Pattern Recognition and Applications

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

Lecture - 29
Support Vector Machine

Hello. In the last two classes, we have discussed about the radial basis function neural network. We have also discussed the about if I compared the radial basis function neural network with multilayer perceptron, what are the advantages and disadvantages of the radial basis function neural network with respect to the multilayer perceptron? So, we have said that in case of radial basis function neural network, the network consists of 3 different layers. One of the layer was which is present in all types of neural network that is the input layer, which basically accepts the input feature vector and followers that input feature vector to the layers above it.

In case of RBF neural network, we have 1 hidden layer and 1 output layer. The neurons in the output layer basically determine that what the class belongingness of the input feature vector is. The function of the neural of the neurons in the hidden layer is to compute our radial basis function of value and through this radial basis function, what we effectively do is we map the input feature vector from its original dimension to a higher dimensional space through a non-linear mapping or a non-linear transformation.

This non-linear mapping is actually done by the state of radial basis functions is. The purpose why we perform this transformation a non-linear transformation from a lower dimensional space to a higher dimensional space is that if you increase the dimensionality of the feature vectors, then the feature vectors belonging to different classes, if they are not linearly separable in a lower dimensional space, it becomes quite likely that they become linearly separable in higher dimensional space.

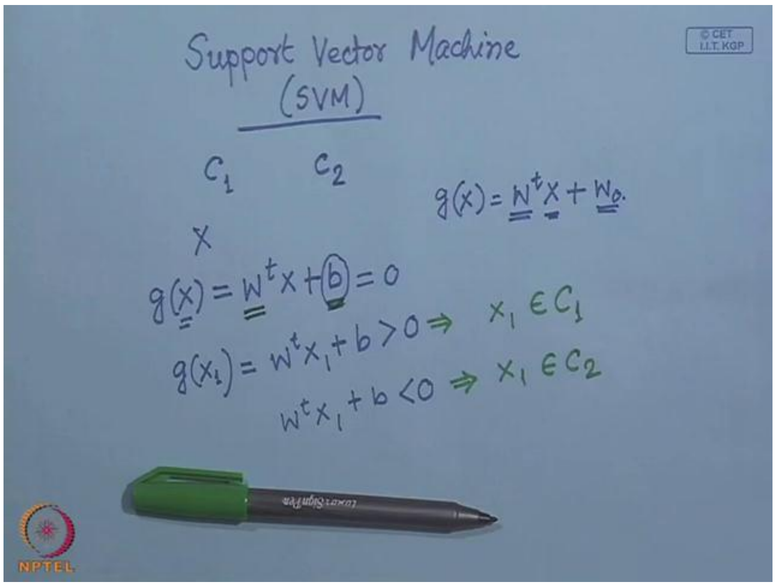
So, that was the basic motivation that we have in case of a radial basis function neural network. As by increasing the dimension of the input feature vectors, we are increasing the possibility of linear separability. So, the output layer of the radial basis function neural network simply computes linear combination of the outputs of the middle layer nodes. Based on this linear combination of the output of the middle layer nodes, it decides to which class the input feature vector x should be classified.

So, as a result, the output layer notes that we have been studying from the middle layer nodes to the output layer nodes. So, that section of the radial basis function neural network is effectively a linear classifier. So, that will have in case of radial basis function neural network is that I knew how many hidden layers I have to have. It is only 1. In case of multilayer perceptron, I cannot easily decide that how many hidden layers I should have.

Similarly, how many nodes are hidden layer that in case of multilayer perceptron that is also cannot be easily decided bias. In case of RBF network radial basis function neural network, I can set how many nodes in the region layer I need to have because it is simply the dimensionality of the spheres to which I want to cast my input feature vectors. The other advantage is the interpretation of the functions of the regular nodes, which is quite easy in case of RBF neural network. The interpretation of the functionality of the nodes in the hidden layers is not that clear in case of multilayer perceptron.

Also, the training in case of hidden layer is faster than the training in case of multilayer perceptron. The only disadvantage operating disadvantage of the RBF neural network is that the classification task in case of RBF neural network takes more time than the classification in case of multilayer perceptron. So, this is what we have discussed over last 2 classes. Today, we are going to discuss about another kind of classifier, which is called a support vector machine.

(Refer Slide Time: 05:26)



Support Vector Machine
(SVM)

C_1 C_2

X

$g(X) = \underline{\underline{W}}^t \underline{X} + \underline{\underline{w_0}}$.

$g(\underline{X}) = \underline{\underline{W}}^t X + \boxed{b} = 0$

$g(x_1) = W^t x_1 + b > 0 \Rightarrow x_1 \in C_1$

$W^t x_1 + b < 0 \Rightarrow x_1 \in C_2$

So, we will discuss today the classifier known as support vector machine or SVM. So, when we start our discussion on support vector machine, let us recapitulate one of the previous classifiers, the linear classifiers. We have discussed that is a linear discriminant function. So, during one of the early lectures, we have said that if we have an input feature vector x. We define a linear discriminant function g of x. So, for simplicity, let us consider that we are considering a 2 class problem.

We have 2 classes. One is class c 1, the other one is class c 2. We are talking about 2 class problems, class c 1 and class c 2. An unknown feature vector say x is to be classified as either belonging to class c 1 or belonging to class c 2. For doing this, when we talked about the linear discriminant function, we have defined a linear discriminant function say g of x, which was put in the form w transpose x plus b. Maybe in the early lectures, we had put g of x as w transpose x plus w naught. It was something like this or x is the input feature vector, w is the weight vector and the w naught is up past.

So, this is a linear function in a 2-dimensional space. If our feature vector is a 2-dimensional vector, then this linear equation represents a straight line. If our input feature vector is a 3-dimensional feature vector, then this linear equation, if I put this equal to 0, then I get a linear regression w transpose x plus b equal to 0. So, this linear equation in 2 dimensions represents a straight line. This linear equation in 3 dimensions represents a plane. If I increase the dimension or the dimensionality of the feature vector is more than 3, this linear equation actually represents what is called a hyper plane and w is nothing but a vector, which is perpendicular to that hyper plane.

So, this vector w it represents the orient orientation of the hyper plane in my d dimensional space where d is the dimensionality of the feature vector. Whereas, this term b or coming to the previous term w naught which is a constant it represents, what is the position of that hyper plane in my d dimensional space. So, the vector w represents the orientation of that hyper plane and the value d or w naught it represents the position of that hyper plane in my d dimensional space.

So, this b term b is usually known as a bias term, which is biasing the position of the hyper plane in the d dimensional space. Now, coming to our classification problem for every feature vector x, I want to compute this linear function w transpose x plus b. If this x lies on positive side of the hyper plane, then I will have g x. Let us take a specific vector say g x 1, this is a specific vector. The first vector, which will be equal to w transpose x 1 plus b, if this x 1 is on the positive side of the hyper plane, then I will have w transpose x 1 plus b because the question, which will be equal to 0 will be equal to greater than 0.

If x 1 belongs to negative side of the hyper plane, then I will have w transpose x 1 plus b, which will be less than 0. If x 1 lies on the hyper plane, then I will have w transpose x 1 plus b is equal to 0. So, this is a hyper plane, which defines my d dimensional space into 2 half spaces. In one of the half space, if I take a vector g of x for that particular vector x will be positive, if I take the vector x into the other half space, then g of x for that vector will be negative. I can have my classification rule. If w transpose x 1 plus b is greater than 0, then I infer that this x 1 will be classified to class c 1 that is it belongs to class c 1, where as if w transpose x 1 plus b is negative, then I can infer that this x 1 belongs to class c 2.

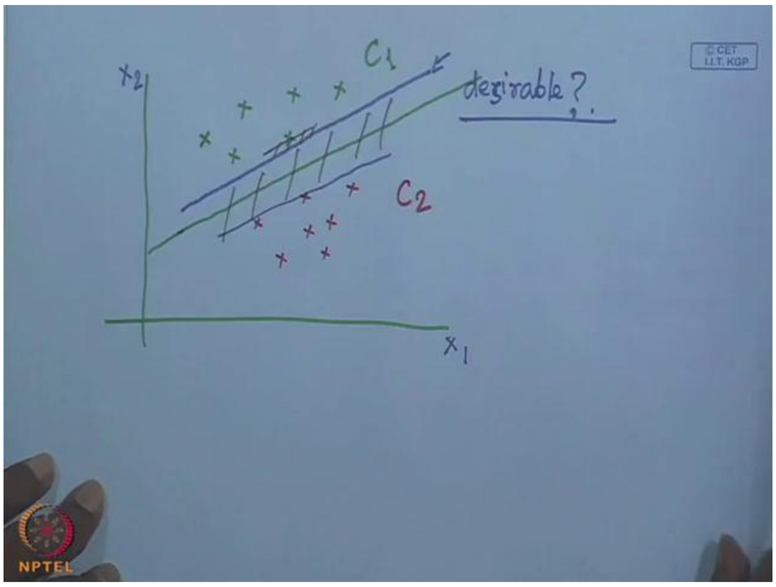
So, this is my classification rule. So, what is said is initially we had to train this classifier that means I have to find out what is this w vector w and what is this last term b, so that this g x with that w the trained value of w and the trained value of b can be used as a classifier. So, for training, we will have used supervised learning show. We have been given a large number of samples, some of the samples coming from class c 1 and some of the samples coming from class c 2.

So, using these training samples, when I try to train the network, then training of the network was done in an iterative fashion. So, for every training sample belonging to class c 1, we started with an initial value of w and p and for every training sample belonging to class c 1, we have tried to see that whether w transpose x plus b is greater than 0 or not because we have taken a sample from plus c 1. So, it has to be greater than 0.

If it is not greater than 0, then we have modified w and b in such a way that the position of the hyper plane or as well as the orientation of the hyper plane is so modified that that particular x, which is taken from plus c 1 is moved to positive side of this hyper plane. Similarly, if I take a vector from class c 2, we have checked whether this w transpose x where x is taken from class c 2 is negative or not. If it is not negative, then again we have modified w and p classes.

So, there again, we have to see that if this is negative or not if it is not negative, then we have modified w and b in such a way that that particular x is moved to negative side of the hyper plane. So, this is what we have turned in case of linear discriminator. We have also said there that if a sample is just beyond this hyper plane w transpose x 1 plus b is equal to 0, even then it might be correctly classified, but the classifier that I so obtain, so I can have a situation something like this.

(Refer Slide Time: 14:47)



I have 1 feature vector. Let us take 2 dimensional feature vectors something like this. So, these are the feature vectors, which say belong to class c 1. I can have a set of feature vectors something like this, which belong to class c 2. This is my x 1 dimension. If I have a 2-dimensional feature vector, this is my x 2 dimension. So, these are the 2 components of the feature vector. Now, if I have a classifier or a hyper plane into dimension, which is a straight line as we said earlier, it is something like this. These are the feature vectors, which are given for supervised planning.

So, if I have a situation something like this, even here you find that this straight line in d dimension. It is a hyper plane. It correctly classifies all these feature vectors, which are given for training of the classifier but, whether this type of classifier is desirable. Is it desirable? Obviously, a classifier of this nature is not desirable because this classifier gives a large bias in favor of plus c 2, whereas it puts a penalty against plus c 1. The reason is this inter space from here to here this is given to plus c 2. The margin to plus c

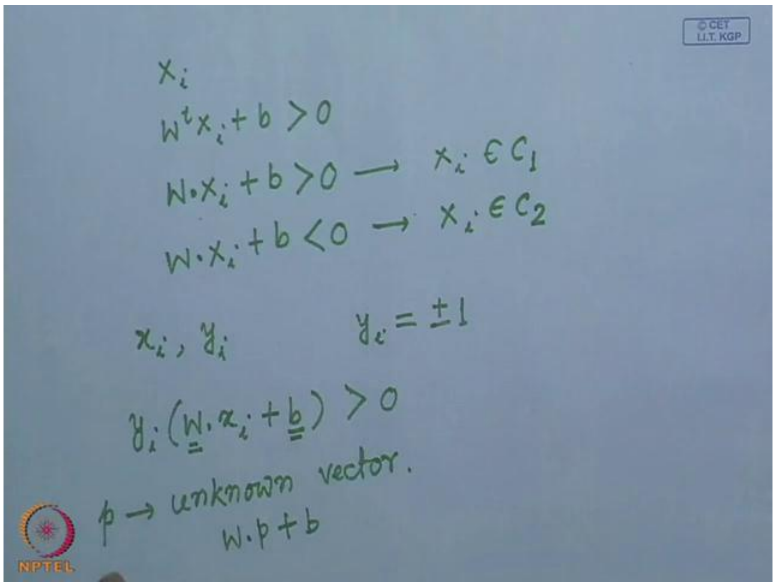
1 is only this much. Rather than this classifier, I would prefer to have a classifier somewhere over here.

So, the training vectors both from cross c 1 and cross c 2, both are equally apart from my classifier. The plane separating these 2 different classes and the support vector machine actually tries to find a classifier, which will be positioned in a form something like this. Now, let us see how such a support vector machine can actually be designed. So, the basic aim of the support vector machine is given.

If I am standing on one side of the boundary, if there is a possibility that I can cross the boundary on the other side, I will feel safer, if my distance from the boundary is larger. If my distance from the boundary is very small, so this is my boundary. I am standing somewhere over here, and then a little disturbance or a little noise can push me to the other side of the boundary in which case I will misclassify it. If I am standing somewhere over here where this is my boundary surface, then the margin that I have is quite large.

So, to push me to the other side of the boundary to make me misclassified, there has to be a large disturbance. So, I feel that I am safer if my distance from the boundary is very large. I am not safe if my distance from the boundary is very small. So, this is the kind of situation that I have. So, what the support vector machine does is the way of the support vector machine tries to design the classifier is that is it tries to maximize the distance of the separating boundary between the 2 classes by maximizing the distance of the separating plane from each of the feature vectors whether the feature vector belongs to class c 1 or the feature vector belongs to class c 2. When I talk about the support vectors, I will come to that point a bit later.

(Refer Slide Time: 19:36)



x_i

$W^t x_i + b > 0$

$W \cdot x_i + b > 0 \longrightarrow x_i \in C_1$

$W \cdot x_i + b < 0 \longrightarrow x_i \in C_2$

$x_i, y_i \qquad y_i = \pm 1$

$y_i(\underline{\underline{W}} \cdot x_i + \underline{\underline{b}}) > 0$

$p \rightarrow$ unknown vector.

$W \cdot p + b$

So, what I have is suppose that I have a vector x I, if this vector xi belongs to class c 1, then I will have w transpose x i. It has to be greater than 0. This w transpose x i, I can also put in the form of a dot product w w dot x i because this operation is same as this operation plus b that has to be greater than 0. If this xi belongs to class c 2, then I have the situation w transpose x i plus b have to be less than 0. If xi belongs to class c, so here w transpose xi plus b will be positive here, w transpose xi plus b will be negative.

Now, what I can do is when I go for designing of the classifier, I know to which of the class the sample xi belongs. I know whether x i belongs to class c 1 or x i belongs to class c 2 proceed so along with every x i. I can assign class belongingness. So, what I can put is along with every xi, I can also give a y I, where this y i can be either plus 1 or minus 1.

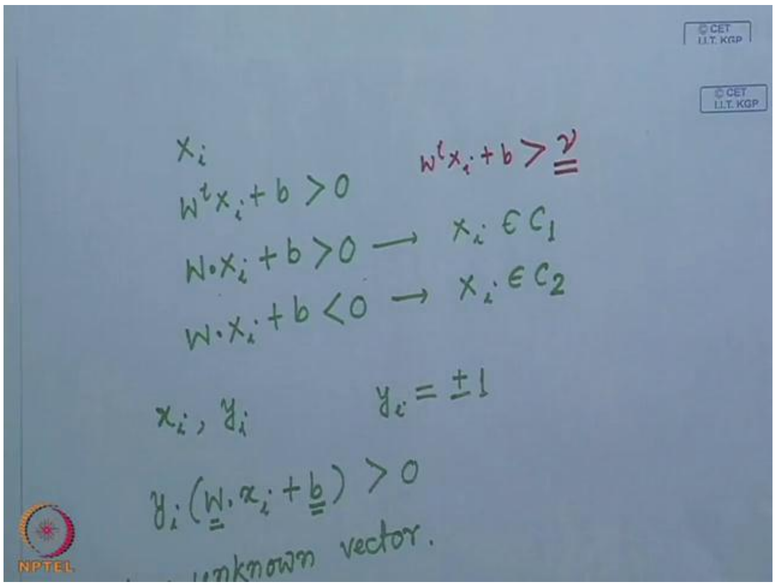
So, if xi belongs to plus c 1, the corresponding y i will be positive. If xi belongs to plus c 2, the corresponding y i will be negative. If I feed the data in this form, I compute y i into w transpose d w dot vi plus b, this will always be positive irrespective of whether x i belongs to plus c 1 or x i belongs to plus c 2.

This is because w transpose d w dot vi plus b y is positive. If x i belongs to plus c 2, then w dot xi plus b is positive y i is also positive, which will be specified. So, this product will be positive. If xi belongs to plus c 2, then w dot xi plus be will be negative, yi is also minus 1. So, look at the product of these 2. The product then will be positive.

So, this is what I will have. Using this concept, I can go for the designing of the classifier 1 and get w and b. Then for unknown feature vectors, say let us take an unknown feature vector p. This is an unknown vector, which has to be classified. I have to put this p into either c 1 or c 2 using that w and p that has been obtained after designing the classifier. So, once if do that, I do not have anyone yi because p is unknown. So, I simply compute w dot p plus b while w and p, they have already been decided during the training process.

So, if this w dot p plus b becomes greater than 0, I would classify this p to belong to plus c 1. If this is less than 0, I classify p to plus 0. So, this is the kind of situation that I have. As I said that it is of support vector machine, my aim is that I want to maximize that distance of the hyper plane of the separating boundary from each of the feature vector, so that every feature vector feels that they are safer so far as this classifier is concerned and the distance. We have made a modification in our classifier design, linear classifier design 1. We talked earlier that we consider the w transpose xi plus b to be greater than 0.

(Refer Slide Time: 24:17)



x_i

$w^t x_i + b > 0$ $\qquad$ $w^t x_i + b > \underline{\underline{\gamma}}$

$W \cdot x_i + b > 0 \longrightarrow x_i \in C_1$

$W \cdot x_i + b < 0 \longrightarrow x_i \in C_2$

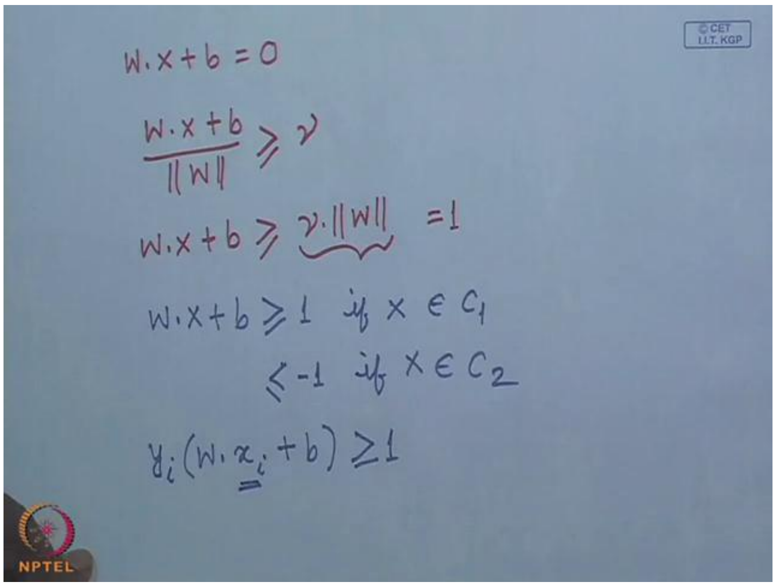
x_i, y_i $\qquad$ $y_i = \pm 1$

$y_i(\underline{\underline{W}} \cdot x_i + \underline{\underline{b}}) > 0$

unknown vector.

For correct classification, I want w transpose xi plus b should be greater than sum gamma by some margin. This margin is nothing but a measure of distance of xi from the separating plane.

(Refer Slide Time: 24:40)



$$W \cdot x + b = 0$$

$$\frac{W \cdot x + b}{\|W\|} \geq \gamma$$

$$W \cdot x + b \geq \underbrace{\gamma \cdot \|W\|} = 1$$

$$W \cdot x + b \geq 1 \quad \text{if } x \in C_1$$

$$\leq -1 \quad \text{if } x \in C_2$$

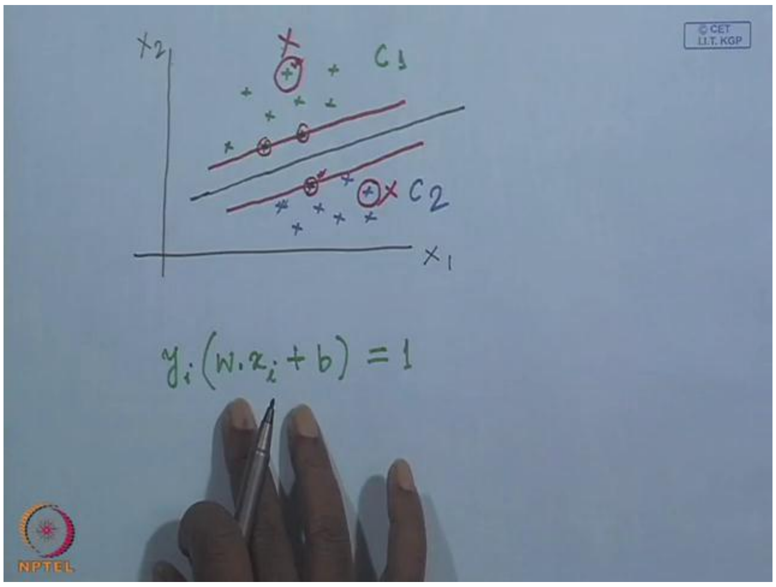
$$y_i(W \cdot \underline{x_i} + b) \geq 1$$

So, if I have a hyper plane whose equation is given by w dot x plus b is equal to 0, distance of point x from this hyper plane is simply given by w dot x plus b upon mod of w. So, this is known from school level geometry. So, this is the distance of x font x from this hyper plane w dot x plus b is equal to 0. What we want is we want that this has to be greater than or equal to some margin gamma sigma. My decision regarding the correct classification of the feature vectors is in this is independent of scaling vector w because vector w simply tells me that what is the orientation of the plane. So, whatever scaling factor I apply to w, my decision remains the same.

So, I can simply put that this w trans dot x plus b has to be greater than or equal to this into w. By proper scaling, I can set that this is equal to 1. I can set this term is equal to 1. This is simply a matter of scaling in every vector. I have to have a situation that w dot x plus b has to be greater than or equal to 1, if x belongs to c 1. It has to be made less than or equal to minus 1, if x belongs to plus c 2.

Now, for a vector x i, if I multiply by the corresponding y i during the training during the learning process or training process, I have a situation that y i into w dot x i plus b will always be greater than or equal to 1. This equality this y i into w i x i w w dot x i plus b will be equal to 1, if x i is a support vector. It will be greater than 1, if x i is not a support vector. So, then the question becomes that what is my support vector. Then I explain what my support vector is.

(Refer Slide Time: 28:38)



I simply go to my 2dimensional example. So, I have 2dimensional vectors x 1 and x 2. I have a set of feature vectors from say class c 1, which is like this. So, this is class c 1. I have a set of feature vectors coming from class c 2. So, I put it of this form. So, these are the feature vectors, which are from class c 2. Now, you can find that I can easily find out that I draw a straight line over here.

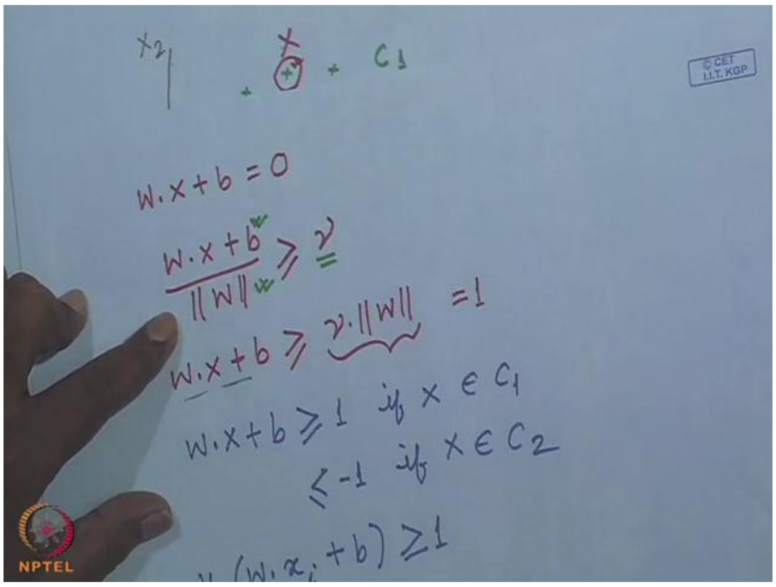
If I draw another straight line over here, these are the feature vectors from the training vectors. They are just on the boundary. So, any disturbance to the feature vectors will greatly affect this. If this feature vector is slightly disturbed, my classification result is not going to vary that much. So, I had to play put my classifier somewhere in the middle of this 2. This is a classifier, which will be more reliable. It is called more generalized. My risk of misclassification will be less.

You find that the position of this classifier of this hyper plane depends on the position of this feature vector. It depends on the position of this feature vector. It depends on the position of this feature vector. It does not depend upon the position of this vector feature vector even if I remove this feature vector from my training set. Even if I remove this feature vector from my training set, the position of the hyper plane remains the same.

If I remove this feature vector from my training set, the position of the hyper plane is going to be different. So, these are the feature vectors, which are called support vectors. The other vectors are not support vectors. So, this support vector machine is again a linear machine whose design is greatly influenced by the positions of the support vectors. Its position is not that much influenced by the vectors by the feature vectors, which are not support vectors.

So, that is why I said that from this that y i into w i dot x i plus b will be equal to 1 when this x i is a support vector. So, I have y i into w dot x i plus b, this will be equal to 1. If x i is a support vector, so this equality holds only for the support vectors. Now, what I have to do is if you look at this expression that what I want to do is this w dot xi plus b is a measure of distance stamps of point x i from the plain w dot x i plus b equal to 0. So, this is a factor, which has be and I want to maximize this. So, this is a factor, which has to be used for designing of my support vector machine.

(Refer Slide Time: 31:41)



x_2
C_1
© CET I.I.T. KGP
NPTEL

$$w.x + b = 0$$

$$\frac{w.x + b}{\|w\|} \geq \gamma$$

$$w.x + b \geq \underbrace{\gamma.\|w\|}\quad = 1$$

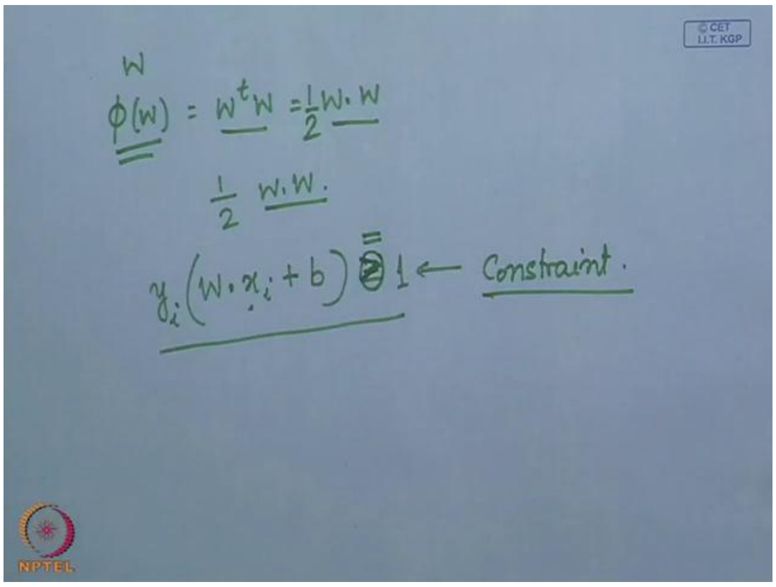
$$w.x + b \geq 1 \quad \text{if } x \in C_1$$

$$\leq -1 \quad \text{if } x \in C_2$$

$$y_i(w.x_i + b) \geq 1$$

I want that this distance or over this distance, the margin has to be as maximum as possible, which can be done from this expression by minimization of mod of w and at the same time by maximization of the wire speed. So, when I try to make the support vector machine, I will try to minimize this w. Simultaneously, I want to maximize this b. So, I want to minimize this w.

(Refer Slide Time: 32:17)



W

$$\underline{\underline{\phi(W)}} = \underline{W^t W} = \frac{1}{2}\underline{W \cdot W}$$

$$\frac{1}{2}\underline{W \cdot W}.$$

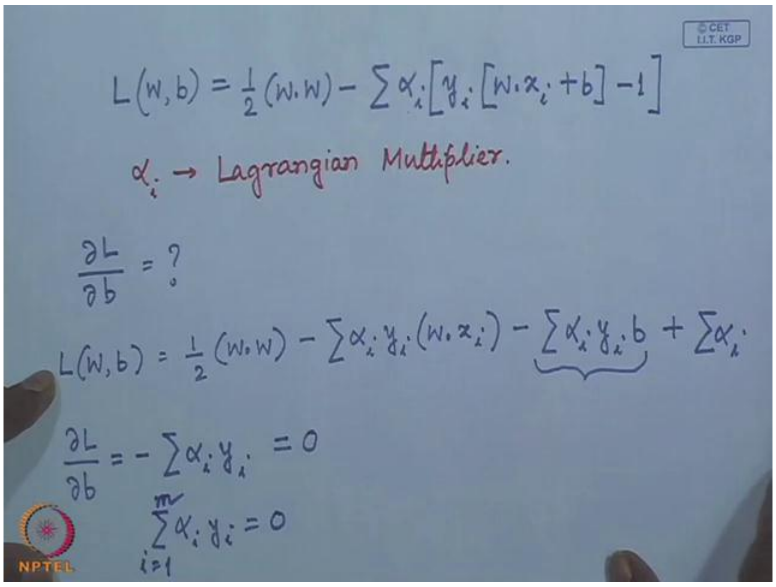
$$\underline{y_i(W \cdot x_i + b) \overset{=}{\geq} 1} \leftarrow \underline{\text{Constraint.}}$$

So, this is this w of the weight vector that I want to minimize. So, the minimization of w is same as if I want to minimize a function of w. This is nothing but w transpose w or w dot w. These 2 are same equations, positions for mathematical convenience which will be clear later on. I just put a term half. So, half of w dot w that I want to minimize and if I want to minimize this, the trivial value will obviously be w equal to 0, which is not the solution that solution that I am looking for.

So, for minimization of this, I have to look for the other constant. I have my constant is that 4 support vectors. I have to have w transpose dot x i plus b. This is into y i that have to be equal to 1. So, this is my constant. So, I want to minimize w or I want to minimize phi of w subject to the constraint that y i into w dot xi plus b has to be equal to 1. I say that this has to be greater than or equal to 1.

If x i is a support vector, then this has to be equal to 1 and because my support vector machine depends upon the support vectors. So, I will not take this inequality, rather I will take this equality that is y i into w dot xi plus b has be equal to 1. That is the constant and under this constant, I had to minimize my weight vector w. So, because it is a constrained optimization problem, this problem can become hearted to an unconstrained optimization problem by using the Lagrangian multiplier.

(Refer Slide Time: 34:36)



$$L(W,b) = \frac{1}{2}(W.W) - \sum \alpha_i [y_i [W.x_i + b] - 1]$$

$\alpha_i \rightarrow$ Lagrangian Multiplier.

$$\frac{\partial L}{\partial b} = ?$$

$$L(W,b) = \frac{1}{2}(W.W) - \sum \alpha_i y_i (W.x_i) - \underbrace{\sum \alpha_i y_i b} + \sum \alpha_i$$

$$\frac{\partial L}{\partial b} = -\sum \alpha_i y_i = 0$$

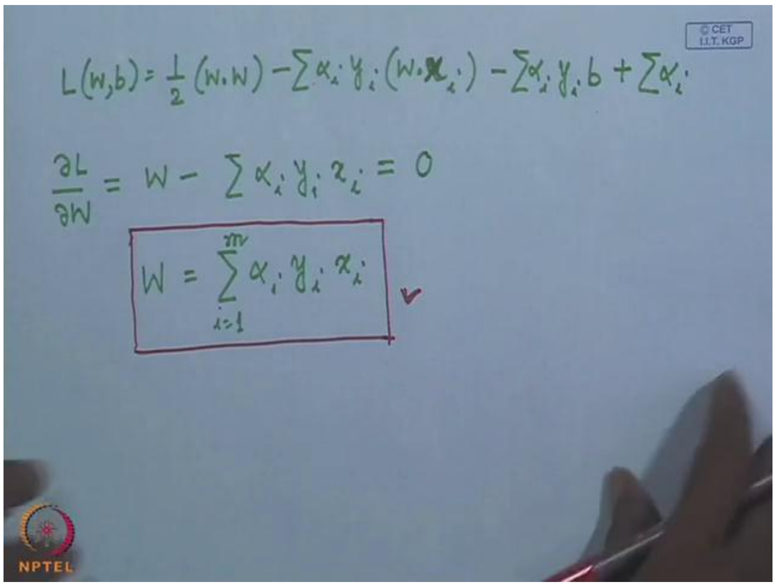
$$\sum_{i=1}^{m} \alpha_i y_i = 0$$

So, I can put Lagrangian multiplier. I can define a function of the form a w b because as I said that I want to minimize w. I want to maximize b, which will be half of w dot w minus sum of alpha i into y i into w dot x i plus b minus 1. I have to optimize this j that this Lagrangian and this i is the Lagrangian multiplier. So, when I go for optimization of this Lagrangian from my school level mathematics, we know that this optimization has to be done by taking the derivative of the Lagrangian with respect to w and by taking the derivative of the Lagrangian with respect to b.

So, if I take the derivative of this Lagrangian with respect to b, then what I have? So, what I want to compute is del l upon del b. This is what I want to compute. I have to equate this to 0. So, for doing this, let us try to expand this equation. Let us see what this expression is. So, in the expanded form, I will have l w b is equal to half of w dot w minus sum of alpha i into y i into w do x i minus sum of alpha i y i b plus sum of alpha i. So, this is the expression of the expanded form I have if I take the derivative of this with respect to b. So, I want to compute del l d del b. So, this del l del b obviously from here, you find that this is independent of l. This is independent of b. This is also independent of b.

So, only term that impacts b is this. So, del l del b will be simply minus alpha i y i and that has to be equal to 0. So, I get 1 constant that sum of alpha i y i that has to be equal to 0 for i ranging from 1 to say n, where n is the number of feature vectors, which are different for designing the classifier. So, this is one of the constraints that I have. What I do is I will take the derivative of this Lagrangian l with respect to my weight vector w. So, then this is the same Lagrangian. I put in the expanded form. I take the derivative of this.

(Refer Slide Time: 38:55)



$$L(w,b) = \frac{1}{2}(w.w) - \sum \alpha_i y_i (w.x_i) - \sum \alpha_i y_i b + \sum \alpha_i$$

$$\frac{\partial L}{\partial W} = W - \sum \alpha_i y_i x_i = 0$$

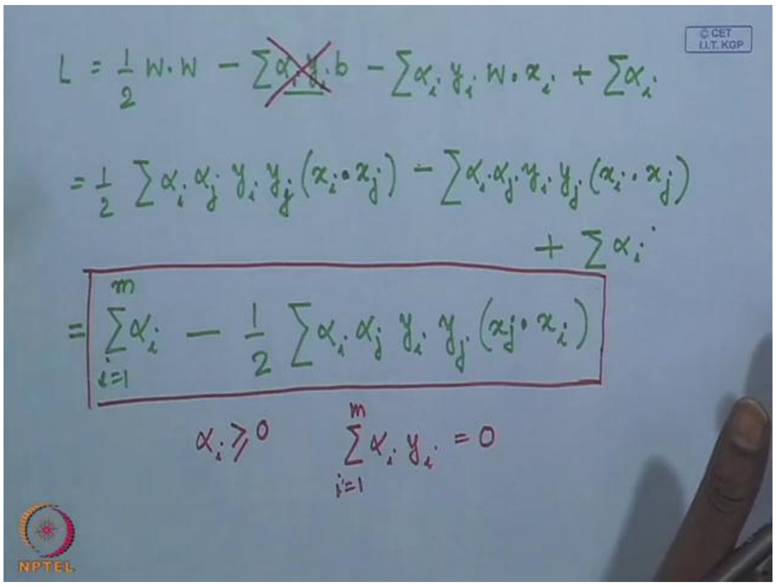
$$\boxed{W = \sum_{i=1}^{m} \alpha_i y_i x_i} \checkmark$$

Let me rewrite this Lagrangian. So, l of w b is equal to half of w trans dot w minus sum of alpha i y i w dot alpha x i minus sum of alpha i y i b plus sum of alpha each of these formations will from 1 to n before if I have n number of feature vectors provided for designing the classifier. Now, if I take the derivative of this Lagrangian with respect to w, so what I have is I have del l del w. So, when I take del l w del l w, this will simply be the derivative of this with respect to w is nothing but w and derivative of this with respect to w is sum of alpha i x i y i.

You find that this term is independent of w. This term is independent of w. So, derivative of these terms with respect to w will be equal to 0. Once I take this derivative, I have to take this equal to this derivative equal to 0, which gives me w is equal to sum of alpha i y i x i where this i will be from 1 to n as n is the number of training samples. So, through this derivative process, so this is the w that I have which is my weight vector. You find that this is not a dot product. So, this is why y i into x i x i is a vector alpha, i is a scalar, y i is also a scalar. So, this entire term is a vector.

So, for this optimization of the Lagrangian, I have got 2 equations. One is this sum of alpha i into y i for i is equal to 1 to m is equal to 0. I have this expression for my weight vector w where w is equal to alpha i y i x i where x i is the ith feature training feature vector for i ranging from 1 to m. So, these are the 2 expressions. Now, if I put this expression in my original Lagrangian, then let us see what we have. So, by putting this in my original Lagrangian expression as you said that the Lagrangian was in the expanded form.

(Refer Slide Time: 42:23)



$$L = \frac{1}{2} w \cdot w - \cancel{\sum \alpha_i y_i} \cdot b - \sum \alpha_i y_i w \cdot x_i + \sum \alpha_i$$

$$= \frac{1}{2} \sum \alpha_i \alpha_j y_i y_j (x_i \cdot x_j) - \sum \alpha_i \alpha_j y_i y_j (x_i \cdot x_j) + \sum \alpha_i$$

$$= \boxed{\sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum \alpha_i \alpha_j y_i y_j (x_j \cdot x_i)}$$

$$\alpha_i \geq 0 \qquad \sum_{i=1}^{m} \alpha_i y_i = 0$$

It was l is equal to half w dot w minus summation of alpha i y i b minus sum of alpha i y i w i xi plus sum of alpha i. So, in the expanded form, which we said earlier, this is half of w dot w minus alpha i y i w dot xi that is this term minus sum of alpha i y i b which is this term plus sum of alpha i. Now, over here, this alpha i y I, this term is equal to 0. So, this term simply gets cancelled. So, the expression I have is l is equal to half of w dot w minus this plus sums of alpha i.

This value of w is put into this expression. Then what I will have is this will be simply half of summation alpha i because I had the term w dot w dot product of the vector with itself. So, I had to have to have 2 different substrates over here, one i and the other one I will put as j. So, this will simply be alpha i alpha j y i y j into x i dot x j. So, this is a dot product minus this; will simply be again w dot x i and w is sum of alpha i y i x i.

So, this will simply be minus summation of alpha i alpha j y i y j into x i dot x j plus sum of alpha i. If I get this minus this, this term and this term they are same. So, this will be simply sum of alpha i for i varying from 1 to m minus half of sum of alpha i alpha j y i y j, this will be j into x i x j dot x i, which is same as x i dot x j. So, it does not matter.

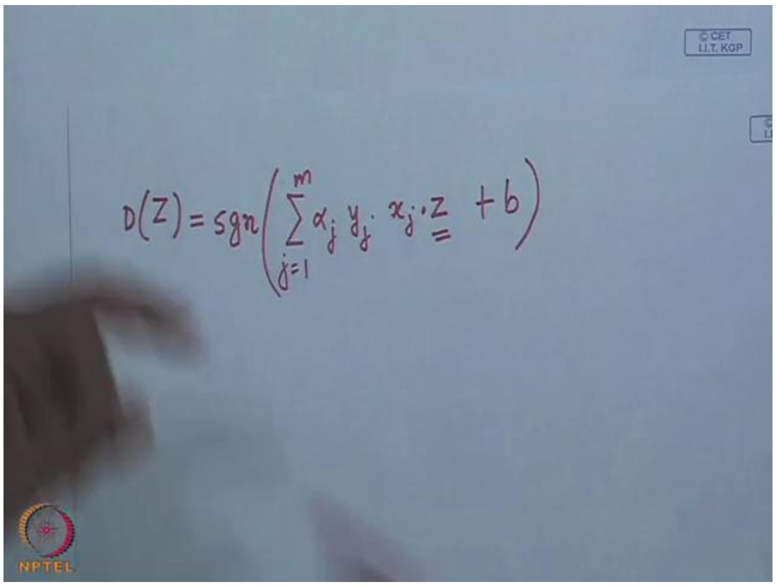
So, I have the Lagrangian expression, which is this. So, for design, what I have to do is I have to maximize this Lagrangian with different values of alpha, which are my Lagrangian multipliers and Lagrangian multipliers are always positive. So, I have 1 constant that alpha i always have to be greater than or equal to 0. The other constraints that we have are from here that sum of alpha i y i, i varying from 1 to m that has to be equal to 0. So, I have to find out such Lagrangian multipliers, which maximize this expression this Lagrangian expression.

When you try to maximize this Lagrangian expression, it is quite likely that some of the Lagrangians will be equal to 0. Some of the few of the Lagrangian multipliers will be equal to 0 and few of the Lagrangian multipliers value will be very high. So, if a Lagrangian multiplier and alpha i is equal to 0 that indicates that the corresponding training feature vector x i is not a support vector.

If a particular alpha i is very high for that indicates that the corresponding feature vector x i has a high influence over the position of my decision surface of the hyper plane. The other possibility is of course, there if I get an alpha i which is extraordinarily high, I can infer that the corresponding feature vector, which is given is a spurious point. It might have this. It is a disturbed point or it is an outlier.

All these different types of interpretations I can have over alpha i. So, obviously if alpha equal to 0, the corresponding x i is not a support vector. So, it does not influence the position of the hyper plane. So, what I have to do is I have to use these alpha i’s. I have to use with this alpha i’s. The value of w that I compute is given by this. This w goes into my decision making process. So, as a result, my classification decision will be like this.

(Refer Slide Time: 48:48)



$$D(\underline{\underline{z}}) = \mathrm{sgn}\left(\sum_{j=1}^{m} \alpha_j y_j \cdot x_j \cdot \underline{\underline{z}} + b\right)$$

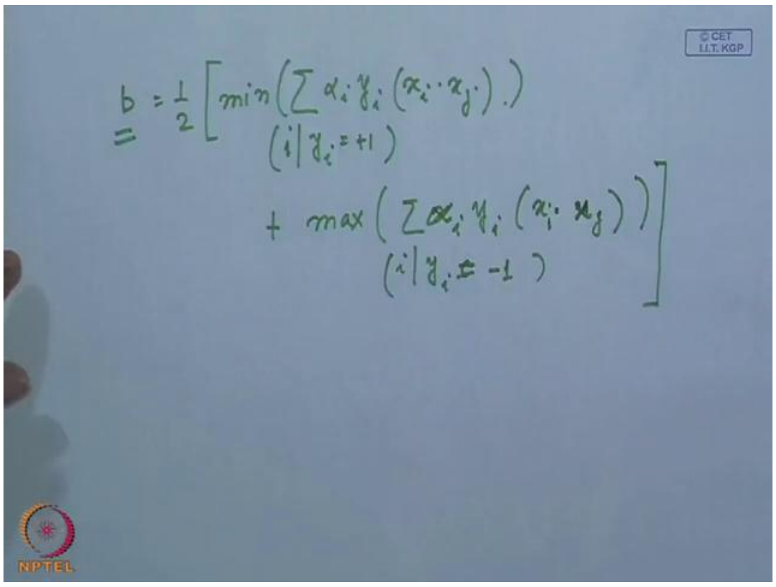
What I will compute is for an unknown z. If I have a feature vector z, which is unknown, the classification decision, if I said that the decision is d z, d z will be simply this. I have to compute alpha j y j into x j times z because you find that my w is nothing but alpha i y i x i perfect. What I have to compute is this w dot z. So, that is what I am doing alpha j simply, subscript i is replaced by subscript j does not matter.

So, I have to compute alpha j y j x j dot z. Take the summation over j is equal to 1 to m plus b and only the sign of this is important for classification. I do not want; I do not need what is of value of this function. I simply need the sign of this function. So, sign of this is important for classification. If the sign is positive, then this z will be classified to class c 1. If the sign is negative, then this z will be classified to class c 2. Now, if I write the steps of the support vector machine design, the steps of the support vector machine will be something like.

One more point that is I should mention over here as you are done in case of radial basis function, if the original set of feature vectors in lower dimensional space is not linearly separable, I cannot have the support vector machine. This is because it assumes that the samples are linearly separated. So, if they are not linearly separable in lower dimensional space, I had to cast these feature vectors into higher dimensional space by using functions like variables functions, which are functions. So, once I cast them into higher dimensional space, then by taking the samples in the higher dimensional space, I can try to design the support vector machine.

Then, again for classification, when we have classified this feature vector z before classification, I had to cast that into same higher dimensional space. After casting that into higher dimensional space, I have to compute this term in the higher dimensional space. Then I can compare the sign of this term. If the sign is positive, then it will be in plus class c 1. If it is negative, it will become to class c 2. So, I need 2 terms; one is w which has been obtained by optimization as this particular expression. The other one I need will be the value of b. So, the value of b can be computed.

(Refer Slide Time: 52:12)



$$b = \frac{1}{2}\left[\min_{(i|y_i=+1)}\left(\sum \alpha_i y_i (x_i \cdot x_j)\right) + \max_{(i|y_i=-1)}\left(\sum \alpha_i y_i (x_i \cdot x_j)\right)\right]$$

This is the margin. b will be half of minimum of sum of alpha i y i. I will put it as x i dot x j for of all i for which y i is equal to plus 1 plus maximum of sum of alpha i y i into x i x j for all i where y i is equal to minus 1. So, that is how I get the value of the b and this w.

This w and t, this b goes in this expression for classification of an unknown feature vector set. So, this is how our support vector machine work. As I said before, the support vector machine is again nothing but a linear machine. The only thing is it helps in placing the separating boundary between the 2 crosses, classes or it simply states where my hyper plane should be based. So, the classifier that I get that is more robust or more generalized.

Now, instead of 2 class problem, if I have a multiclass problem, then I have to have multiple numbers of support vector machines to support. So, I have to have 1 support vector machine, which tells me whether a sample belongs to plus class c 1, whether it does not belong to class c 1. Then I have to have a support vector machine, which tells me whether a sample belongs to plus class c 2 or it does not belong to class c 2.

So, I have multiple numbers of support vector machines for a multiclass problem. It can be shown that if I have n number of classes, then what I need is n minus 1 number of support vector machines conditions. I have to have n minus 1 number of linear classifiers. So, with this, I stop here today.

Thank you very much.