

Pattern Recognition and Applications
Prof. P. K. Biswas
Department of Electronics and Electrical Communication Engineering
Indian Institute of Technology, Kharagpur

Lecture - 20
Perceptron Criterion

Good morning, so we are discussing about pattern recognition problem, and we are talking about the different discriminant functions, which are helpful for classification of unknown patterns. So, earlier what we have seen is starting from the base decision theory that if you know the probability density function of the parametric form of the probability density function. We can design the different discrimination functions for pattern classification or classification of pattern vectors or feature vectors. We have discussed about specific case that, if the probability density function is either Gaussian or Normal distribution function.

And we know the mean and co-variance matrices of the different classes, then we can have different types of discriminant functions or different types of decision boundaries among different classes. In particular if the co-variance matrices of the samples belonging to all the classes are same then we have seen that discriminant function that we get is a linear discriminant function. Whereas, if different classes of the samples belonging to different classes they have different co-variant matrices. In that case in general the kind of discriminant function that we get is a quadratic discriminant function.

So, in multi dimension it is called hyper quadrics. And accordingly the decision between two different classes in one case is hyper plane. And if the co-variance matrices are arbitrary or different classes have different co-variance matrices, then the decision between two classes becomes a hyper quadric surface. However, following the base decision theory if we want to have the discriminant function, we have to know that, what is the parametric form of the probability density function of the samples belonging to that particular class.

And given the number of samples if we know the parametric form of the probability density function, we may not know the parameter values exactly. If we know the value of the probability density function then we have seen in previous classes that we can have the maximum likely hood estimate of those parameter values. So the maximum likely

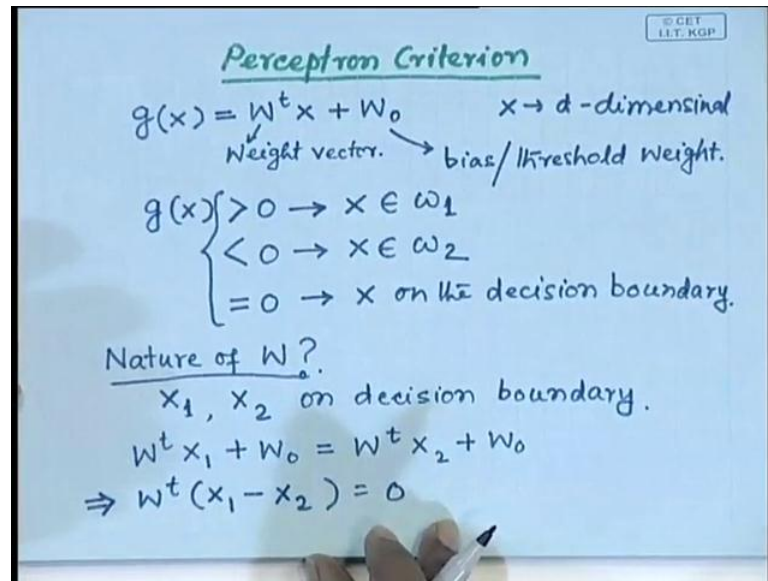
hood estimation says that these are the best representations or best supported by the samples, which are available from that particular class.

And then also we have discussed about the dimensionality problem because not every feature has equal discriminating power. So, following the principle component analysis or following multiple discriminant analysis, we can reduce the dimensionality of the feature vectors ensuring that the features in the reduced dimensional feature vector will have better discriminating power than the features which are avoided. Now, today we will discuss about this same linear discriminant function, but unlike in case of what we have obtained from this decision theory that we have to know the parametric form of the probability density function.

In case the parametric form of probability density function is not known. So, then what you have to do is we have to design the discriminant function or we have to find out the decision boundary between two different classes by using the samples which are available from different classes. So, here we do not assume any probability density functions. So, we know so initially we will discuss about the linearly separable classes. So, we know that the classes are linearly separable and because we are discussing about the supervised learning process.

So, we know a set of samples belonging to one class ω_1 , we know set of samples belonging to another class ω_2 . So, using this information and without using any assumption of the nature of the probability density function. We have to find out the linear discriminant function which discriminates between these two different classes. So, we will assume that the classes are linearly separable. And as the classes are linearly separable, so we should be able to formulate we should be able to design linear discriminant function. Well, no probability density function form is assumed so because the discriminant functions will be linear.

(Refer Slide Time: 05:24)



So, we know that a linear equation can always be written as $g(x)$ is equal to W transpose x plus W naught. For x is the feature vector. And we assume that the feature vector x is of dimension d . So, it is d dimensional feature vector. And W is called weight vector and W naught is called the bias or threshold weight. So, we see that this particular expression is a linear expression. For x is a d dimensional vector, W is also a d dimensional vector. And this term w transpose x is nothing but inner product of vector W with a vector x . And our decision criteria will be therefore, an unknown feature x , if $g(x)$ is greater than 0. Then we classify x to belong to class ω_1 , if it is less than 0 then x will be classified to class ω_2 .

And if it is equal to 0 then we say x is on the decision boundary. So, if $g(x)$ is greater than 0 we decide x belongs to class ω_1 , if it is less than 0 we decide x belongs to class ω_2 , if $g(x)$ is equal to 0 then x is on the decision boundary. So, the decision boundary between two classes x , one ω_1 and ω_2 is given by the equation $g(x)$ is equal to 0 or W transpose x plus W naught that is equal to 0. Now, let us see that what is the nature of this weight vector W ?

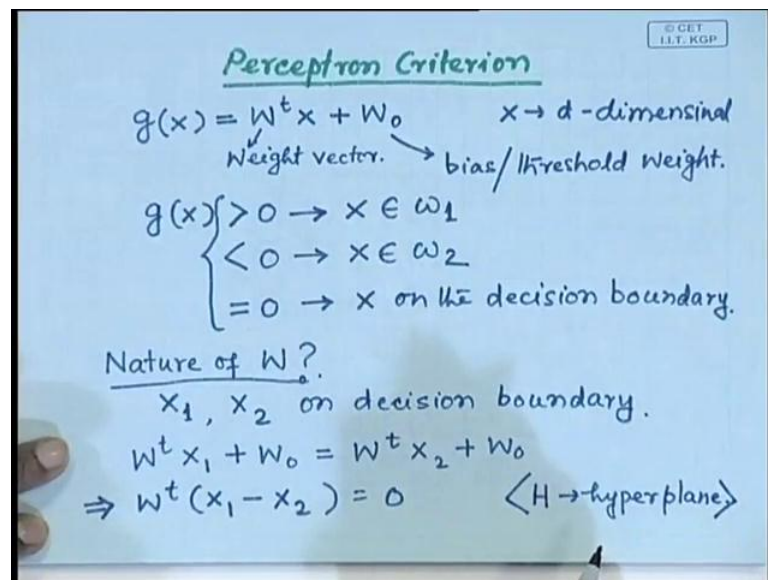
So, to find out the nature of the weight vector W let us take two points on the decision boundary. So, what we are interested in the nature of weight vector W . So, we take two points x_1 and x_2 on decision boundary. Now, because x_1 and x_2 are on the decision boundary so W transpose x_1 plus W naught will be equal to W transpose x_2 plus W

naught. And both of them equal to 0 because X_1 and X_2 are lying on the decision boundary.

So, we can simply write that $W^T X_1 + W_0$ is equal to $W^T X_2 + W_0$. So, from this you find that $W^T X_1 - X_2$ this is equal to 0. Now, what is this $X_1 - X_2$, $X_1 - X_2$ is nothing but, a vector lying on the decision surface, because X_1 is on the decision surface, that is a vector X_2 is also a vector that is lying on the decision surface $X_1 - X_2$ is a vector which is lying on the decision surface.

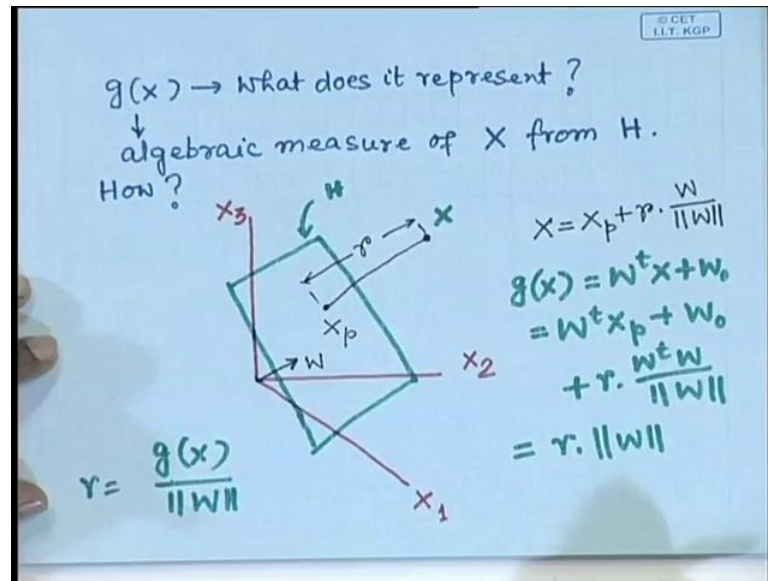
And $W^T X_2$ is the inner product of W with the vector $X_1 - X_2$, and because this inner product is equal to 0. So, that clearly indicates that this vector W is orthogonal to any vector line on the decision surface. Because we have taken X_1 and X_2 arbitrarily so $W^T X_1 - X_2 = 0$. That indicates that vector W is orthogonal to any vector lying on the decision surface that means the vector W is orthogonal to the decision surface. And this surface being a planar surface or a linear surface in d dimensional space we call it a hyper plane.

(Refer Slide Time: 10:51)



And let us represent that hyper plane H , H is the hyper plane. So, this vector W is orthogonal to hyper plane H . Now, what does this $g(x)$ actually represent?

(Refer Slide Time: 11:21)



You will find that $g(X)$ gives you some measure of the algebraic distance of vector X from the decision surface or from the height of the H . So, this $g(x)$ gives an algebraic measure of X from H . Let us see how? How you say that $g(x)$ represents an algebraic measure? It is not the exact distance value, but it is major of the distance value and is something to proportional to the distance algebraic distance of the vector X from the decision surface from the hyper plane H . How does it give us that measure? Let us consider a case in 3 dimensions.

So, I have a 3 dimensional space with the vector components represented by X_1 , X_2 and X_3 . Suppose, I have a decision surface something like this, in three dimensional planes. So, this is my plane H and let me take a point X somewhere over here. And when I take this point X over here, let me drop a perpendicular from this point X onto the hyper plane. And let me assume that X_p represents the foot of the perpendicular on this hyper plane. And suppose the distance between X_p and X , the length of this vector joining X_p and X is given by say r .

Now, if it so in that case I can write X is equal to X_p plus r , r is the distance multiplied by W upon mod of W , because we have seen earlier that the vector W is orthogonal to the hyper plane orthogonal to the decision surface. So, the direction of the W is in the same direction of X_p to X , because X_p to X this vector is orthogonal to this hyper plane. W is also orthogonal to the hyper plane. So, this is W . So, I can write this X is equal to X

p plus r into W upon mod of W because W upon mod of W is the unit vector in the direction perpendicular to the hyper plane.

So, I can write it in this form. And when I write it like this then what is $g(X)$, $g(X)$ is nothing but $W^T X + W_0$ which in this case will be $W^T X_p + W_0$ plus r into W upon mod of W is it. I am simply replacing this X by this $X_p + r$ into mod of W . Now, out of this the point X_p which is of the perpendicular from X onto the hyper plane that lies on the decision surface. So, $W^T X_p + W_0$ because X_p is on the hyper surface $W^T X_p + W_0$ will be equal to 0 because it is on the decision surface.

So, what I get is this will be simply r into and what is $W^T W$ this is nothing but mod of W square. So what is simply get is r into mod of W . So, find that this discriminate function $g(X)$ the value that you get is nothing but r we said is the distance perpendicular distance or orthogonal distance of point X from the decision surface, because it is the distance between X_p and X . So this r is the orthogonal distance of point X from the decision surface.

So, this $g(X)$ is the orthogonal distance scaled up by the modular's of W . And if I want to find out what is the actual orthogonal distance that is given by r is equal to $g(X)$ upon mod of W . So, that is why we said that this discriminate function $g(X)$ actually gives you a measure algebraic measure of the distance of the point orthogonal distance of vector X from the decision surface. Now, why are we calling it algebraic distance?

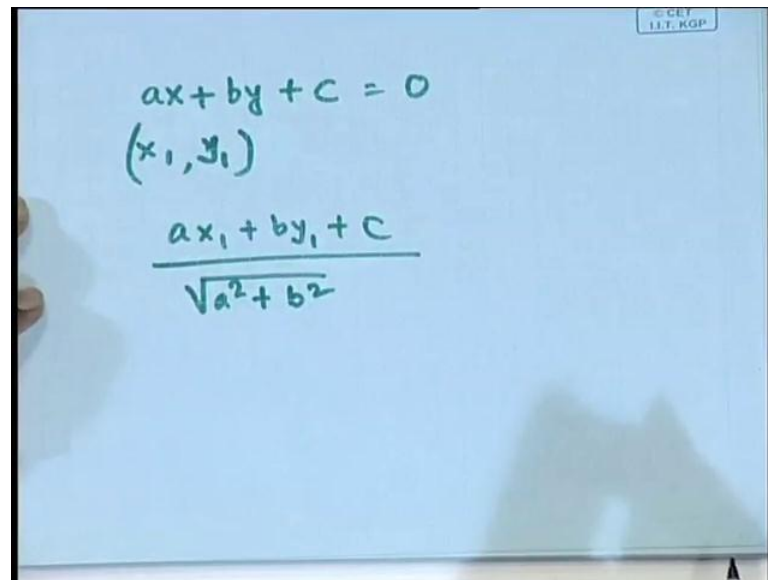
Student: Sir, here only the first term ((Refer Time: 18:25)) which one ((Refer Time: 18:30)).

You come to our original expression $g(X)$ is equal to $W^T X + W_0$. This is your discriminate function. What you said is if $g(X)$ is greater than 0 then belongs to ω_1 . If $g(X)$ is less than 0 then X belongs to ω_2 . If it is equal to 0 then X is on the decision boundary. That is $g(X)$ is equal to 0 not $W^T X$ equal to 0. And $g(X)$ is $W^T X + W_0$. So, because X_p is lying on the decision surface so $g(X_p)$ will be equal to 0.

And $g(X_p)$ is nothing but $W^T X_p + W_0$. So, here why we are calling it as algebraic distance is that if r is positive then X lies on the positive side of the hyper

plane. If r is negative then X is on the negative side of the hyper plane. And this expression that I have said that r is equal to $g \cdot X$ upon mod of W . This is an expression which is nothing new I mean this is what you have already done in your school level mathematics.

(Refer Slide Time: 20:31)

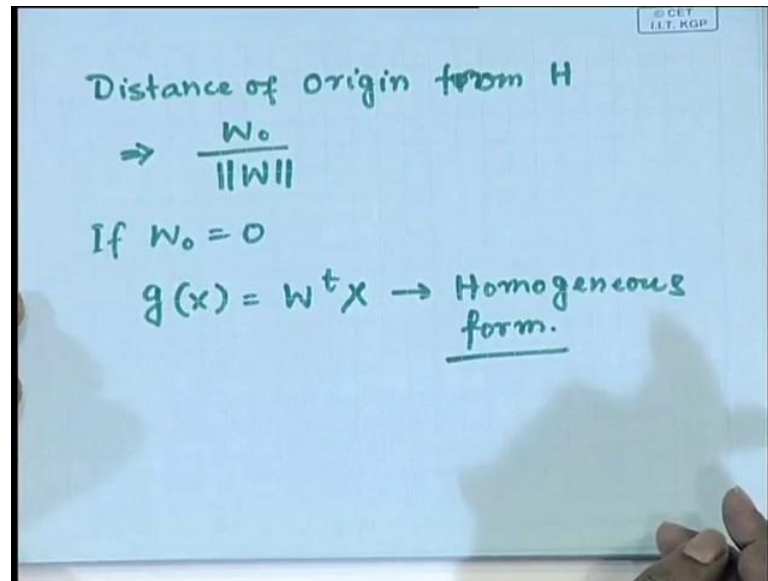


A photograph of a whiteboard with handwritten mathematical formulas. The top line is $ax + by + c = 0$. Below it is a point (x_1, y_1) . The bottom line is the expression $\frac{ax_1 + by_1 + c}{\sqrt{a^2 + b^2}}$. In the top right corner of the whiteboard, there is a small logo that reads 'CET 11.7. KGP'.

We know that in school level mathematics what you have done is, if I have the equation of a straight line given say $a x$ plus $b y$ plus c is equal to 0 . Come to your co-ordinate geometry. And given a point $x_1 y_1$ what is the distance of point $x_1 y_1$ from this straight line which is nothing but $a x_1$ plus $b y_1$ plus c upon square root of a^2 plus b^2 . And if you compare this expression with this expression they are exactly identical. This expression is in the 2 dimensional, this expression is in d dimensional for the d dimensional ((Refer Time: 21:21)).

So, what you have derived over here is nothing new, this is something that you have already done in your school level mathematics. So, that we are saying is that if r is positive then the vector X lies on the positive side of the hyper plane. If r is negative then X lies on the negative side of the hyper plane. And what is the distance of origin from the hyper plane?

(Refer Slide Time: 22:03)



So, distance of origin from the hyper plane that is simply given by W naught upon mod of W . So here if W naught is positive then origin lies on the positive side of the hyper plane, if W naught is negative then origin lies on the negative side of the hyper plane. And if W naught is equal to 0 then the hyper plane passes through the origin. So, if W naught is equal to 0 then the hyper plane passes through the origin.

And not only that your discriminate functions takes a particular form that $g X$ will be simply given by W transpose X , but I do not have any bias or threshold. And this is the form which is called homogeneous form. And in mathematics it is always convenient that I can represent an expression in a homogeneous form that avoids many of the problems that we may face while we try to analyze different given topics. So, after I will discuss about the nature of the decision surface if it is hyper plane then what are the conditions or what are the different positions of the hyper plane.

That we can have then the hyper plane has two sides. Actually, hyper plane divides your feature space d dimensional space into two half spaces. One of the half spaces is positive the other half space is negative. If a feature vector falls on the positive half space then it will be classified into one task. If the feature vector falls on the negative half surface it will be classified into another class. If it is on the decision surface then we cannot classify it actually, because it is equally likely that it may belong to class ω_1 or it may belong to class ω_2 .

So, if the sample falls on the decision surface or falls on the hyper plane, we cannot really take any decision, it is an ambiguous condition. Now, after discussing this let us see that how we can design this vector W , ultimately because it is a linear class. I have to find out the discriminate function $g(X)$ and if I can find out the weight vector W and the bias of the threshold weight W_0 then my linear classifier is designed.

So, I have to design or I have to decide about W and W_0 . Based on the samples that are available because it is supervised learning so we are given samples from both the classes ω_1 and ω_2 . So, I have to decide W and W_0 based on the information available from those known labeled samples.

(Refer Slide Time: 25:46)

The slide shows the following content:

Design of weight vector W
 → Two category linearly separable case.
 $g(x) = W^t X + W_0$
 $\approx a^t y$

$y = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \\ 1 \end{bmatrix}$

$\begin{bmatrix} W_1 & W_2 & \dots & W_d & W_0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \\ 1 \end{bmatrix}$
 $= \sum_{i=1}^d W_i x_i + W_0$
 $= W^t X + W_0$

So, let us talk about design of the weight vector W . And here again for simplicity initially, we will assume that we have two category linearly separable case. Later on we will generalize this to multiple category problems or we have c number classes, but to start with let us start with on the two category phase that is we have classes ω_1 and ω_2 . So, what we have discussed so far is we know that we have a discriminate function of the form $g(X)$ is equal to W transpose X plus W_0 .

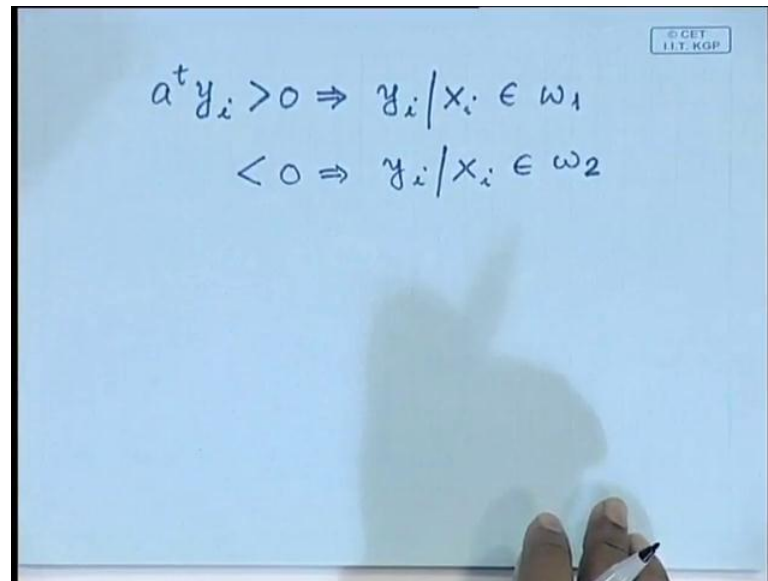
And as we said that this expression of the discriminate function this is not in the homogeneous form. However, if I can convert it into homogeneous form then my analysis will be easier or my design will be easier. So, how we can convert this into homogeneous form I can very easily convert this into homogeneous form. If I write this

expression in the form $a^T Y$ where, a is my modified weight vector. And this modified weight vector will include all the components of the weight vector W and also the bias weight or the threshold weight W_{naught} .

So, if this weight vector a has to include all the components of W as well as W_{naught} then I have to do some augmentation in the feature vector X . So, what I will do is, I will write this Y is actually augmented feature vector X . So, what will be y will be nothing but, I have to take all the components of x . So, I have to take the first component x_1 second component x_2 , all the d components of x that is up to x_d . And I make last component is equal to one. So, if I do that for every x if I augment one to give me an augmented feature vector y .

Then I can write this expression in an homogeneous form as $a^T y$. Simply, because what is this $a^T y$, $a^T y$ is nothing but w_1, w_2 up to w_d, w_{naught} . This is a row vector because I have to take transpose multiplied by column vector x_1, x_2 up to x_d then one. So, if you do this multiplication, this multiplication is nothing but, summation of $w_i x_i, i$ varying from 1 to d plus so up to d components I have taken care by this summation term $x_1 w_1$ plus $x_2 w_2$ plus $x_3 w_3$ up to plus $x_d w_d$. So, this is $x_i w_i$ times x_i, i varying from 1 to d plus w_{naught} into 1. So, plus w_{naught} right which is nothing but $w^T x$ plus w_{naught} . So, I get exactly the same expression as this. Only thing I have to do is I have to augment one to the feature vector x giving me the augmented feature vector y . And after having this homogeneous expression my decision rule remains the same.

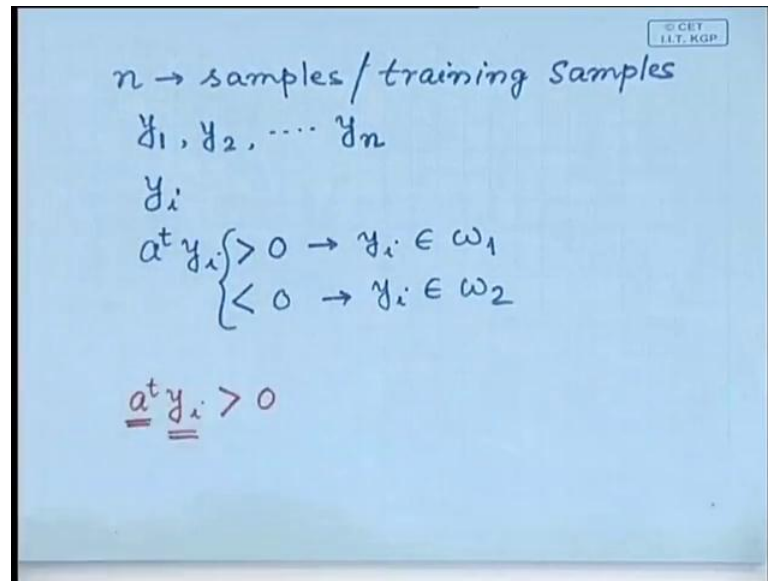
(Refer Slide Time: 31:04)


$$a^t y_i > 0 \Rightarrow y_i / x_i \in \omega_1$$
$$< 0 \Rightarrow y_i / x_i \in \omega_2$$

That if a transpose y^T , for the i th augmented feature vector x_i is greater than 0. Then I decide that y_i of the corresponding or inner vector x_i that belongs to class ω_1 . And if it is less than 0 then I decide that y_i of the corresponding feature vector x_i , do not confuse between this subscript and this subscript here. I have meant this subscript means the i th component of the feature vector. And this means i th feature vector. So, if it is less than 0 then my decision will be this belongs to class ω_2 .

So, my decision rule remains the same. Only thing is I have a modified formulation of the same problem. Now, let us see that how we can really design this weight vector W or in this case the weight vector a because a consists of all the components W and also the bias vector or the threshold vector w_0 . So, our aim is to design the weight vector a by using the knowledge of the leveled samples from class ω_1 as well as class ω_2 . So, we assume that we have n number of samples and when I say n number of samples that means we have n number of augmented samples where I have augmented appended one to each of the feature vector.

(Refer Slide Time: 33:07)



So, I have n number of samples called training samples, because these are the samples which are used to train the classifier. So, this n number of samples ω_1, ω_2 up to ω_n . So, each of this y_i 's are actually augmented feature vectors by appending one to the original vector x . Some of this samples some of this feature vectors are labeled as class ω_1 . Some of this feature vectors are labeled as class ω_2 . Now, for a particular feature vector augmented feature vector y_i as I say the earlier my decision rule will be a transpose y_i greater than 0 indicates y_i is in class ω_1 , less than 0 indicates y_i is in class ω_2 .

If it is equal to 0 I cannot take any decision. So, what I have to do is I have to find out the value of this weight vector w . I have some samples which are labeled as class ω_1 . I have some samples which are labeled to belong to class ω_2 . Now, given a weight vector a , if I take all the samples which are labeled as ω_1 , If I find that for each of those samples $a^t y_i$ is greater than 0. That means that given weight vector a is correctly classifying all the samples, all the feature vectors which are labeled as class ω_1 .

If I also find that for the same weight vector all the samples belonging to class ω_2 , which are labeled as belonging to class ω_2 . For all of them a transpose y_i is less than 0 then the feature vector then the weight vector a is also classifying the samples which are labeled as belonging to class ω_2 . So, that

particular which we have obtained, that is the correct word vector because it is classifying correctly all the samples which are in class ω_1 . It is also classifying correctly all the samples which are labeled as ω_2 .

So, that is a feature vector that we want, but how to obtain such a feature vector. So, to obtain such a feature vector another modification that can be done is that now, you say that I have two decision criteria. In one case I am checking that it has to be a transpose y_i has to be greater than 0. In another case I am checking that a transpose y_i have to be less than 0. So, instead of having these two different criteria cannot we have single criteria, that if some condition is true, a single condition is true I say that the sample is classified category.

If the single condition that is not true, I decide that the sample is not classified category. So, cannot I make something like this that, if a transpose y_i is greater than 0, I will say sample is correctly fixed irrespective of whether this feature vector y_i is labeled as belonging to class ω_1 or it is labeled as belonging to class ω_2 . So, I do not loop at the level of feature vector y , whatever be its level if I can somehow convert this decision in the form that a transpose y_i irrespective of level of y_i . If a transpose y_i is greater than 0 I will say that y_i is correctly fixed otherwise I will say that it is miss classified.

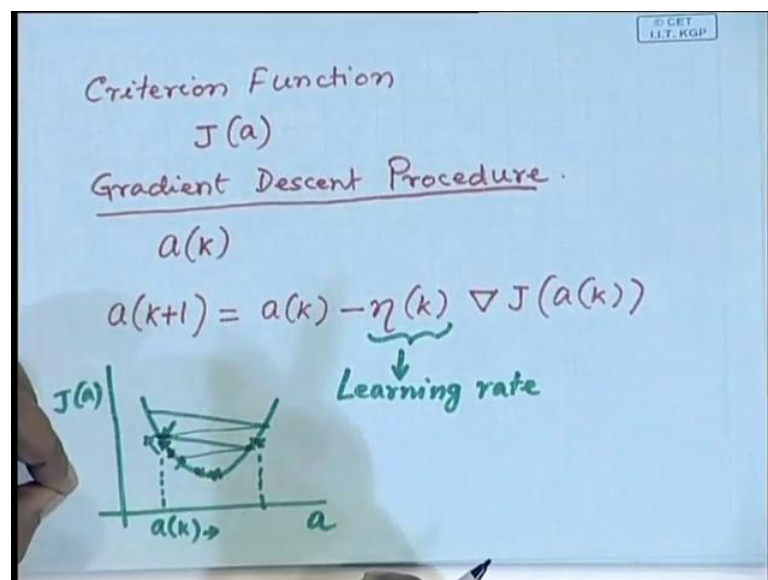
So, otherwise includes less than 0 as well as equal to 0 because equal to 0 does not give me any information. I cannot decide about the class belongingness of the sample. So, whether it is less than 0 or equal to 0 I will say it is miss classified. If it is greater than 0 I will say it is correctly classified. So, how we can do that? From here you said that if a transpose y_i is greater than 0 then y_i belongs to class ω_1 . If a transpose y_i is less than 0 then y_i belongs to class ω_2 .

So why do not I do one thing I take all the samples which are labeled as ω_2 and then negate it. So instead of considering y_i , I consider minus y_i , because I know all the training samples which are given from class ω_2 . So, I simply negate all those training samples. So, when I try to find out this feature vector, this weight vector a the samples belonging to class ω_1 , I will take them as it is, as it is means after augmentation after appending one.

And for the samples belonging to class omega 2, I will augment them by appending one and then take negative of it. So, if I take the negative then this a transpose y i which is supposed to be less than 0 now, it will be greater than 0. So, I get uniform decision criteria after negating all the samples belonging to class omega 2. In all the cases I should have a transpose y i greater than 0.

So, I will say that a transpose y i is greater than 0. My sample is correctly classified irrespective of whether the sample is taken from class omega 1 or the sample is taken from class omega 2. So, what I will do is I will take all the samples from class omega 1 as it is. The samples from class omega 2 I will take negative of it. And then use this for designing or to find out what will be the weight vector a.

(Refer Slide Time: 40:53)



Then again it is better if we can define some sort of criteria function. So, we take some sort of criterion function say $J a$ and the $g a$ has to be minimized. If a is the correct weight vector that means for that weight vector a which classifies all the training samples correctly, $J a$ will be minimum. So, I will define some sort of criterion function we will see what kind of criterion function we can have. And then my job will be, by varying a in an iterative way I have to minimize this $J a$.

And this $J a$ has to take a minimum value when I reach a weight vector which properly classifies all the training samples. And for minimization of this $J a$, the criteria function $j a$ we can make use of an approach which is called gradient descent approach or gradient

descent procedure. What is the gradient descent procedure? Suppose at the k th iteration I know what is a^k , a^k is the value of the vector a in the k th iteration. I have to update this weight vector a . So, from k th iteration I have to go to $k+1$ st iteration.

So, from here when I want to find out the value of a in the $k+1$ st iteration, that is a^{k+1} which will be equal to $a^k - \eta^k \text{grad of } J(a^k)$. This is what gradient descent procedure is? And what is this η^k , η^k actually specifies the learning rate. So, what does this procedure mean? Suppose, I have a criteria function something like this. This is the point where it is minimum. Suppose, in the k th instant whatever the value of a^k , I have the criteria function somewhere over here.

So, I put that this criteria function this is $J(a)$ with respect to a . So, at the k th iteration suppose, this is a^k and the criteria function value of the criteria function is $J(a^k)$. If I take the gradient of this, gradient will increase in this particular direction. So, what I am doing is, I am moving a^k in a direction which is negative to the gradient. That means I am moving a^k in this particular direction.

So, in $k+1$ st iteration depending upon the value of this η^k which is learning rate I may move either over here or over here or over here or even somewhere over here. So, if the value of η^k which is called learning rate, if the η^k is very large then when I modify this a^k , a^{k+1} may be found somewhere over here. That means I am overshooting the minimum value. And if η^k is very small I may land somewhere over here. So, which says that your landing rate is very slow.

If η^k is very large I may overshoot the solution and this may lead to oscillation. So, what I will have is from over here I will move to this point then to this point then to this point then to this point and so on. So, I will have oscillation around the solution vector. So, this is what this gradient procedure does. That you start with an initial arbitrary value of the weight vector, for that weight vector you find out what is the criteria value of the criteria function. Then try to minimize the criteria function following the gradient descent procedure or this is also called procedure of steepest descent. That means you follow the negative of the maximum gradient value path of the negative of the maximum gradient value. So, this is also called steepest descent procedure.

(Refer Slide Time: 46:55)

© CET
L.T.KGP

Perceptron Criterion

$$J_p(a) = \sum_{\forall y \text{ misclassified}} (-a^t y)$$
$$\nabla_a J_p(a) = \sum_{\forall y \text{ misclassified}} (-y)$$

$a(0) \rightarrow$ arbitrary.

$$a(k+1) = a(k) + \eta(k) \sum_{\forall y \text{ misclassified}} y$$

Now, one such criteria function as I said that, we will talk about the Perceptron criterion function one such criterion function is called Perceptron criterion. Now what is this Perceptron criterion? Let us assume that at the k th instant, somehow we have been able to obtain the weight vector a_k . I am not saying how we have obtained it? Somehow in the k th iteration I am about to find out the weight vector a_k , because it is in the k th iteration I say it is a_k .

Now, this a_k , I will try whether it can correctly classify all the training vectors or not, all the training samples or not. And when I consider all this training samples, as I said that the training samples from class ω_2 , they will be negated after augmentation. They will be negated training samples belonging to plus ω_1 they will simply be augmented they will not be negated so that if all the samples are correctly classified or all the samples I must have $a_k^T y$ greater than 0.

If there is any sample for which $a_k^T y$ is not greater than 0, it may be less than 0 it may be equal to 0 I say that particular training sample is not correctly classified. So, our aim will be to find out an weight vector a which will classify all the training samples correctly. I do not want to have even a single training sample which is not correctly classified or which is miss classified. So, I can try to design the criteria function which will make use of the training samples which are not correctly classified.

If the training sample is correctly classified, I do not bother about it because my job is done. If the sample is not correctly classified then I have to think to modify my weight vector. So, the criteria function that I want to design; I will make use of the training samples which are not correctly classified by that particular weight vector a^k . So, accordingly I can define the criterion function, I call J^p because we are saying it Perceptron criterion, which is equal to sum of minus $a^T y$ for all y , which are misclassified.

So, find that if y is misclassified then $a^T y$ it is negative. So, minus $a^T y$ has to be positive. So, as a result this Perceptron criterion function J^p , it can never have a negative value. It will always have a positive value, and the minimum value can be 0. So, I have a global minimum for this Perceptron criterion function. And this global minimum I can find out by gradient descent procedure. So, if I want to apply that gradient descent procedure then I have to take the gradient of J^p with respect to this weight vector a .

So, I will have grad of J^p with respect to which is nothing but sum of if I take the gradient of $a^T y$ with respect to a it simply becomes y . So, it is simply sum of minus y for all y misclassified. And my update rule or the algorithm to find out this weight vector a will simply be, I take an initial weight vector a^0 arbitrary. So, a^0 will be arbitrary then a^{k+1} that is the weight vector in the $k+1$ iteration will be simply $a^k + \eta_k$, which is the learning rate into summation of y , for all y misclassified.

So, this is the algorithm for design of my weight vector. So, once I have chosen a^0 arbitrarily. From a^0 I can find out what is a^1 by following this updation rule, a^1 from a^0 I can find out a^1 which is nothing but $a^0 + \eta_k$ which is the learning rate into summation of all the vectors y , which are actually misclassified by a^0 . So, I have a^1 , from a^1 , I can find out a^2 , a^2 is nothing but $a^1 + \eta_k$ plus summation of all the samples which are misclassified by a^1 .

So, by doing this iteratively I can finally, find out a weight vector which will finally, classify properly all the training samples. May be the number of iterations will be more depending upon your initial choice. And the number of feature vectors that you have the distribution of feature vectors, but if the feature vectors are linearly separable will get an

weight vector which will properly classify all the training samples. So, we will stop her today next day will continue with this.