transpose Y is equal to 0; that means, the vector a is orthogonal to the plane separating the two feature vectors right. So, can we have can we try to find out what should be should be the limits on a?

(Refer Slide Time: 04:46)



So, in order to see that let us take a very small example; a simple example. So, something like this here I have samples taken from class omega 1, I have taken smaller sample size for clarity of the pictures. So, these are the feature vectors which are taken from class omega 1 and these are the feature vectors which are taken from class omega 2. So, what I want to have is I want to have a plane which separates these two sets of feature vectors; one from class omega 1 and one from class omega 2.

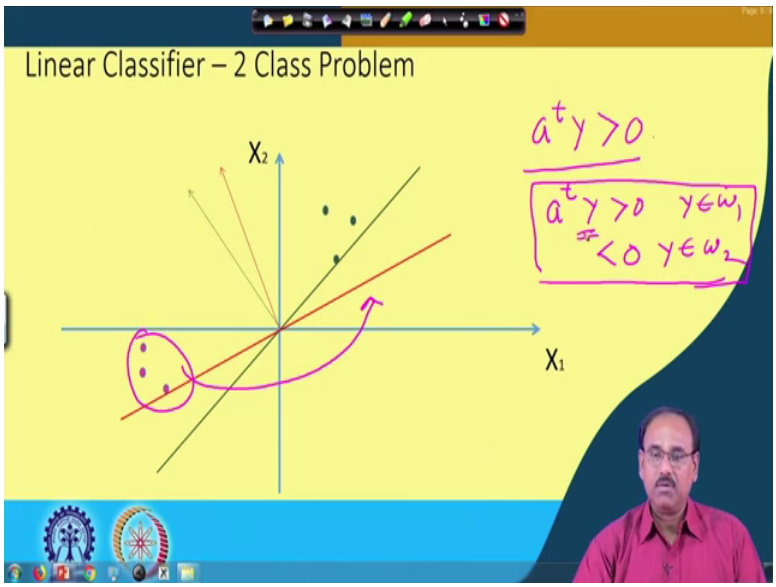
(Refer Slide Time: 05:22)



So, how we do this? So, I am trying to find out obtain something like this. So, you find that this pin clearly separates in the feature vectors from class omega 1 and the feature vectors from class omega 2. Now you find that there is a limit of orientation of this particular plane. If I rotate this plane in say anti clockwise direction. So, this is my new position of the plane, this is the limit to which I can rotate because if I rotate it further, then this is the point which is going to be misclassified.

Similarly coming to the other side; if I take this plane, this is a limit on the other side. Because if I rotate it further in the clockwise direction, then this is a vector which is going to be misclassified. So, that range in which I can have this separating plane between these two classes is given by this. So, this is the separating plane given the separation plane, what happens to our solution vector a because this solution vector a as we have seen is orthogonal to the plane a transpose Y equal to 0 where Ys are the feature vectors taken from class omega 1 and omega 2.

(Refer Slide Time: 06:53)

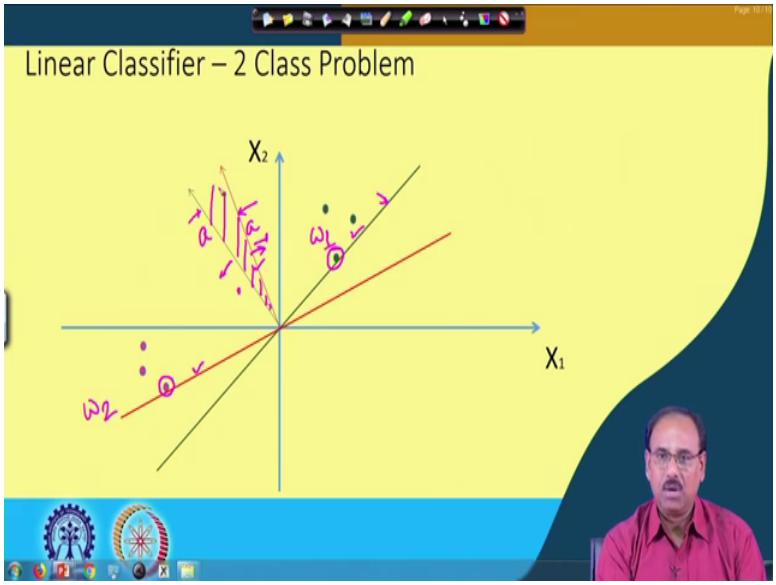


So, if I look at that you find that over here what I have done is in earlier case, our feature vectors these were the feature vectors corresponding to class omega 2 on this side. And what he said is for design purpose, we want to negate this feature vectors. So, that always I will have a transpose Y greater than 0 while designing. This is my decision for correct classification and I can always do it because now the feature vectors are the training vectors. So, for every vector I know what is this class belongingness.

But you remember once your classifier is designed; that means, once your separating plane is designed, my decision rule will always be or the classification rule will always be that if a transpose Y is greater than 0 then Y should belong to class omega 1. If it is less than 0, then Y will be classified to class omega 2. This is what we will do while classification or during testing because in that case the feature vector Y that we went to classify I do not know from which class this has been taken because that is what I have to decide.

But during design with the training vectors, I know from which from which class which training vector has done. So, I can negate Y in that case for all the Y taken from class omega 2 and that is what has been done over here.

(Refer Slide Time: 08:28)



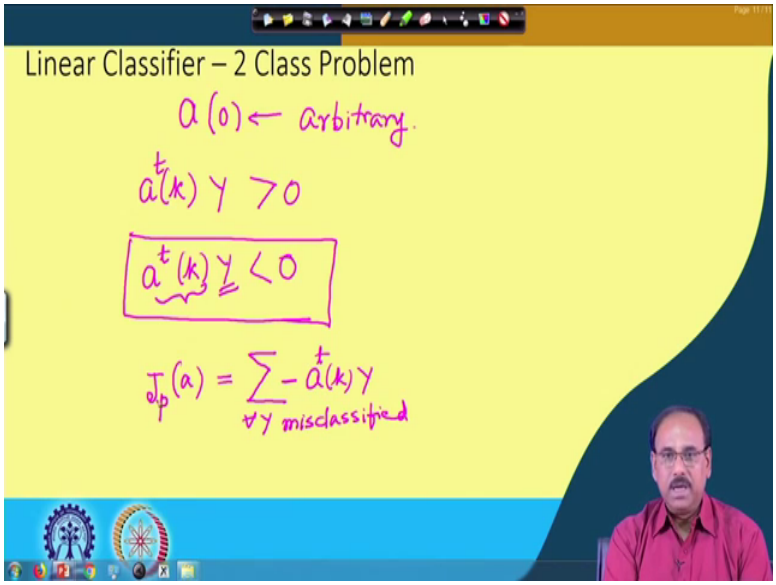
So, given this you will find that when I have this limiting position of the separating plane the vector a is this one which is orthogonal to this particular plane. Similarly on the other limiting side when this is the position of the separating plane, then my solution vector a is this one. And you find that if the solution vector a is rotated towards left from this limiting position, then this is a vector which is going to be misclassified.

Similarly if the feature if the solution vector a is rotated in the clockwise direction from this position, then this is a feature vector belonging to class omega 2 which is going to be misclassified. So, that clear clearly tells me that I have a solution region and my solution vector a vector which is perpendicular or orthogonal to the separating plane must lie

within this solution region. If it is outside this, then vectors belonging to class omega 1 will be misclassified; if it is outside on this side, then the vector belong to class omega 2 are going to be misclassified.

So, my solution vector must lie within this conical region. So, how I can have a feature vector of this solution vector of this problem?

(Refer Slide Time: 10:11)



So, for doing this, I start my algorithm like this. So, initially I assume that at 0 th instant, my feature vector a 0 is chosen arbitrarily right and we have I have also said that at any k th step in my iteration, I have a solution vector say a k. And it is possible that this solution vector a k at the k th step, we will correctly classify for some of the samples and will be miss correctly classify some other samples.

So, for all the samples which are correctly classified, I will always have a k transpose Y greater than 0 and for all the samples which are incorrectly classified, I will have a k transpose Y less than 0. Irrespective of whether the vector is taken from class omega 1 or taken from class omega 2 because all the training vectors taken from class omega 2 have been negated.

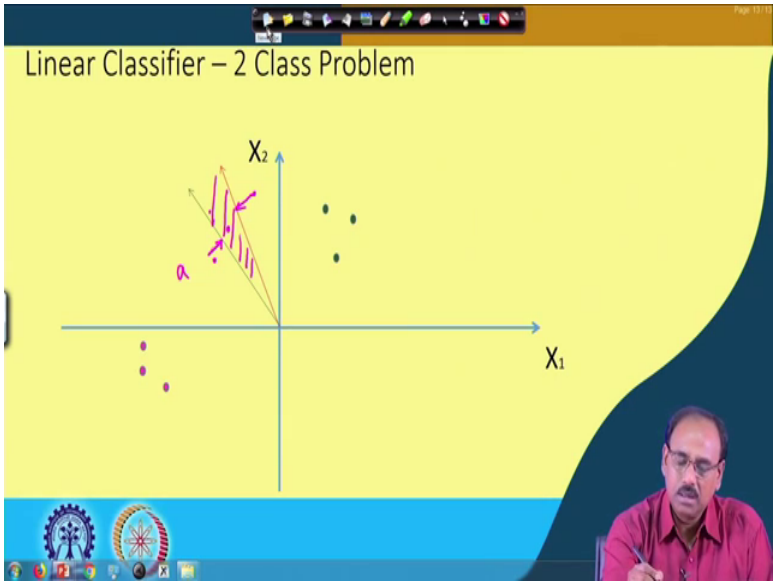
So, during design or during training whenever I come across this type of situation that a k transpose Y less than 0, immediately I can conclude that current a k has misclassified

the vector Y. And for this miss classification, I have to generate an error term and the error term that can be generated is a k transpose Y and as it is negative, I negate this.

So, whenever a k transpose Y is negative, I generate an error term which is minus a k transpose Y which will always be positive; that means, if I have any misclassified sample by a k, I will have a positive error and this I do for all the samples which are misclassified by a k. So, you have take the summation of this over all Y which are misclassified and I call it the error J as a function of a because as all the Ys are fixed because those are my training samples, but what I can vary is a.

So, I represent this as an error j which is a function of a and this is an error which is normally called perceptron criteria. So, sometimes it is also written as J p a.

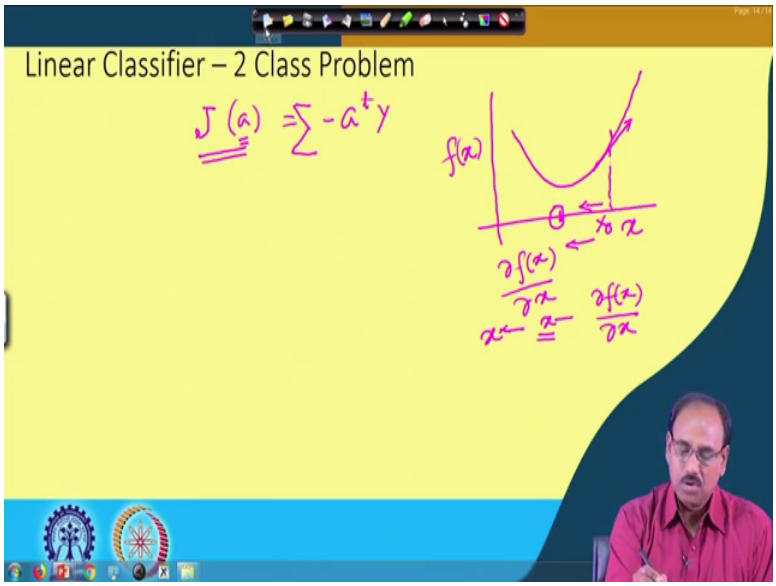
(Refer Slide Time: 13:07)



So, what is my solution approach? If you look at the figure over here, you find that this was my solution region. So, if I have at any instant of time a solution vector a over here, my approach should be that I should be able to push this a towards this side. So, that it moves inside the solution region or at any instant of time if my solution vector a is on this side, I should push it in this direction or push it towards the solution region.

So, that eventually my vector will land in the solution region and I get the proper separating boundary. So, what is the approach that I should take for this?

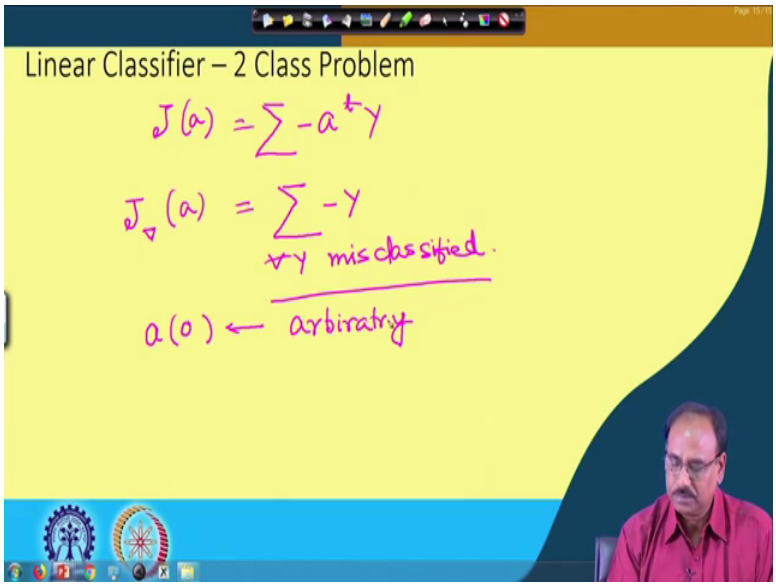
(Refer Slide Time: 14:00)



The approach can be that as I said that my error J a is given by a transpose Y minus sum of this. So, I can take an approach to reduce this error or to minimize this error by following and algorithm known as gradient descent algorithm. What is gradient descent algorithm? Let us take a very simple case something like this. Say I have a variable X and I have a function f x and the function f x is something like this and iteratively I want to get a value of x for which f x is minimum. So, somewhere over here as in this case, I want to get a value of a which will minimize this error J a.

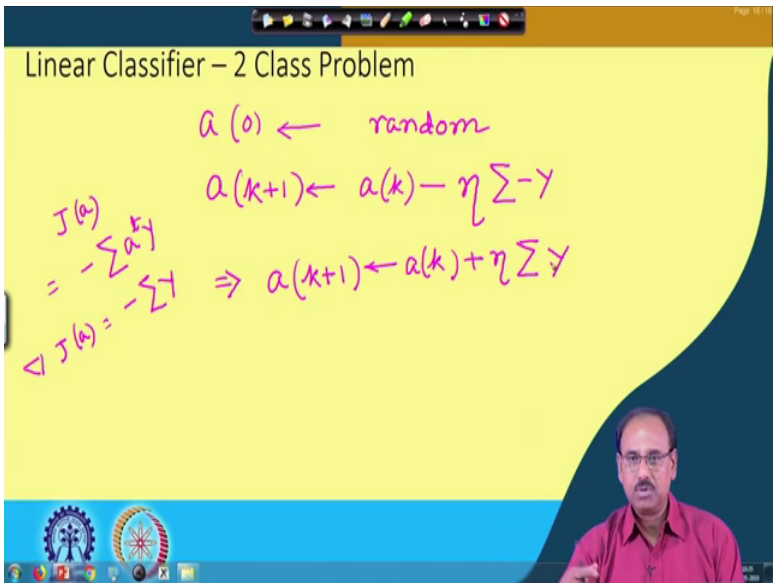
So, you find that here what I can do is if I start somewhere over here so, this is my x naught; I will take the gradient of fx with respect to x. So, what I will compute is del f x upon del x and over here the del f x upon del x will be this. What I do is I move x in the negative direction of del fx del X. So, I move in this direction. So, my x next time will be the previous x minus del f x del x and if I take this movement; if I change x in the negative direction of the gradient, I am moving in the direction of minimum f x. So, this is what is known as gradient descent algorithm and we will talk more of this in details later. But for the time being let me take that I will use this gradient descent algorithm to minimize my error term J.

(Refer Slide Time: 16:10)



So, what I have is I have J a is equal to a transpose Y negative of this sum of this over all Y which are misclassified and what are I want to perform is I want to take the gradient of J with respect to a and which is nothing, but from here it is Y minus take the summation over all Y which are misclassified right. So, my decision rule can be as we said before that initially at the 0 th instant, I assumed that a 0 will be chosen arbitrarily or at random.

(Refer Slide Time: 17:13)

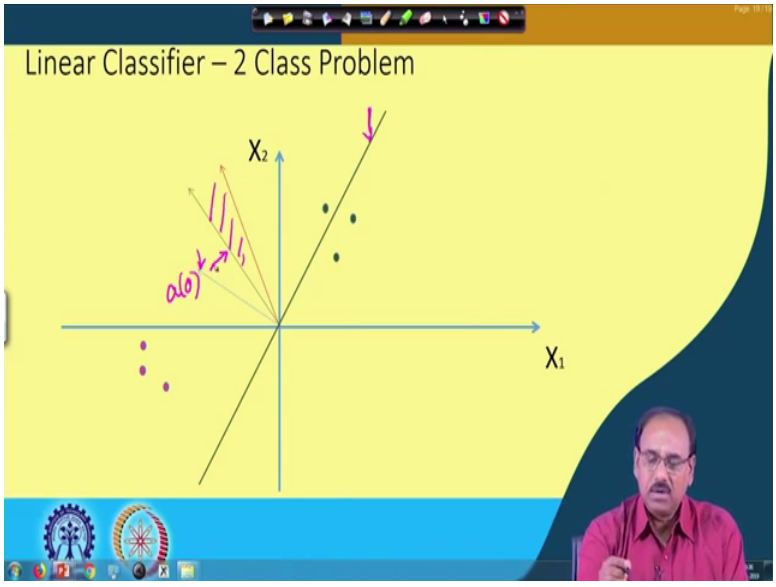


So, the better term is I choose a 0 at random and then if I have a k at k th iteration from there, I want to find out a k plus 1 which should reduce my error J a which was of the form minus a transpose Y and for that I go for the gradient descent approach.

So, gradient of J a, we have seen was minus sum of Y for all Y which are misclassified. So, I can get a k plus 1 from the previous value a k as a k minus sum of Y and I can put a rate of convergence which is eta. So, that simply becomes my weight updation rule or vector updation rule as a k plus 1 same as a k plus eta times sum of Y for all Y which are misclassified. And you remember that when I am talking this Y talking this feature vector Y, the Y will be as it is for the samples taken from class omega 1 and Y will be negated it is negated vector for all the vectors taken from class omega 2.

So, given this, now let us see how this rule upgradation rule actually tries to refine a k plus 1 or the weight vector. So, that eventually the weight vector pushes falls in the solution region.

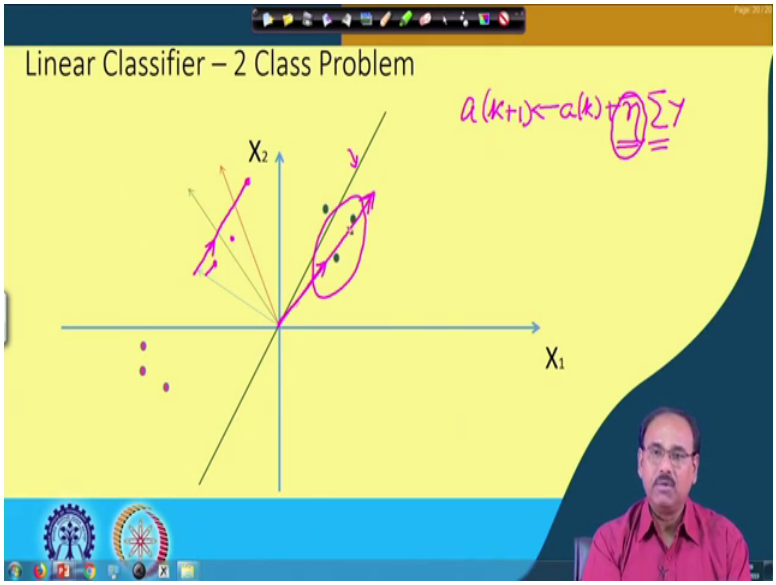
(Refer Slide Time: 19:24)



So, this is what we were talking about. This is my solution region; solution region is within this. Now if I take an initial separating plane something like this. So, I am assuming that this is my initial separating plane and the weight vector a or a 0 is this one which is orthogonal to the separating plane. So, here you find clearly that the initial solution vector which I have chosen at random because this is the pending plane has been chosen at random, it falls outside the solution region because the solution region is this.

So, naturally I have to upgrade or I have to modify this solution vector a 0. So, that it is moved towards the solution region over here.

(Refer Slide Time: 20:33)



And for that what updation rule we have used, the updation rule was a k plus 1 is a k plus eta times sum of Y where Y is misclassified. So, what is the misclassified sample in this case? I have these two samples which are misclassified by this separating vector right. So, if you take the sum of these two feature vectors which are misclassified, I will get a feature vector; I will get the sum of the misclassified Y which is in this direction and this sum is scaled by sum eta which is known as rate of convergence. I will come to what is the importance of the state of convergence later.

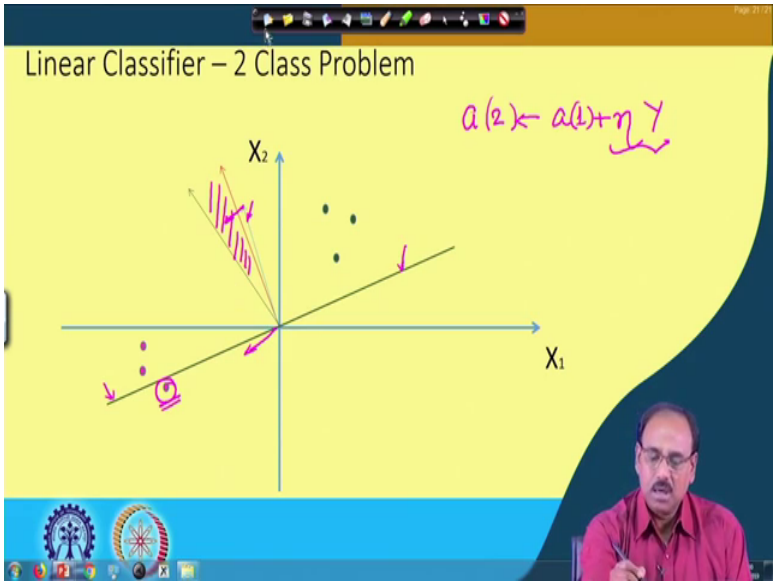
So, this is a k or a 0; you add this sum of Y or sum of these vectors this particular vector scaled by eta to this vector to get your a 1. So, a 1 will be moved in this direction because sum of Y misclassified Y is in this direction. So, eta times this is added to this initial vector which was chosen at random at the vector moves towards this direction.

Now what is the importance of the state of convergence eta? If the value of eta is very high, then it is possible that I will jump this solution region and my modified vector will be somewhere over here. So, you find that you have crossed the solution region for a larger value of eta. If the value of eta is very small maybe I will be landing somewhere over here which is again not in the solution region. So, if the value of eta is very small, then your rate of convergence or the rate at which you are moving towards the solution is

small. If the value of beta eta is very high, then that risk is that you may overshoot the solution region and to go to the other side of the solution region. So, your number of iterations to reach the solution will be larger.

However, if the value of eta is appropriate and then, it is possible that your next modified solution vector the vector that you obtain will be in your solution region and which is the solution that you are looking for. Now whatever it is, once I have these misclassified samples using misclassified samples you refine the weight vector a.

(Refer Slide Time: 23:13)

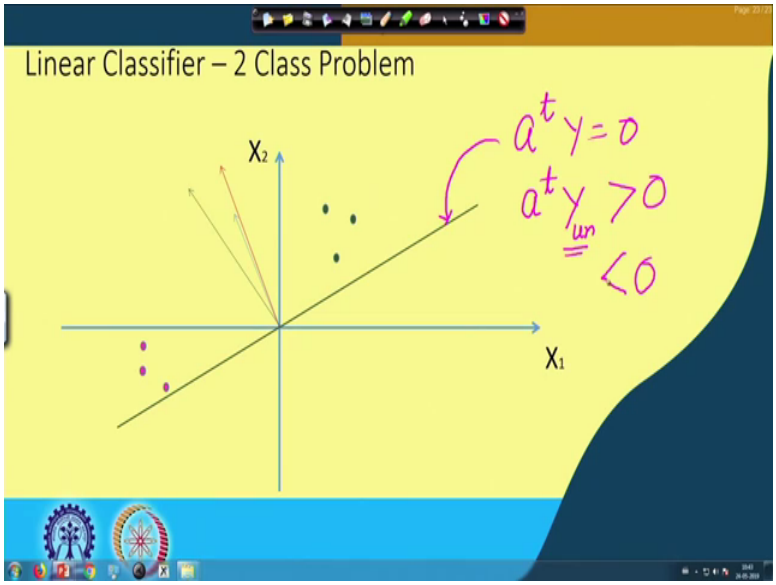


And for by this refinement, it is possible that the next weight vector; the next separating plane that you will land that you will achieve is this one. And you find that for this separating plane, the weight vector is this which is again as we said that if the value of eta is very large; in that case, you can cross the solution region you can overshoot the solution region and that is what exactly has happened over here.

So, this is my weight vector which is falls in which falls on the other side of the solution region and the sample which is misclassified by this separating plane is this one right. So, at the next weight vector that I should get which is a 2 that gets from a 1 that was the previous solution vector which is this will be a 1 plus eta times this misclassified sample Y. And as this misclassified sample is in this direction, this solution vector as I add eta times Y two this will also move in this direction.

So, it is possible that the next solution that I obtain will be within this solution region is somewhere over here. So, this approach clearly says that by taking the gradient descent approach to reduce the error, it is possible that I will get eventually I will get a separating plane which separates the 2 classes of the feature vectors the samples belonging to class omega 1 and class omega 2.

(Refer Slide Time: 24:55)



And that is possible when I have the samples which are really linearly separable. If the samples are not linearly separable, then it is not possible to obtain a separating boundary is something like this. And the approach that we have to take in such cases is what is known as minimum error criteria so, that the separating plane that you obtain does not totally remove the error, but the plane will try to reduce the squared error or the minima or it will try to minimize the squared error of classification.

So, in today's lecture what we have done is we have tried to find out a boundary between the samples taken from the 2 different classes, the samples which are provided for designing the separating plane or the training samples. And for designing this separating plane what we have done is we have appended one to all the attaining vectors and the vectors taken from class omega 1 which actually falls on the negative bound; negative half of the boundary.

We have negated them for design purpose only, but you keep in mind that once you are separating plane is properly designed or once I get this equation a transpose Y equal to 0

which is the equation of this separating plane by using the training vectors for classification of the unknown sample. Now my classification rule will be for an unknown sample say Y unknown; if it is greater than 0, then Y unknown is to be classified to class omega 1; if it is less than 0, then Y unknown has to be classified to class omega 2.

So, with this, we will stop today's lecture. Next class, we will talk about the support vector machine.

Thank you.