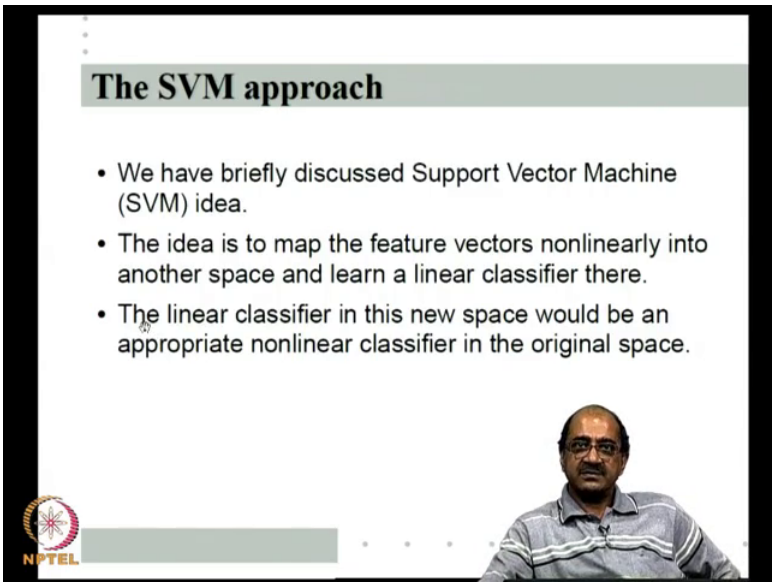
Pattern Recognition
Prof. P. S. Sastry
Department of Electronics and Communication Engineering
Indian Institute of Science, Bangalore

Lecture - 31
Support Vector Machines – Introduction,
obtaining the optimal hyperplane

Hello and welcome to this next lecture on pattern recognition course. We have been looking at essentially neural network model for the last few classes. We looked at feed forward networks with sigmoidal activation function as well as real based function networks. Learning algorithms for using them, both of them are useful models for learning non linear classifiers and regression functions, right? Then end of last class we said that we will move on to a next approach for learning non linear classifiers, where instead of wanting a general parameterized class of non linear classifiers. We are looking at learning a linear classifier in a transformed space, right? That is the so called support vector approach. So, that is what we start with today the support vector machine approach, right?

(Refer Slide Time: 01:08)

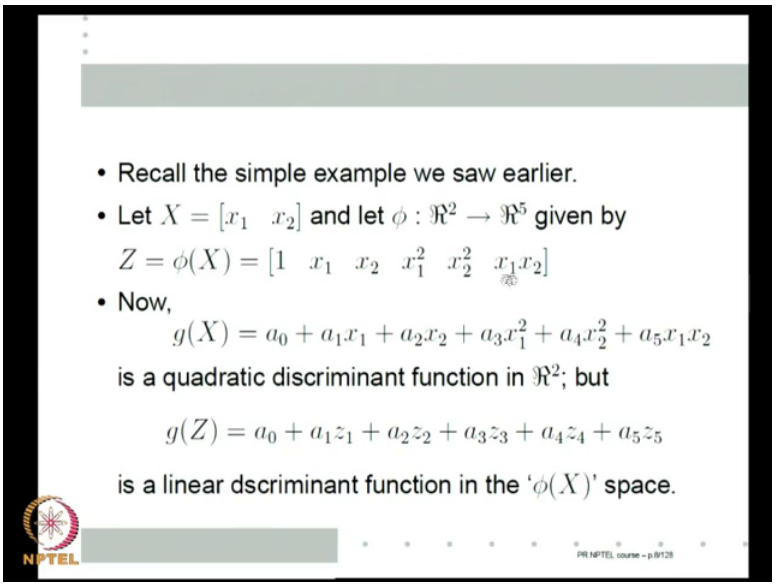


We have briefly discussed it last class, so the basic idea is the following. You from the original feature space, you non linearly map the feature vectors into some other space. Normally a higher dimensional space, so not necessarily, so you map into a different space. In the new space learn a linear classifier. Now, this works because the linear

classifier in the new space could be an appropriate non linear classifier in the original space.

If you do the mapping properly if we choose the right mapping, then the linear classifier in the new space would be a nice non linear classifier in the original space, right? As an old Bob Dylan song refrain use to have one man sailing is another man's floor, so what is linear depends on which space you are in? So, what is linear in one space could be non-linear in another space.

(Refer Slide Time: 02:14)



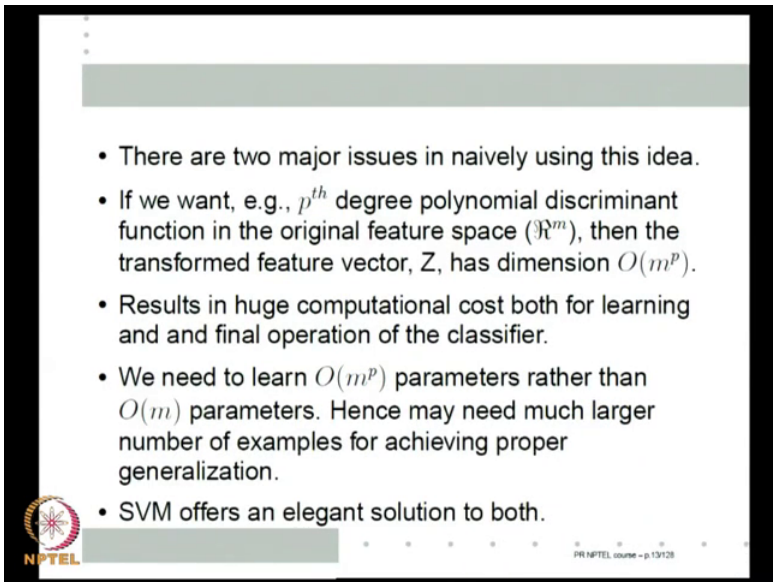
So, we shown an example of this last class. Let us recall it it is a two dimensional. Example, suppose those will feature space is two dimensional and let us say x 1, x 2 are the two feature components. Now, let us define a function a non linear function that maps r 2 to r 5; that means you are transforming the two dimensional feature space into a five dimensional feature space. So, it is a function phi that maps r 2 to r 5 given by given any x, the image of x phi x which we called by z is given by 1 x 1 x 2 x 1 square x 2 square x 1 x 2, it is a five component vector, right? Because it is an r 5, I am sorry its actually six component vector, it should have been r 6. I am sorry but, it is not r 5, but it is r 6. Now, the main idea is that if I look at a function of x, the two dimensional vector x which is a 0, a 1 x 1, a 2 x 2, a 3 x 1 square a, a 4 x 2 square a 5 x 1 x 2.

Then this is a quadratic discriminant function in the original two dimensional space, because it has quadratic terms in the feature components. On the other hand, if I take a

function of the vector z, let us call the components of z as you know this is z 0, z 1, z 2, z 3, z 4, z 5, right? So, essentially because of this one it is like an augmented feature vector in r 5, the rest of the feature vectors is r 5, but I am doing an augmentation, so we will call this as z 0.

Then z 1, z 2, z 3, z 4, z 5, then if I read a g z as a 0 plus a 1 z 1 plus a 2 z 2 plus a 3 z 3 plus a 4 z 4 plus a 5 z 5, this is a linear discriminant function in the phi x space, right? So, what is linear in terms of the components of z would be non linear in terms of the components of x. This is the basic idea. So, if I can find the right transformation, can use techniques for learning linear models for learning non linear classifiers in the original space.

(Refer Slide Time: 04:23)



Of course, there are as we mentioned if I just naively try this idea, there are two major problems, right? By naively try I mean simply think of a function phi transform all x's into z's and then on use any linear method that will act. What is the problem? As we can see from the previous example, when I wanted quadratic, I need all quadratic terms in the in the components of z, right? So, I need x one square x two square x one x two and so on. So, essentially if I need a p th degree polynomial at the discriminant function in the original space.

Then in the in the transformed vector I should have terms that contain all p th degree terms in components of x, which means they'll be order m to the power p terms in the

vector z. So, z would have very huge dimension. So, even in the quadratic case I have taken 2, but if I have got let us say a 100 dimensional vector which is not very uncommon pattern recognition, then I will have of the order of 100 square components in z; that is 10,000. On the other hand if I want a cubic discriminant function, discriminant function up to third degree polynomial, then I would not have 10 power 4, but, I would have 10 power 6 components in z a billion components in z instead of being a 100 components in z.

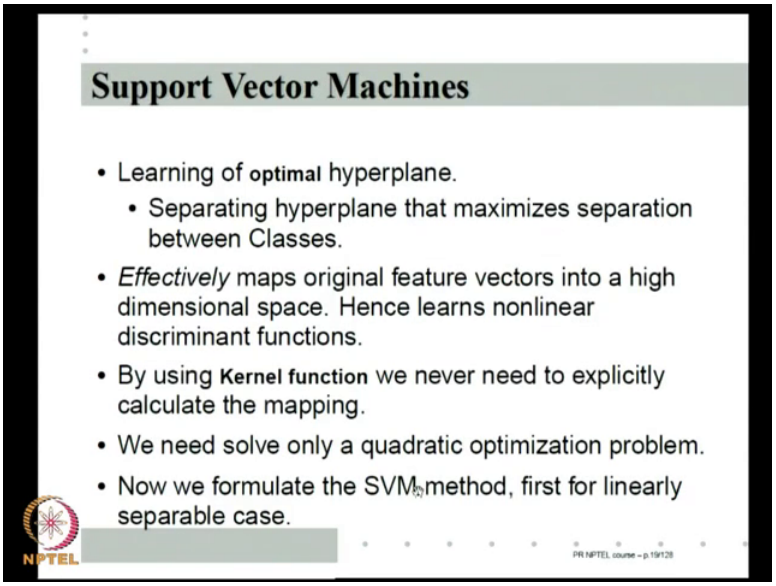
So, the the dimension of z can be very huge even for second and third degree polynomial discriminant functions that we want to learn. They are not particularly non linear just quadratic or cubic functions in the original feature space. The transform space can have huge dimension and this can results in huge computational cost. Let us say what do I have to do when to classify? I have to calculate w transpose x, if x is a feature vector for any linear classifier I essentially calculate something like w transpose x in the original space. It will be in only 100 multiplications, if it is a 100 diameter space. But if you are learning a cubic classifier a a cubic discriminant function it will be a million component z, so it will mean a million multiplication.

Every time I do a simple w transpose x operation it is a million multiplications, right? So, learning the classifier as well as the final operation of the classifier can be very costly. Now, during learning I might have to store, let us say a 1000 100 dimensional vectors, instead of that now to store a 1000 million dimensional vector, right? That is that is a that is a you know a 10,000 fold increase in the memory needed. So, it it it incurs large computational cost not only that to learn the linear classifier in this new space. I have to learn order m power p parameters, w will be the same order as the feature vector.

So, if I am learning a linear classifier in the jet space the w in that linear classifier should have the same dimension as the vector z. So, that means I have to learn order m power p parameters. If I am learning a p th degree polynomial in the original space. If I am learning a linear in the original space, I will learn only order m because instead of learning a 100 parameters for a hyperplane in in the 100 dimensional feature space, I am learning a million parameters, because now my feature space has become a million dimensional. Now, to learn 100 parameters based on v c dimension hyperplanes, v c dimensional hyperplanes is of the order of the dimension of the space is actually dimension of the space plus 1.

So, in the 100 dimensional space the family of hyperplanes should have v c dimension of 100, so maybe I need 1000 examples here. Because this space has dimension 1 million I may need 10 million examples. 10 million examples are huge, we cannot get them is even getting 1000 examples may be difficult, but 10 million examples is much more difficult, right? Now, this is also a a a a a an extra problem, so basically both in terms of computation and in terms of being able to learn well with the available examples. This this new idea will not work. So, the nice thing about what we call support vector machines is that they offer an elegant solution for both these problems, so let us look at what these are?

(Refer Slide Time: 09:12)



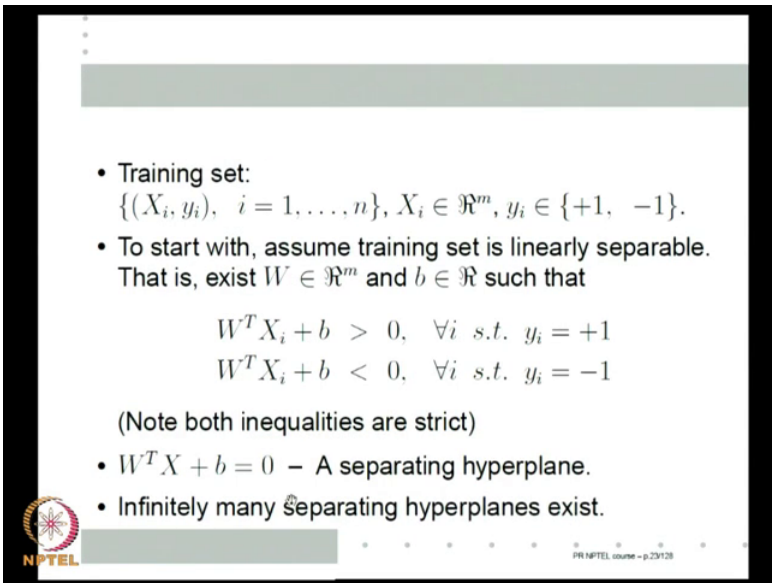
They essentially learn not any hyperplane, but what we term optimal hyperplane? An optimal hyperplane is a separating hyperplane that maximizes separation between classes, we will formally define it in a couple of minutes. Now, learning optimal hyperplane is essentially what allows SVM's to learn well with fewer examples even in a very large dimensional space.

Then they only effectively map the original feature vectors into high dimensional space. Of course, they learn a non linear discriminant function by learning a linear discriminant or or linear classifier in a high dimensional space, but this is done using what is known as a Kernel function, whereby we never actually explicitly do this mapping. We never calculate phi of x, we never work in the range space of phi, we shall see why, how it is

feasible? By using Kernel functions, we never actually explicitly calculate this mapping. So, we only effectively find a linear classifier in the high dimensional space.

So, ultimately as it turns out for the sake of solving a quadratic optimization problem we will we will look at the dimension of the quadratic optimization. It turns out to be dimension of the number of examples rather than the dimension of the feature vector. But we solve only a quadratic optimization problem and are able to do this mapping into high dimensional space and learning a linear classifier there. So, with that introduction, let us move on to formulating the SVM method. First we look at formulating it for the linearly separable case and then we look at the more general case, okay?

(Refer Slide Time: 10:55)



So, this is the training set given to us we will revert back to our old training set notation. Remember when we are considering all linear models we arre using x's for feature vector x or and y's for targets or classes and small n as the number of examples. So, earlier when we considered neural networks because y is happen to be outputs of the network nodes, we used d i as the desired output in the training set. Now, that we do not have that problem we will revert back to the originals notation we have.

So, x i's are the inputs and y i's are the targets of the class labels. So, we are given n examples x i are in m dimensional input space. So, we will take m to be the dimension of the feature vector space and we choose class labels for here to be plus 1 minus 1 earlier when we considered linear classifiers, we mostly chosen 0 and 1 as the two class labels.

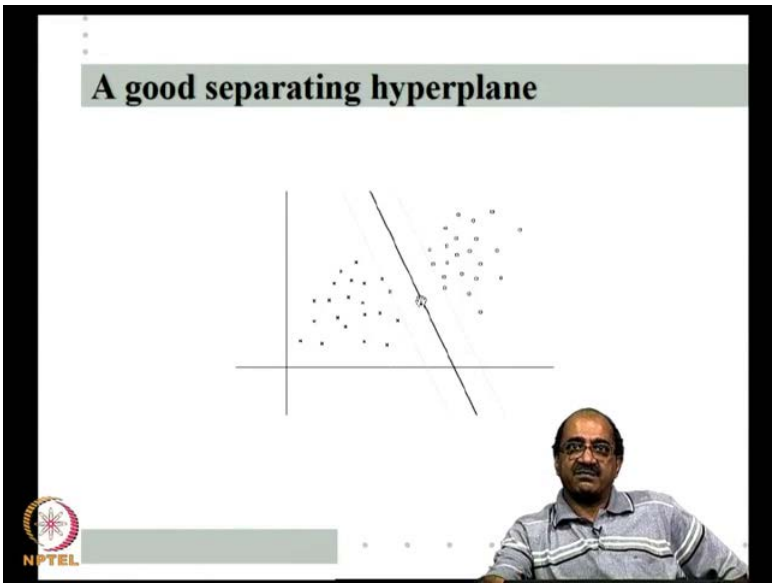
But we will take plus 1 and minus 1, we are only going to look at two class problem. Multiclass will be solved much like in in the linear models by using you know one of the many techniques as we discussed there.

So, we will consider only the two class problem, the two class will be taken as plus 1 minus 1. To start with we're assuming that the training set is linearly separable. That means there is a weight vector w belonging to r m and a constant b belonging to r s. Of course, there are such that whenever y i is plus 1; that means x i belongs to the class plus one w transpose x i plus b is greater than 0. Whenever y i is minus 1, we have w transpose x i plus b less than 0, so for all class one patterns w transpose x i plus b greater than 0. All class minus 1 patterns w transpose x i plus b less than 0.

Because I have only finitely many patterns, if there are separable. They will be exactly separable, right? So, no pattern need be on the separating hyperplane, so we will take both inequalities to be strict. Now, of course, we know we have already seen when we considered perceptron that such a thing is called a separating hyperplane, such as w transpose x plus b is equal to 0 is the equation of a separating hyperplane as we seen when we discussed perceptron. If there is one separating hyperplane, if the if the training data is linearly separable, then there are infinitely many separating hyperplanes, right?

So, the question comes which is a good separating hyperplane? Earlier say for example, when we did perceptron all we are bothered is the about finding any separating hyperplane; that is all we actually proved about perceptron that it can find a separating hyperplane. But because we know that exist infinitely many separating hyperplanes, can we say that some hyperplanes have better than others?

(Refer Slide Time: 13:37)

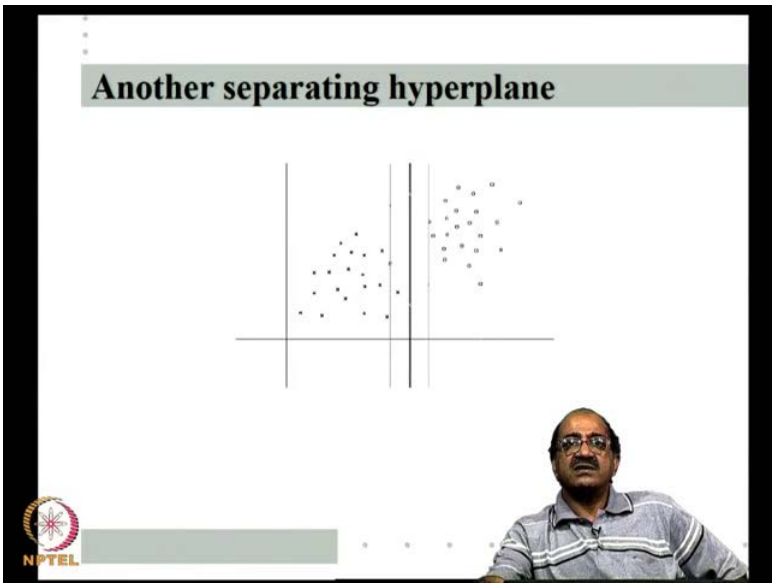


So, here is one way of looking at it look at that dark line in the center for now forget about the two faint lines on either side. Let us say for this problem we would like to say that this is a good separating hyperplane, whereas this is not so good hyperplane. Why do we say, why do why why do you want to say that? Essentially looking at this spread the classes seem to be spread like this, so this line seem to be exactly you know bisect the main line along which classes are spread, right?

But that is intuitive that is more difficult to capture, but another way of seeing it is you can see the nearest patterns on either side of the hyperplane are so far apart for this hyperplane. Whereas, for this hyperplane the nearest patterns are much closer to the hyperplane, so this hyperplane is making more commitment about unseen parts of the space compare to this hyperplane. This hyperplane is keeping itself as forever from both side patterns as possible. Hence, getting lot of what I may call marginal separation, what we will actually ultimately call margin of separation, right?

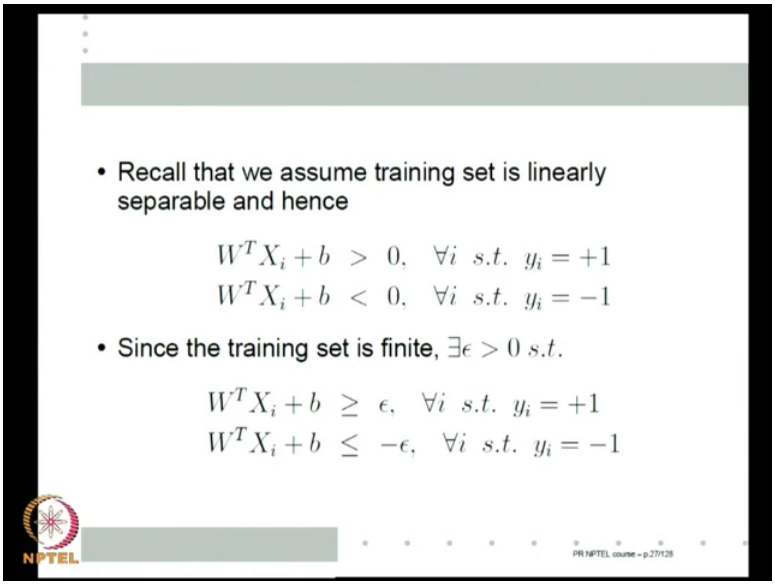
So, on either side of the hyperplanes that I am finding the the patterns are sufficiently far away. That gives me some intuitive feel that this might be a better hyperplane. So, this is essentially what is one intuitive way of thinking good hyperplane as it turns out even more formal is a good separating hyperplane. If the closest pattern to the hyperplane on an either side is far away for one hyperplane, then that hyperplane is better than another hyperplane for which it is closer.

(Refer Slide Time: 15:20)



As here it is too close, so this this is not as good as separating hyperplane as that one.

(Refer Slide Time: 15:30)



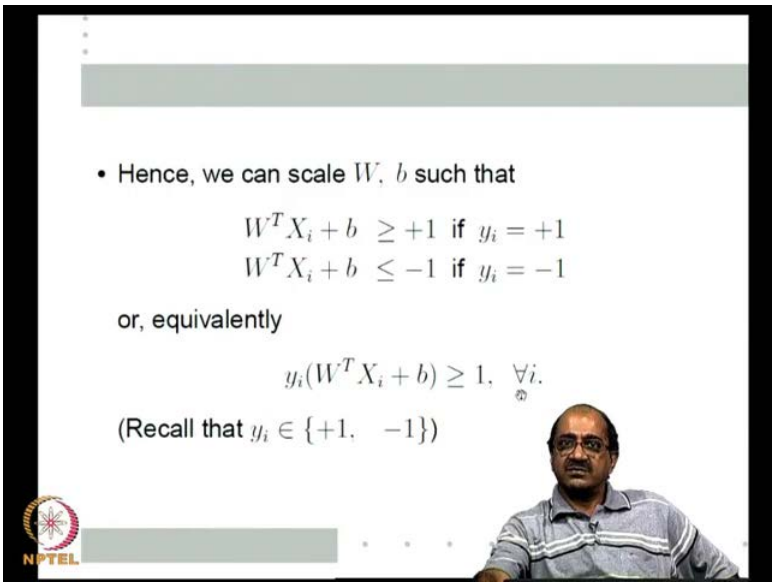
So, let us now try to formalize this notion that the closest pattern should be far away. Recall that the training set is linearly separable and hence we have this W b. Now, because the training set is finite and for each of the class plus one pattern is strictly greater than 0, each of the class minus 1 pattern is strictly less than 0. So, if I take minimum of this over all class one patterns, it will be some epsilon 1, which is greater than 0 strictly. If you take maximum of this over all class minus one patterns, this some

minus epsilon 2, where epsilon 2 is positive.

If I take smaller of the epsilon 1 epsilon 2 is epsilons, I can always satisfy w transpose x i plus b is greater than or equal to epsilon. If x i is class plus 1 less than or equal to minus epsilon, if it is minus 1. Because this they are only finitely many patterns and if they all if w transpose x i plus b is positive for all of them, there must be a smallest number for it. We have will calling that epsilon 1. Similarly the largest number for this we are calling that minus epsilon 2 and then we are taking smaller of epsilon 1 or epsilon 2.

So, that we can satisfy this, once we can satisfy this, now I can now divide the center equation by epsilon. Because it is a positive number, right? Dividing the the inequality by epsilon simply means, I am scaling W on b. Now, if W transpose x plus b is equal to 0 a separating hyperplane, then k w transpose x plus k b is also a separating hyperplane. So, k W plus k b will also be a separating hyperplane.

(Refer Slide Time: 17:11)

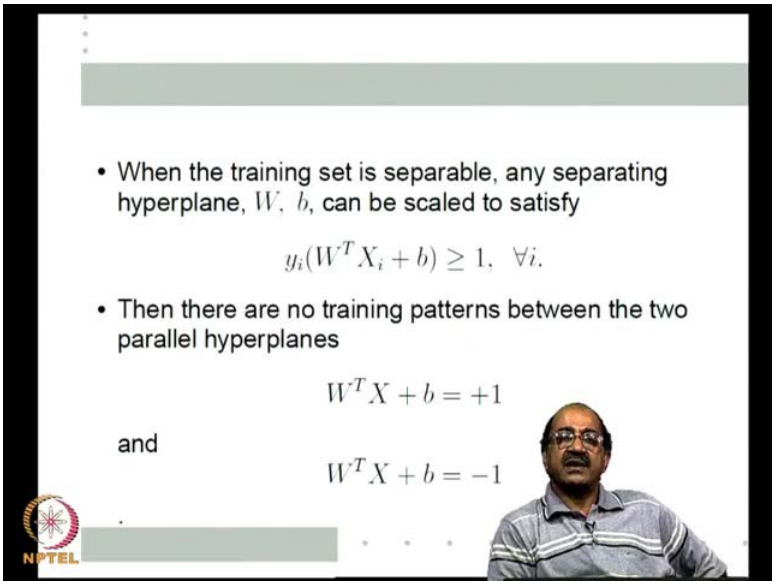


So, I just I can I can scale the W on b to satisfy W transpose x i plus b is greater than plus 1, if x i is in class plus 1 is less than minus 1. If x i is in class minus 1, what is this mean? Now, we can rewrite this as because y i which is chosen to be plus 1 minus 1. I can write both these inequalities together compactly as y i into w transpose x i plus b greater than equal to 1.

Actually to be able to write like this and hence simplify our algebra is why we have

taken y i to be plus 1 minus 1 in this case. So, we can always scale any separating hyperplane, so that it will satisfy y i into w transpose x i plus b is greater than equal to 1. What does it mean?

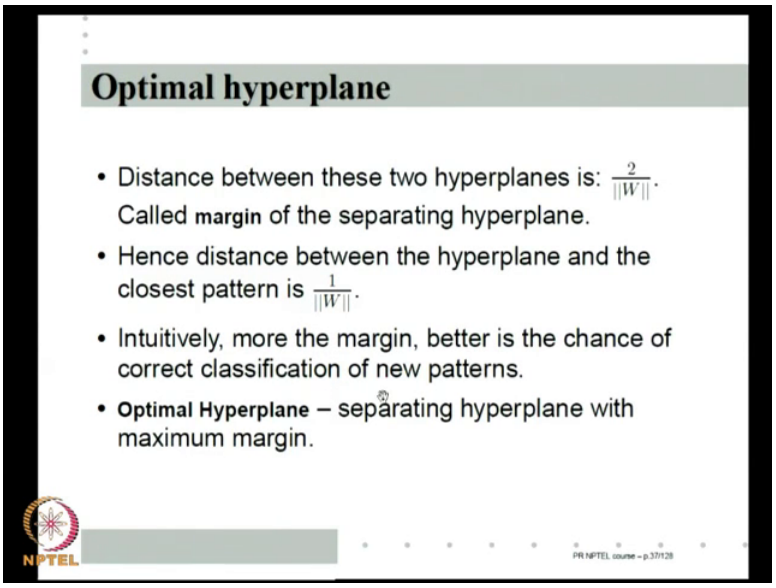
(Refer Slide Time: 17:53)



When training set is separable, any separating hyperplane W b can be scaled. That is we can always take a W b to satisfy y i into w transpose x i plus b greater than or equal to 1. What is that mean? If x i is class plus 1, then w transpose x i plus b is greater than plus 1. If it is class minus 1, w transpose x i plus b is less than minus 1, which means for no training pattern W transpose x i plus b is between plus 1 and minus 1, which is same as saying there is no training patterns between the two parallel hyperplanes.

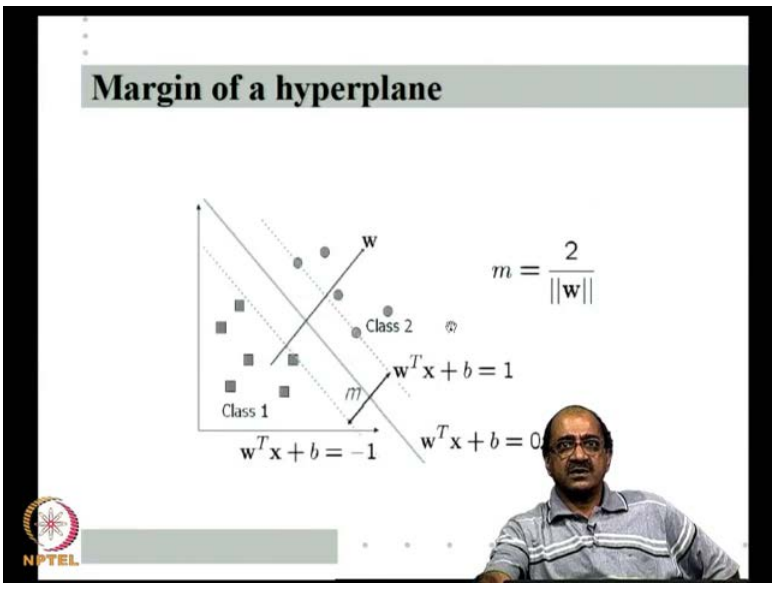
W transpose x plus b is equal to plus 1 W transpose x plus b is equal to minus 2. W transpose x plus b is equal to 0 is my hyperplane. But for all class one pattern W transpose x plus b is greater than 1 greater than or equal to 1. All class minus one patterns W transpose x plus b is less than or equal to minus 1, which means if I take the two parallel hyperplanes W transpose x plus b is equal to plus 1 and W transpose x plus b is equal to minus 1, there will be no training pattern between them.

(Refer Slide Time: 18:55)



Now, the distance between these two hyperplanes will be 2 by norm W. Norm W is W transpose W square, norm W square is W transpose W. This is a simple coordinate geometry problem i am sure u can easily do this.

(Refer Slide Time: 19:08)



So, let us look at what this means see this is the W transpose x plus b is equal to 0 hyperplane. So, W transpose x equal to plus 1 and minus 1 are on either side of this hyperplane, because there they will be parallel. They will be just the hyperplane moved parallel to itself on either side. So, this one will be W transpose x plus b is equal to plus 1

other will be W transpose x plus (( )) equal to minus 1. What is showed as m here that is the margin. Simple coordinate geometry will show you that this margin is 2 by norm W, right? So, actually W is as you know a hyperplane, because hyperplane equation is W transpose x plus b is equal to 0, W is a normal vector to the hyperplane, right?

So, that that is why how we shown the W there. Because the distance between the two hyperplanes is 2 by W distance between the hyperplane on the closest pattern is 1 by W. Because we we are only guaranteeing that between W transpose x plus b is equal to plus 1 and is equal to minus 1. I will not be able to they will not be any training pattern of saying W b can be just can be scaled up to this cannot be scaled beyond this. If I scale beyond this, it will go out. So, ultimately the closest patterns will be either on W transpose x plus b is equal to plus 1 or W transpose x plus b is equal to minus 1 as you, as as in this figure.

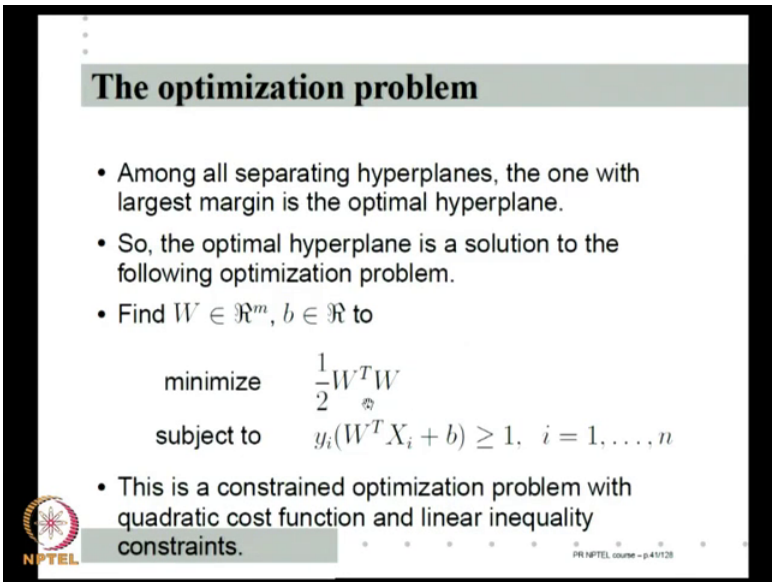
So, if I take this orientation of the hyperplane, then this is the separation I can get. So, this will be symmetric on either side. So, the distance between hyperplane and the closest pattern will be one by instead of 2 by norm W. It will be 1 by norm. See in general, because both sides the patterns need not have to be symmetric, one side patterns may not actually be on this hyperplane by the time I move this hyperplane this much. I may have it, this patterns on this side the patterns may be far away, but I cannot hit any more because these two hyperplanes have to be W transpose x plus b is equal to plus 1 and W transpose x plus b is equal to minus 1.

So, ultimately if margin is 2 by W, I know at a distance one by norm W, there is a pattern and no pattern is closer than 1 by norm W, right? So, distance between the hyperplane and the closest pattern one by norm W intuitively more the margin better the chance of correct classification of new patterns, right? (( )) if if the margin is more, I am leaving much more space for me for some more patterns to come here. So, given that these are i i d examples very unlikely that a pattern from the this class come this side, so more the margin intuitively better is my hyperplane.

So, essentially you want a hyperplane that that has maximum margin. So, we will define optimal hyperplane to be a separating hyperplane with maximum margin. So, we want to find a separating hyperplane W comma b for which 1 by norm W is maximum or norm W is minimum or norm W square is minimum or W transpose W is minimum, right? I

want to maximize margin, so I have to maximize 1 by norm W, which is same as minimizing norm W norm or which is same as minimizing norm W square. So, I essentially want a separating hyperplane for which the margin is maximum or W transpose W is minimum, okay?

(Refer Slide Time: 22:38)



So, let us formulate this among all separating hyperplanes the one with the largest margin is the optimal hyperplane. So, we can now write this as an optimization problem what is the optimization as we seen. We want something that has separating hyperplane that has least margin. That is maximum margin or least W transpose W. So, finding an optimal hyperplane is same as finding a W on b W belong to r m and b belong to r to minimize W transpose W subject to the constraint that this W and b form a separating hyperplane for them to form a separating hyperplane. They have to satisfy y i into W transpose x i plus b greater than or equal to 1 i is the index of the examples, so i goes from 1 to n.

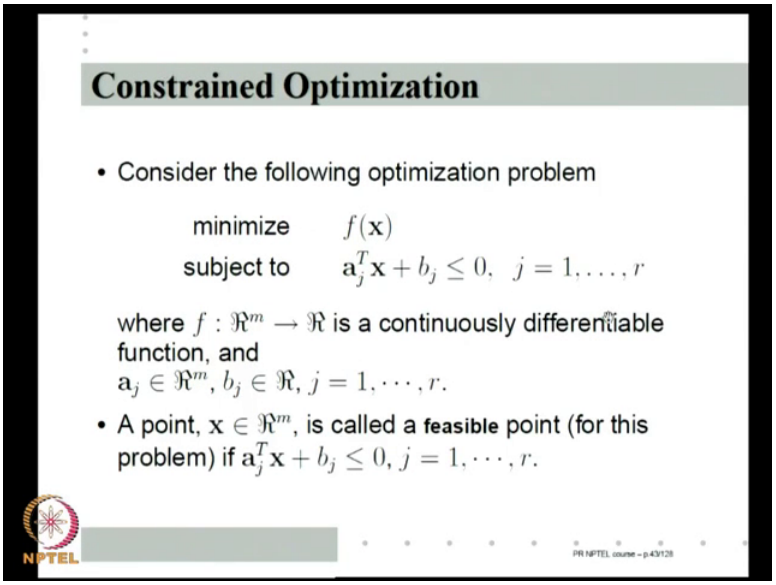
So, for each example x i y i into W each example x i y i y i into W transpose x i plus b should be greater than or equal to 1. This tells me that the hyperplane given by W would not be correctly separates correctly. I mean no pattern in between W transpose x i plus W transpose x plus b is equal to plus 1 and minus 1 among all. Say W b we satisfy this, I want that one which has the least W transpose W or maximum margin, right?

Such things are called constrained optimization problems, right? We want to minimize

this, but do not want to minimize this over all W and b, but only over those W and b which also satisfy this. So, among all W b that satisfy this constraints which is the one that minimizes this. Such optimization problems are called constrained optimization problems. These are unlike the optimization, we come across so far where when we are minimizing empirical risk is just some function to minimize is like finding maximum or minimum of a of a high dimensional function with just a function f of x.

You are asking which x will minimize this among all x in in the space is now no longer that right among all like that satisfies some constraints, which one will minimize this function such thing are called constrained optimization problems. Since, some of you may not have done constrained optimization earlier I will give you a very brief review of constrained optimization.

(Refer Slide Time: 24:52)



So, any general constraint optimization problem is of the following kind minimize some function f of x subject to some constraints. We are only interested in what are called linear constraints. We will take them to be linear constraint. So minimize f of x subject to a j transpose x plus b j less than or equal to 0 j is equal to 1 to r. So, I have r constraints a 1 transpose x plus b 1 should be less than 0 a 2 transpose x plus b 2 should be less than or equal to 0 and so on so forth. f is a function that maps r m to r and we assume it to be continuously differentiable.

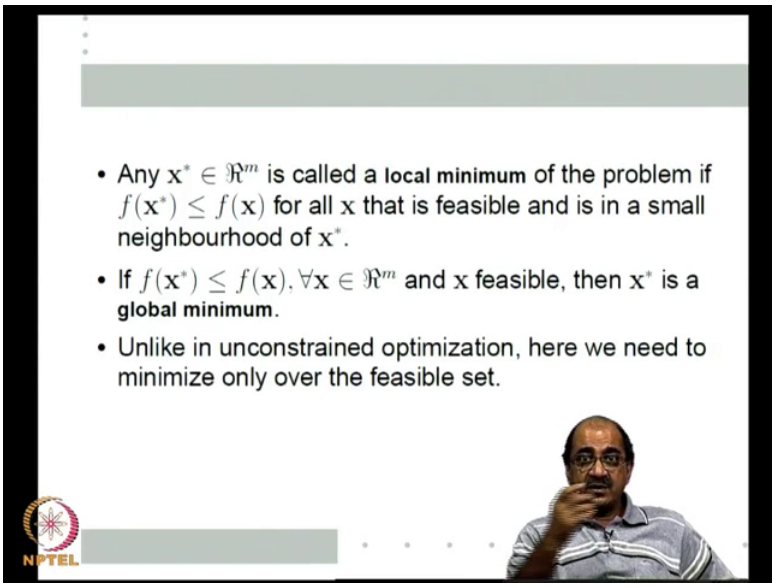
These a j vectors are also in r, because I want to calculate a j transpose x. So, this is some

linear function of x a j affine function of x. Normally you call this as the objective function or cost function or criterion function. So, you want to minimize a cost or a criterion function objective function subject to these are called the constraints, right? So, what it means is search only among those x in r m, which satisfy all these constraints. Among all those x find the x that minimizes f that is the problem such a problem is called a constrained optimization problem, okay?

See in general in an unconditional optimization problem, we have no constraints from just finding an x over over r m that locally minimizes f or globally minimizes f a simplest way of looking. It is sometimes there might be a solution for a constrained optimization problem, but there will be no solution to unconstrained optimization problem. Suppose, I want to minimize 3 x now, there is no no minimum. We can take x as low as you want and I keep going down, so ultimately I hit minus infinity, right? This on the other hand, suppose I want to minimize 3 x subject to x greater than 0, then obviously the minimum with 0 minimum x is right.

So, essentially that is what is meant by when I am, when I want to minimize 3 x. I have to search over all possible x, so I have no no minimum, but when I say subject to minimize 3 x subject to x greater than 0, I am I need to only search over those x, which satisfy my constraints a point x in r m, which satisfies all the constraints is called a feasible point. So, given this constrained optimization problem or optimization problem any x that satisfies the constraints is called a feasible point. So, any x that satisfy a j transpose x plus b j less than or equal to 0 j is equal to 1 to r. All the constraints have to be satisfied then it is called a feasible point. So, essentially we are going to minimize f over those x that are feasible, okay?

(Refer Slide Time: 27:32)

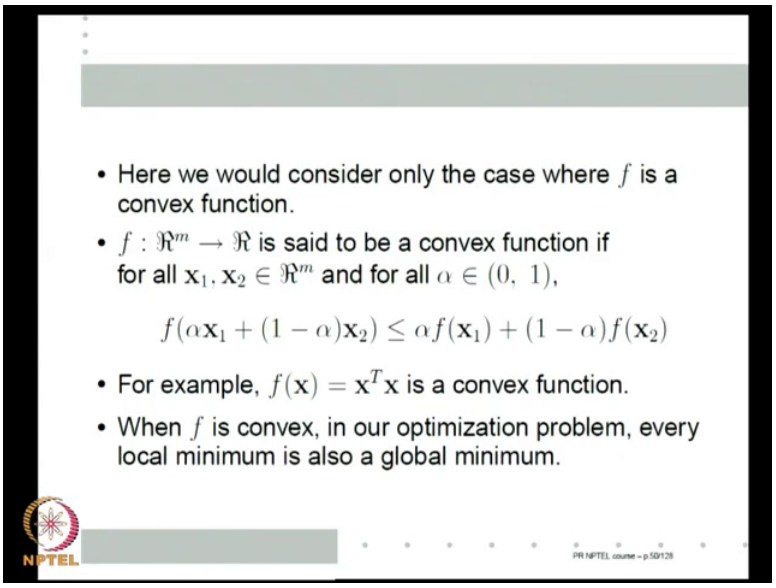


Now, we can define what is a solution any x star in r m is called a local minimum of this problem or local solution to this problem. If f x star is less than f x for all x that is feasible and is in a small neighborhood of x star for an unconstrained problem, we define x star to be a local minimum. If f x star is less than f x for all x in a neighborhood around x star. Now, I want f x star to be less than f x for all x in a neighborhood around x star, but not every x in neighborhood need to satisfy this only all feasible x in a small neighborhood around x star need to satisfy this.

Similarly, for a global solution I want this to be satisfied by all x in r m, but only those x which are feasible. So, normally with unconstrained minimization x star is a global minimum if f x star is a less than f x for all x in r m. Now, for the constrained problem which is a global minimum if f x star is less than f x for for any x in r m that is feasible x in r m that are not feasible. Do not have to satisfy this; that essentially what a constrained optimization problem will be unlike in unconstrained optimization here. We need to minimize only over the feasible set.

Now, how does one solve this? No ,unconstrained optimization problem, we can differentiate and equate to 0; that is what we have been doing, right? The gradient descent is a simple method to solve the unconstrained optimization problem, how do we solve the constrained optimization problem?

(Refer Slide Time: 29:01)

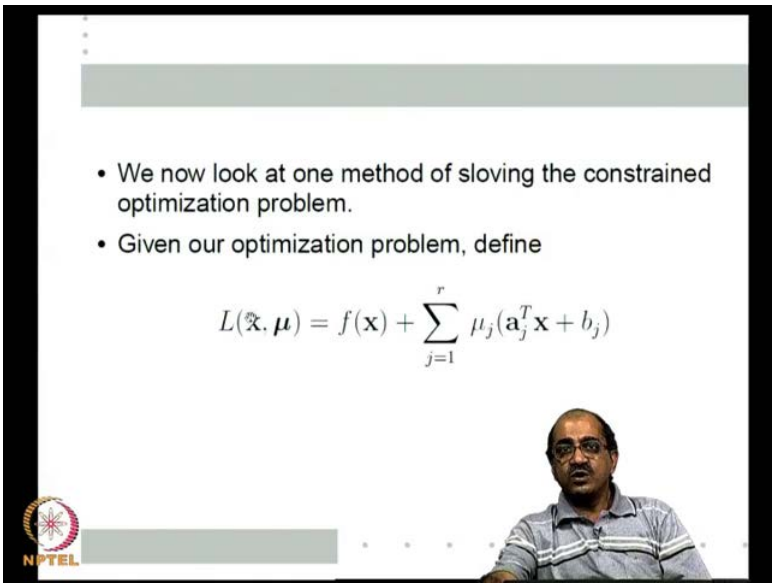


Now, for us for our case we need to for the for the SVM algorithms that we are going to discuss. We need to look at only constrained optimization problems whose constraints are linear and whose cost function or criterion function is convex. So, we will restrict ourselves to only convex functions just in case some of you do not know what a convex function is? A function f r m to r is said to be convex if for every x 1, x 2 in r m and all real numbers alpha in the open interval 0 1; that is at alpha strictly between 0 and 1.

f of of course, it can even be close interval 0 1, but still does not matter f of alpha x 1. That is 1 minus alpha x 2 is less than or equal to alpha times f x 1 plus 1 minus alpha times f x two geometrically. What this means is, if I look at the graph of f x, let us say f is r to r function instead of r m to r, we look at the graph of f. If I take two points; x 1 x 2 and I join them on the graph. Then the actual graph should be below the line that joins is, but anyways this is the condition for convexity.

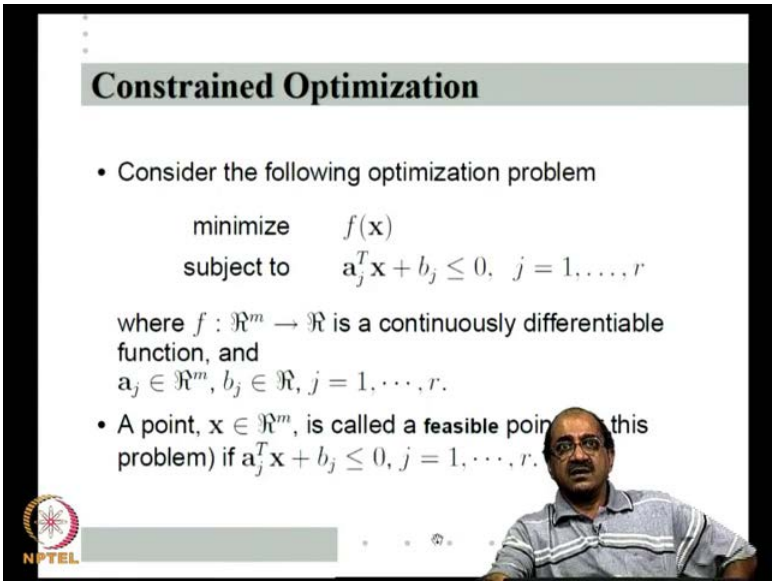
For example, x transpose x is a convex function you can easily verify this when f is convex. The the reason why convexity helps is in the optimization problem every local minimum is also a global minimum. You do not have to worry about local minimum problems when f is convex and the constraints are linear. In that case in the optimization problem every local minimum is also a global minimum. That is a that is a huge advantage when solving an optimization problem, okay?

(Refer Slide Time: 30:45)



Now, how do I solve this constrained optimization problem? Given the optimization problem, we define a function l l is a function of your x x is an r m m vector and also a vector mu mu is has many components as there constraints. They are all constraints, so this vector mu is mu 1, mu 2.

(Refer Slide Time: 31:19)



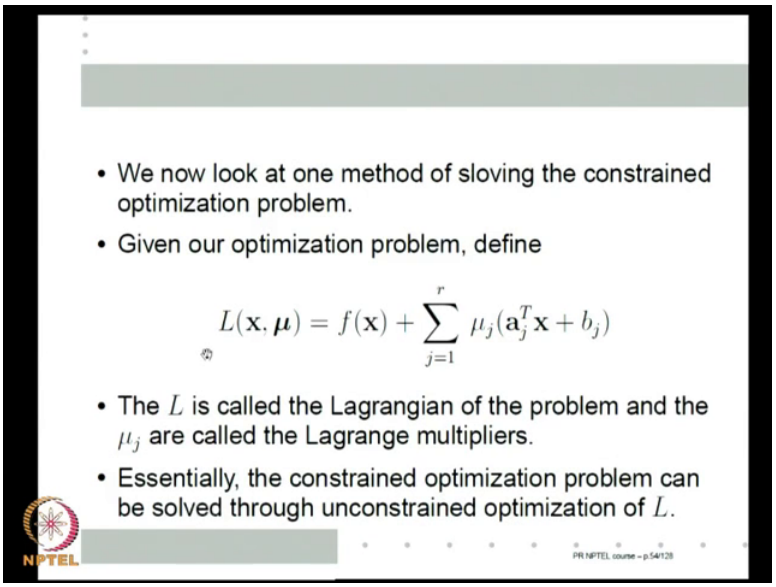
Just a small notational comment just for the optimization tutorials, so to say we will use the notation that all bold figures are x. See so far we have not been distinguishing between vectors and scalars. But just for this optimization part just this three four slides

where I discussed optimization. I will use all bold face characters for vectors, so this x's are vectors a j's are vectors, but b j's are scalars adjust. So, that is a little more easier when we revert back to our pattern recognition. Of course, there we know capital X i feature vector capital W is your weight vector and so on.

So, there we do not use any notation like earlier which are vectors say scalars is clear from context, because this an abstract optimization problem. I decided to use bold face for all vectors, you will see throughout as and when I am discussing this abstract optimization problem. So, this mu is another vector, it has r components mu 1, mu 2, mu r. So, you define a function l a function of x and mu as f of x my criterion function plus j is equal to 1 to all where r is the number of constraints mu j times a j transpose x plus b j.

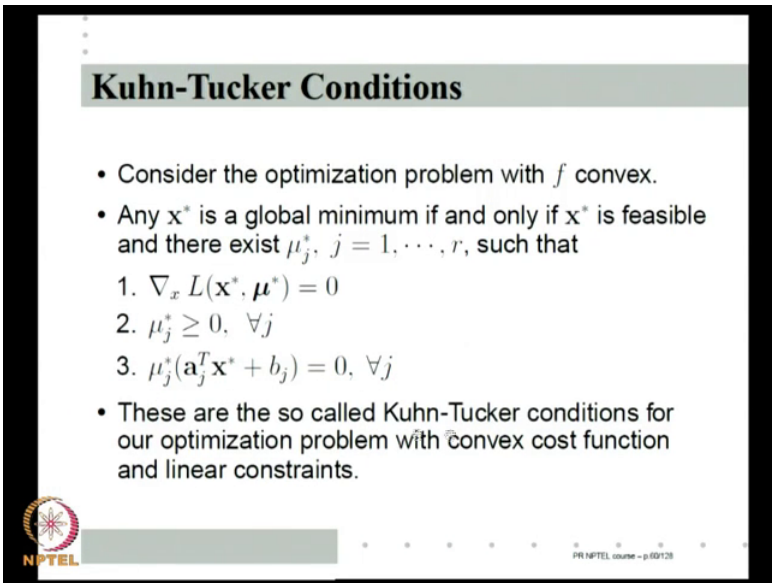
I am writing all my constraints as what are called inequality constraints my constraints are all of the form a j transpose x plus b j less than or equal to 0. So, that l function I am writing is f of x plus some mu j times a j transpose x plus b j summed over j is equal to 1 to r such a l is called the Lagrangian for the problem

(Refer Slide Time: 33:00)



This mu j's are called the Lagrange multipliers, so the Lagrangian essentially adds the criterion function to some unspecified constant times the constraint. The nice thing about this Lagrangian method is that it essentially transform the constrained optimization problem into one of unconstrained optimization of the function l in a in a in a in a specific sense, okay? Let us look at this specific sense.

(Refer Slide Time: 33:32)



So, we are considering the old optimization problem that is minimize f subject to linear constraints a j transpose x plus b j less than or equal to zero we are assuming f is convex now here is the theorem that is of interest to us any x star is a global minimum if and only if x star is feasible when the x star satisfies all the constraints. There exist some constants mu j star or such constants, which satisfy the following one is that the gradient with respect to x the first component. The first vector variable of the Lagrangian Lagrangian is a function of x on the Lagrange multipliers mu.

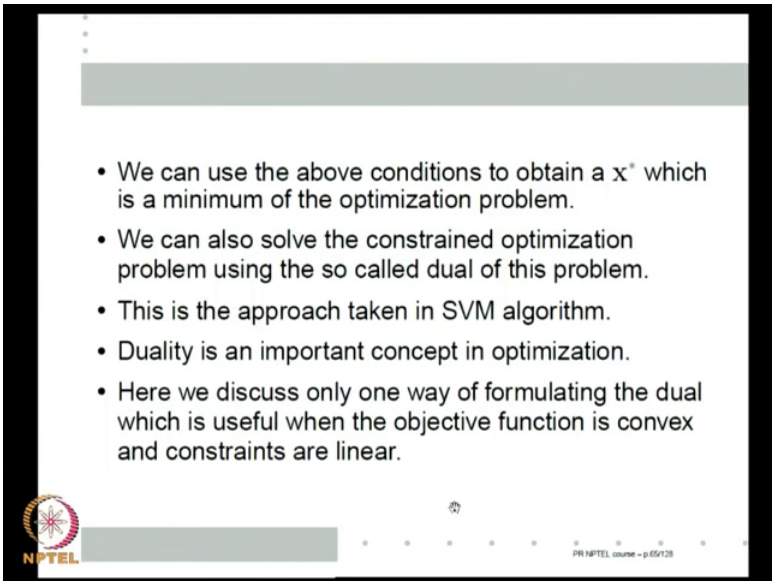
So, the gradient of l with respect to the optimization variables x evaluated x star mu star is 0 2. All the mu j stars are non negative and 3 mu j star into a j transpose x star plus b j star is equal to 0. Now, we know mu j star is to be greater than or equal to 0. So, if mu j star is equal to 0, then anyway this is 0, but if mu j star is strictly greater than 0, then a j transpose x star plus b j has to be 0. I know because x star is feasible a j transpose x star plus b j less than or equal to 0. It may be equal to 0 or it may be strictly less than 0, what the third constraint says is it cannot be strictly less than 0.

If mu j star is greater than 0, if mu j star is greater than 0, the constraint has to be satisfied by equality. This third third condition is often known as a complementary slackness condition, so whenever the Lagrange multiplier mu j star are positive. Then the corresponding constraint has to be satisfied by equality right? So, these are called Kuhn Tucker conditions. So, given this optimization problem where the cost function is convex

and the constraints are linear. These are the conditions for any x star to be a global minimum what are the conditions x star should be feasible.

There must exist Lagrange multipliers mu j star such that gradient with respect to x of l of x star mu star is equal to 0. All the mu j stars are positive and if any mu j star is strictly greater than 0, then a j transpose the corresponding for the for that j a j transpose x star plus b j should be equal to 0, right? This is called the complementary slackness.

(Refer Slide Time: 36:15)



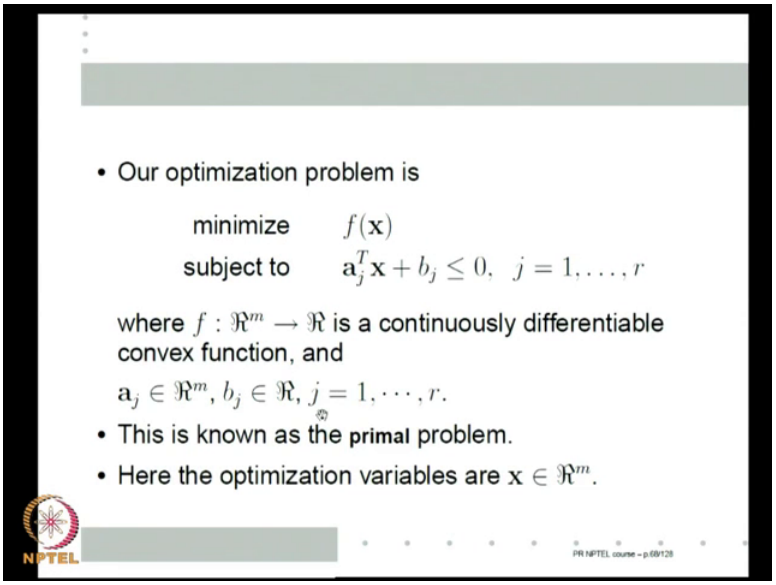
Essentially, because we know all the a j's and b j's and we can f is given to us. I can calculate l, I can actually you know find the gradient of l equate it to 0. Then make some guesses about which of the constraints are active which are inactive right active meaning when is a j transpose x star plus b j equally 0 1 is strictly less than 0, right? So, I make guesses about which mu j's are 0, which if I am decide this mu j's are 0, right? Then for the remaining mu j's, I have the extra for remaining indices j. I have extra equations a j transpose x star plus b j is equal to 0, so I can solve a j transpose x plus b j is equal to 0.

This this gradient equal to zero equation together to obtain all the mu j's and the x's, okay? So, there is a way of solving the constrained optimization problem by by finding an x star, which satisfies this Kuhn Tucker conditions. There are there many other methods there are just like gradient descent. There are other numerical techniques for solving this, but we will look at one particular technique which is used for the support vector machines, which is solving it using. The so called dual of the problem duality is a

is a very very important and a deep concept in optimization.

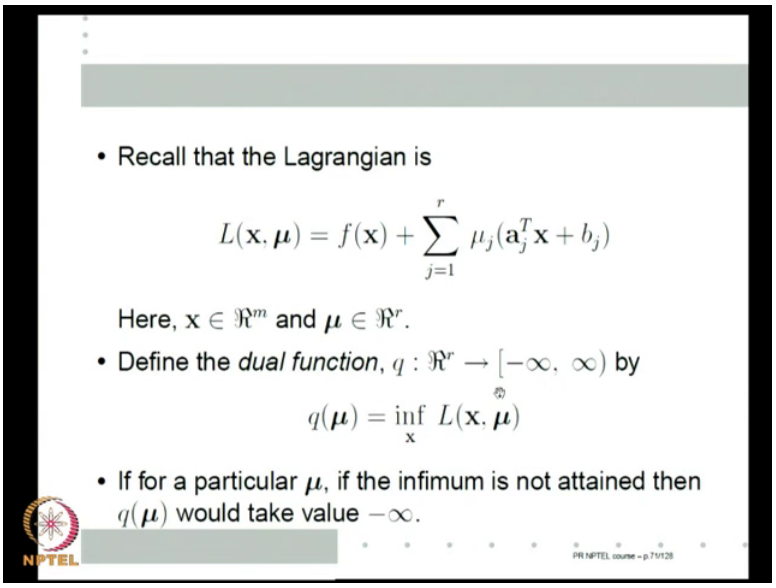
Once again in this course, we will not be able to teach duality, but those of you may not have gone through an optimization course, we will just look at duality in the very very limited context, right? We will just look at one way of formulating another optimization problem, which we will call the dual of this optimization problem, but even that method will work only for optimization problems, where the cost function is convex and the constraints are linear. So, only in that, only for that case we will we will look at how to find duals, okay?

(Refer Slide Time: 38:21)



So, the given optimization problem is to recall minimize f of x subject to a j transpose x plus b j less than or equal to 0. We are assuming f is continuously differentiable and also convex now. a a j's are vectors in r m, so this is we are given r such constraint. This given optimization problem is called the primal problem. In this optimization problem the minimization is over x. x is in r m, so the optimization variables are x belonging to r m. This is what we are minimizing over so the dimensionality of this optimization problem is m. We are you are searching over r m to find the minimum.

(Refer Slide Time: 39:03)



Now, this is the Lagrangian (( )) we for x in r m and mu in r r this small r is the number of constraints. We have l of x mu is f x plus summation j is equal to 1 to r mu j a j transpose x plus b j given the Lagrangian. Let us define a function q which we call the dual function, which maps r dimensional real vectors, where r is the number of constraints to not real line. Slightly extended real line minus infinity to infinity minus infinity included infinity not included in real line, even my symphony is not included, it will just allow the q function to sometimes take minus infinity also, as values where q is defined.

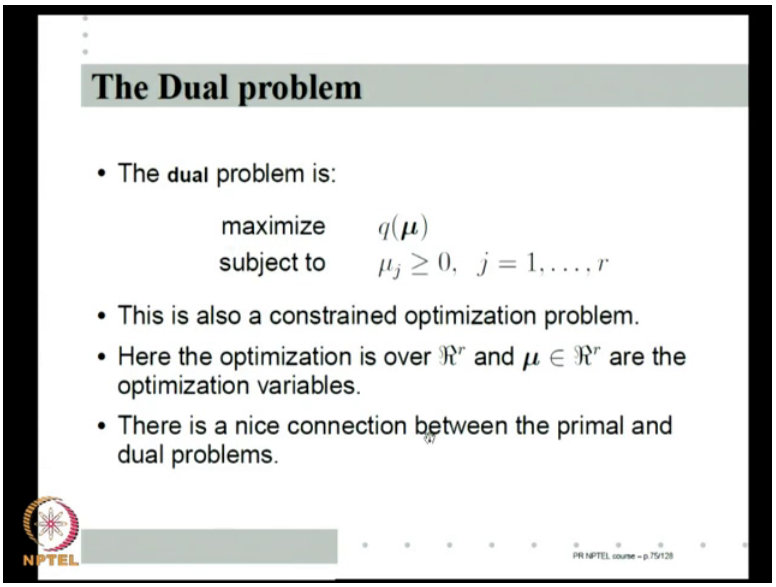
Now, q's domain is r r, so q is defined on vectors of the type mu. So, q mu is taken to be infimum over all x in r m of l x mu where l x mu is this. So, given any mu in r r i define the value of q mu to be the infimum or minimum over all possible x of l x comma mu. The difference between infimum minimum is minima has to attained. So, infimum is the greatest lower bound, we are asking as x goes over all possible vectors in r m. If I take this numbers l x mu, what is the greatest lower bound of this?

So, sometimes it that that is a number that is actually as the value of l x mu for a particular x sometimes, it might be a limit, because it can be a limit sometime the infimum may not be attained. Suppose, l x mu just happens to be minus b times x or summation summation x i b times summation x i or suppose is equal to x 1 just like minimizing 3 x, there will be no minimum. This kind of infimum may not be attained,

right? So, if this is linear in any component of x, this is if there is a linear term in any component of x, then infimum will be minus infinity.

That is the reason we allowed q to take minus infinity as one of the possible values that if a particular mu the infimum is not attained. Then q mu would take value minus infinity just to allow that we define q to be a function from r r to this as an r r to r, because we defined it as infimum. We have to allow the possibility that for some mu q mu may take value minus infinity. Now, so you give me this primal optimization problem, I know f i know a j b j. So, I can calculate my Lagrangian. Once I can calculate my Lagrangian for every single mu, I can calculate q mu.

(Refer Slide Time: 41:47)



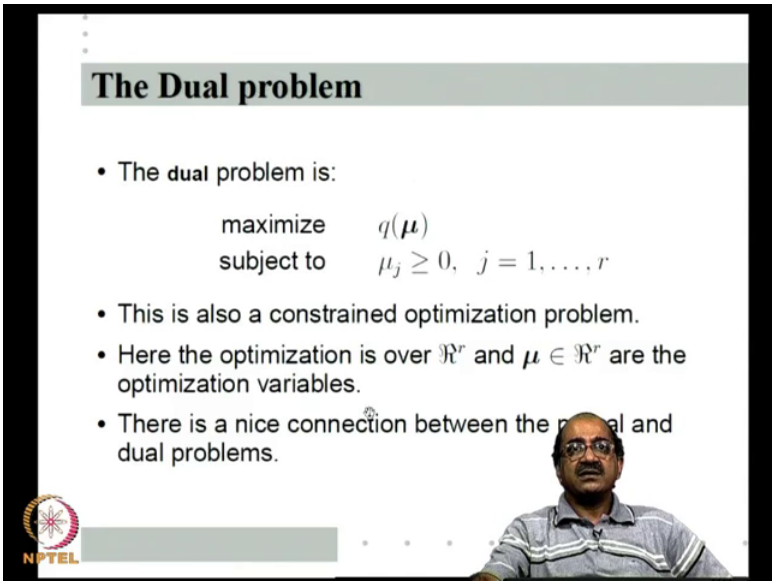
I know the function q mu, so now I formulate another optimization problem maximize. This term q mu subject to all the mu's greater than or equal to 0. This is also a constrained optimization problem, what is a optimization variables here is mu's. So, the here optimization is over r r where little r is the number of constraints we had in the primal. So, here optimization is over r r and the optimization variables are mu. This problem is called the dual to the other problem, right?

So, given this optimization problem using these f and a j's and b j's i formulate the l. Then I formulate the q, then I formulate a new optimization problem, which is also a constrained optimization problem over r r with mu as the optimization variables. This problem is called the dual to that primal problem and there are some very nice

connections between the primal and the dual problems. What is the connections if the primal problem has a solution?

Then dual also has a solution and the two optimal values are same, what does that mean? This is the primal problem x star could be a solution. Then it if a if it is a solution which satisfy the Kuhn Tucker conditions and the optimal values f of x star here, right? This is the dual problem if the solution exist. Of course, this will have its own Kuhn Tucker conditions with respect to the optimization variables mu. If it has a solution, it could be mu star and the optimal value will be q mu star.

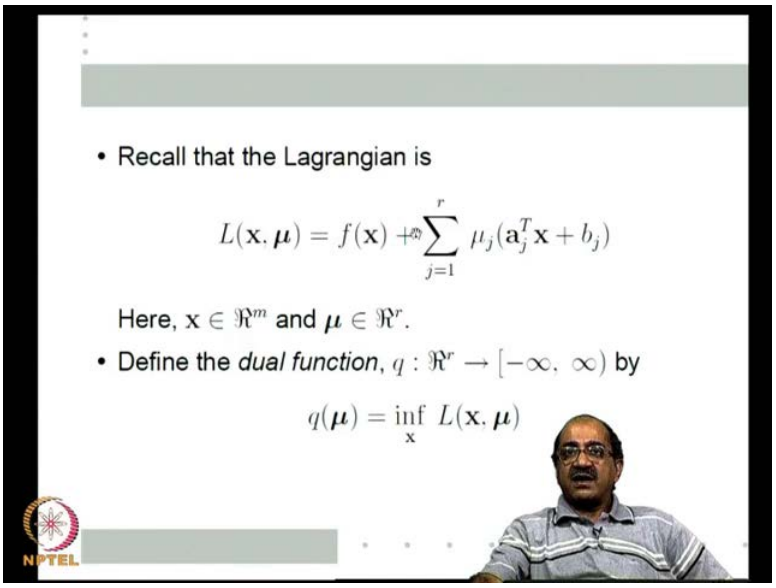
(Refer Slide Time: 43:35)



What the theorem says is that if the primal has a solution x star, then there will also be a mu star which is a solution to the dual and f of x star will be equal to q of mu star. The optimal values are equal 2 for x star to be an optimal solution for the primal and mu star to be an optimal for the dual. These are the necessary and sufficient conditions x star is optimal for primal and mu star is optimal for dual, if and only if the the two conditions are satisfied.

One is x star is feasible for primal and mu star is feasible for the dual obviously it has to be the second condition is f of x star should be equal to l of x star mu star should be equal to min over x of l of x mu star. Yes, while the proof is a little complicated, we can at least see by where the condition comes from now essentially.

(Refer Slide Time: 44:37)

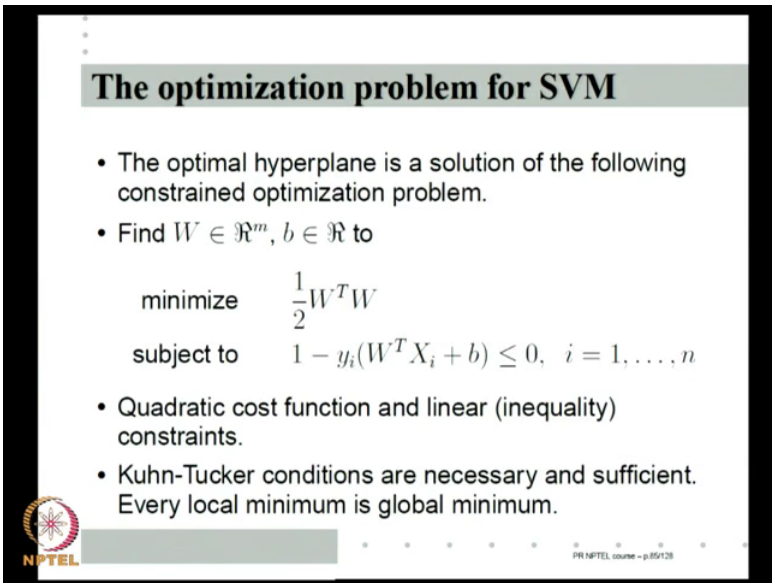


f of x star is equal to l of x star mu star, what does this condition mean? If f of x star should be equal to l of x star mu star at x star mu star this term should be 0, because of feasibility all mu's are of the same sign a j transpose x plus b j are of the same signs. So, nothing can cancel. So, the only way f of x star should be equal to l of x star mu star is if each term here is 0 mu j star into a j transpose x star plus b j is equal to 0, which is my complementary slackness condition, right?

Similarly, if I say f of x star is equal to minimum over x l of x mu star that means q mu star will be l of x star mu star, which will be equal to f of x star, right? So, at least we can see why this conditions are interesting, so this is the primal dual relationship. The primal has a solution so does the dual and the optimal values are equal for x star to be optimal for the primal.

Mu star to be optimal for the dual x star should be feasible for the primal mu star is feasible for the dual and f of x star should be equal to l of x star mu star should be equal to minimum over x l of x mu star. We would be using the dual formulation for the SVM, okay?

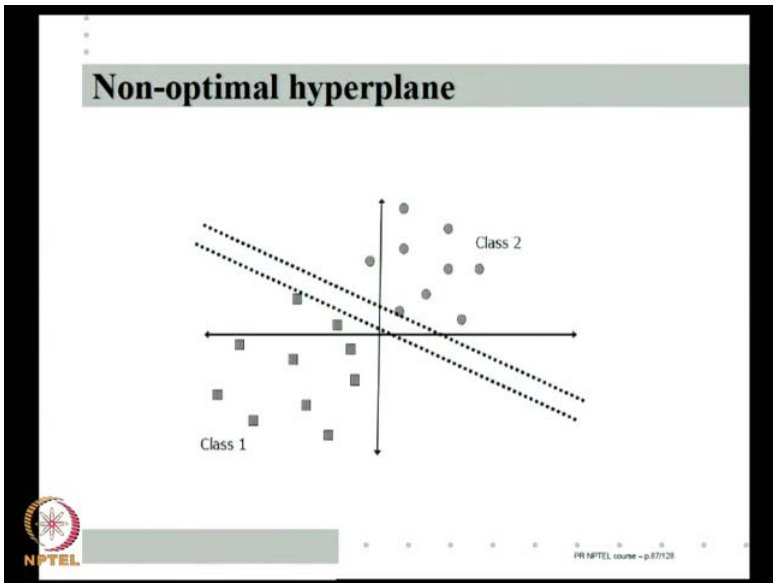
(Refer Slide Time: 45:58)



Now, let us get back to the SVM optimization problem say optimal hyperplane is a solution of what this contained optimal? This is the primal problem minimize half and W transpose W I have written. Now, this in the my standard form I am writing my constraints as less than or equal to 0. So, I wrote it as o1ne minus y i into W transpose x i plus b less than or equal to 0. There n constraints because there are n example. The W has the same dimension as the vector x, which is m. So, I want to minimize half W transpose W, which is a convex function.

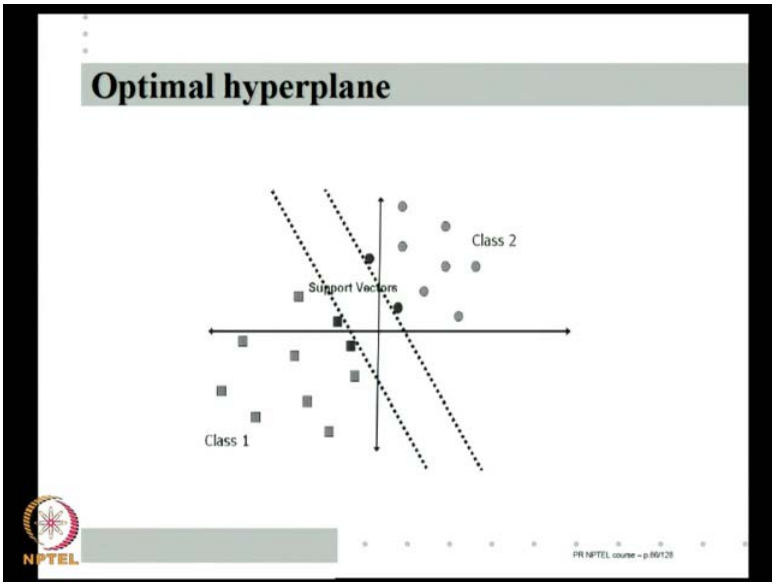
So, my what I called f in the abstract optimization problem is this which is convex and this is our linear constraints 1 minus y i into W transpose x i plus b less than or equal to 0. There are n constraints, this is a quadratic cost function convex quadratic function is a convex function, linear constraints. Hence, Kuhn Tucker conditions are necessary and sufficient every local minimum is a global minimum. So, essentially to appreciate, what this means is what we are asking for is minimization of W transpose W, which is same as maximization of the margin.

(Refer Slide Time: 47:14)



So, if you have a general thing margin is essentially you put your hyperplane somewhere in the center, so on either side of hyperplane you try to make a margin till you hit one of the patterns. So, in this direction, I get so much margin. On the other hand if I change my direction, there is only so much margin possible, right? So, this is a way of looking at what optimality means in terms of maximizing margin.

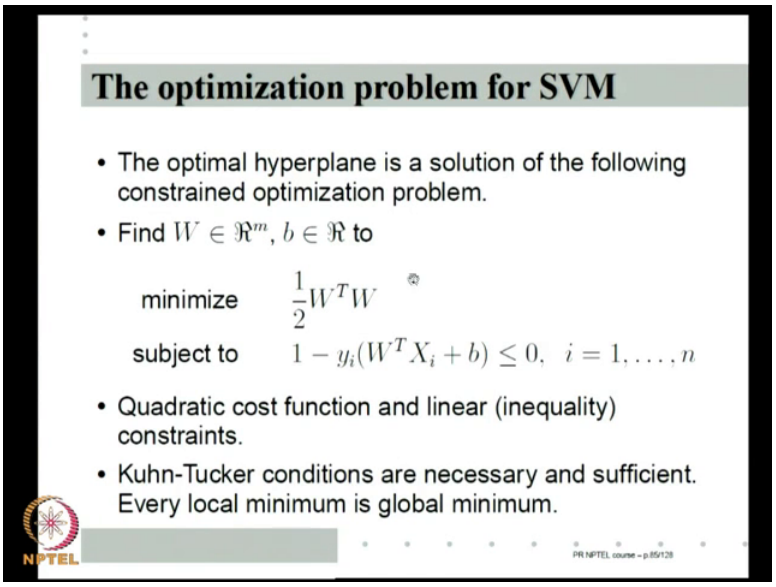
(Refer Slide Time: 47:45)



Hyperplane in this direction has low margin as hyperplane in this direction has high margin. So, I am trying to find that separating hyperplane, which has the higher margin

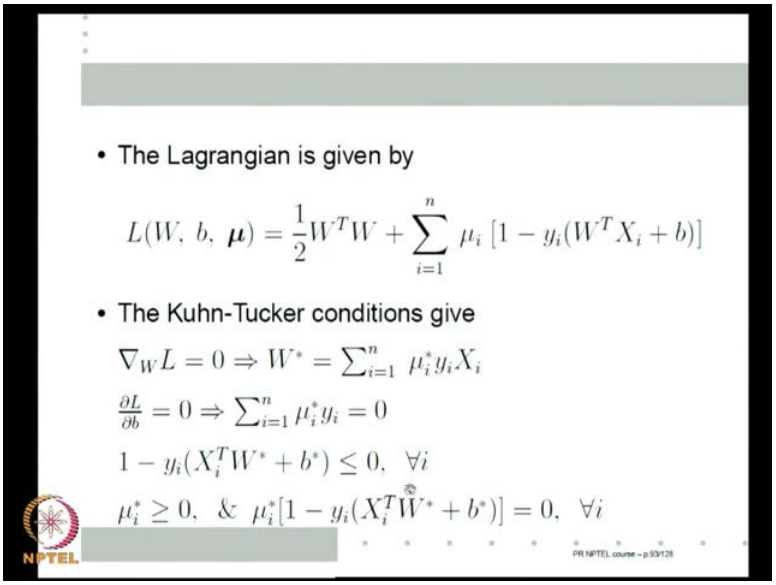
by the, by there are something written as support vectors here. Do not worry about this, we will come back to this figure again and then I will explain that support vectors, okay?

(Refer Slide Time: 48:13)



So, this is my optimization problem. This is f x and these are my constraints.

(Refer Slide Time: 48:19)



So, my Lagrangian will be my optimization variables are W and b mu's will give me a Lagrangian multipliers. This is the f x, this is the cost function these are the constraints mu i times 1 minus y i into W transpose x i plus b. This is my Lagrangian, so if I now apply my Kuhn Tucker conditions, what does the Kuhn Tucker conditions give first is

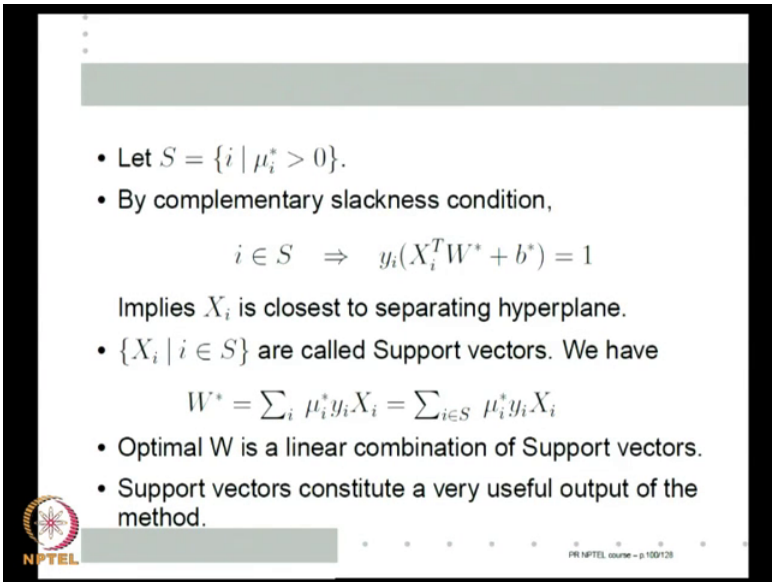
that the gradient of l with respect to the optimization variable should be 0. That is gradient with respect to W should be 0 and derivative with respect to partial (( )) with respect to b should be 0.

So, first take gradient with respect to W of l this term will give me a W, just so that this term will not give me 2 W, but W. I put a half here right from this term. There is only 1 W term here that is mu i y i W transpose x I, so gradient of that with respect to W will be mu i y i x I, right? That will come with a minus term, so if I equate it to 0, I get W the optimal W is summation over i is equal to 1 to n mu i star y i x i.

So, by equating gradient of l with respect (( )) to 0 i get w star s summation i is equal to 1 to n mu i star y i x i. If I equate derivative with respect to b to 0 is only one term here that gives me b. That will be mu i y i b summation i is equal to 1 to n mu i y i b derivative of that with respect to b will be summation mu i y i. So, at optimal Lagrangian multiplies summation i equal to 1 to n mu i star, y i should be equal to 0.

Then Kuhn Tucker conditions also demand feasibility. So, for feasibility I need to have 1 minus y i into W transpose x i plus b star should be less than or equal to 0 for all i and i need complementary slackness. So, mu i star into 1 minus into the constraints 1 minus y i into W transpose x i W star transpose x i plus b star should be equal to 0. All mu i should be greater than or equal to 0. These are my Kuhn Tucker conditions, okay?

(Refer Slide Time: 50:40)



So, W star is given by mu i star y i x i and this is my this is my complementary slackness condition. Let us say this define the set of indices i for which mu the corresponding Lagrange multipliers are strictly positive by the set s by complementary slackness. What is complementary slackness give me? mu i star into 1 minus y i into x i transpose W star plus b star equal to 0, so for any i if mu i star is strictly greater than 0.

Then 1 minus y i into x i transpose W star plus b star should be equal to 0. Now, for every i in the set s mu i star is strictly greater than 0. That is the definition of the set s, so i belongs to s implies mu i star is greater than 0. Hence, y i 1 minus y i into that is equal to 0, which is same as y i into x i transpose W star plus b star is equal to 1.

What does that mean? x i transpose W star plus b star is equal to y i by multiplying y i on both sides by y i. Note that y i has a plus 1 or minus 1, so y i square is always 1. It essentially means x i transpose W star plus b star is plus 1, if x i is in the plus 1 class minus 1. If x i is in the minus 1 class, so actually x i is on the hyperplane. x i transpose W star plus b star is equal to plus 1 or minus 1, which means if mu i star is greater than 0, then the corresponding i x i is closest to the separating hyperplane.

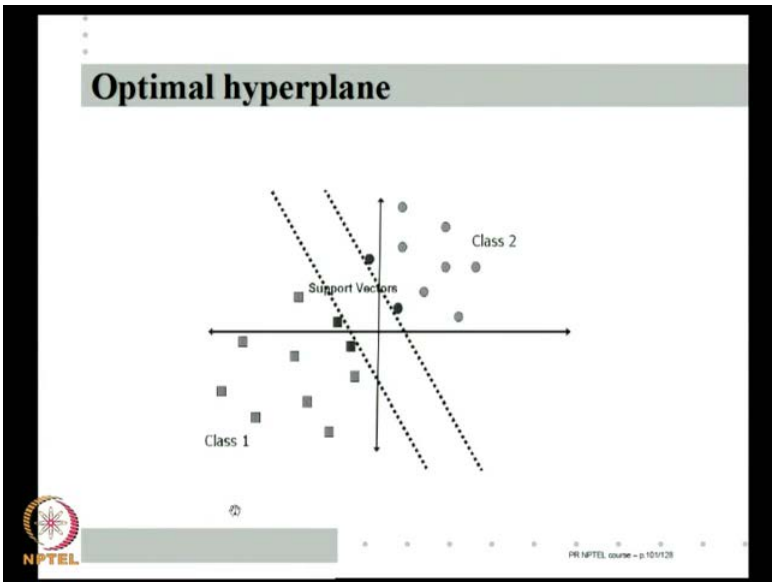
So, we call x i where i belongs to set s as support vectors see there are as many constraints as there are training examples. So, corresponding each x i there is a constraint. Hence, corresponding each x i there is a Lagrange multiplier mu i because there are as many Lagrange multipliers as their constraints. So, we can actually say each Lagrange multiplier is for one example. So, if the Lagrange multiplier is strictly greater than 0, the corresponding example a corresponding x i is the one closest to the separating hyperplane.

These are called the support vectors and W star. We know is summation over i mu i star y i x i if mu i star is equal to 0. Anyway that term does not contribute, so you can actually take W star to be summation over i belonging to s mu i star y i x I, which means the optimal W is a linear combination of the support vectors. Of course, when we did perceptron, we know we knew that essentially a separating hyperplane will be a linear combination of the x i of the examples. But we know that the optimal hyperplane now is a specific linear combination of only some of the examples not all the examples.

These examples are called the support vectors and those happen to be the ones closest to the hyperplane. As a matter of fact support vectors constitute a very useful output of this

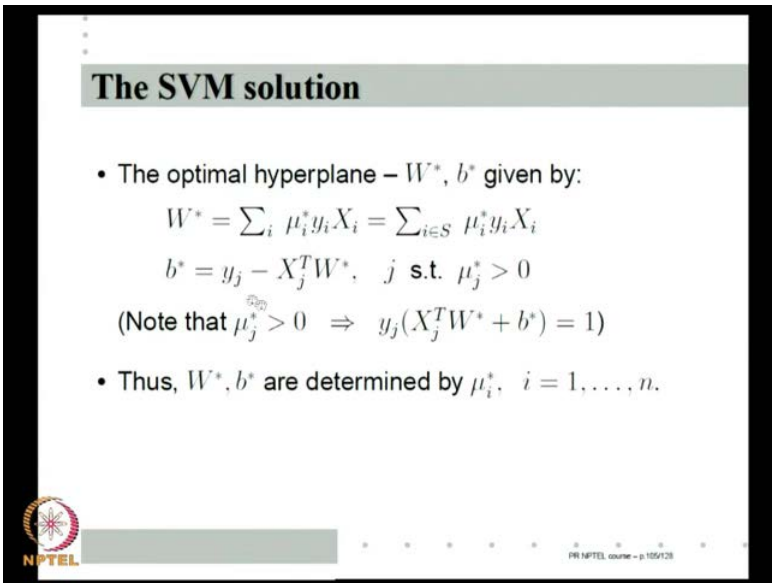
method. Apart from the W star identifying which of the example that are closest to the separating class separating boundary is also a useful by product of the SVM method. Now, because these x i's are called the support vectors and the optimal hyperplane is can be written as a linear combination of the support vectors. The method is called a support vector machine, okay?

(Refer Slide Time: 53:49)



That is just take a look at it now. So, this is if this is my optimal hyperplane, right? So, these the ones which are dark here right these two and these two these two these two of class two and these two of class one patterns; they happened to be the support vectors they are the ones closest to the separating hyperplane, okay?

(Refer Slide Time: 54:12)

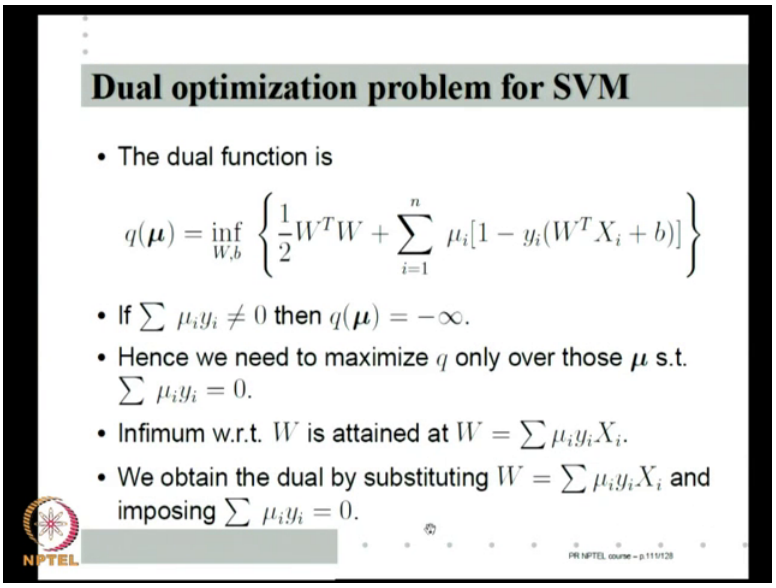


So, let us sum it up the optimal hyperplane W star b star is given by W star is summation over i mu i y i x I, which is same as i belongs to s mu i star y i x i where s is the set of all indices i for which the corresponding Lagrange multipliers such strictly positive. If I know all the Lagrange multipliers, I can also find b star because the complementary slackness says for every j y j into x j transpose mu star with b star into mu j star is 0. So, if if there is any j's has that mu j star is greater than 0, then y j into x j transpose W star plus b star should be equal to 1.

If I multiply both sides by j and take this term on that side i get b star is equal to y j minus x j transpose W star for any j's, such that mu j star is greater than 0. So, every j such that mu j star is greater than 0, I can do this. I will get the same value that is what optimization theory tells me. So, if I know all the Lagrange multipliers mu's, then I can calculate both W star and b star. Thus W star and b star are determine by mu i star i is equal to 1 to n.

If I know all the Lagrange multipliers, then I know the support vectors W star is a specific linear combination of support vectors, where the weights in the linear combination depend on Lagrange multipliers. b star can also be given in terms of the the W star. So, we can obtain the, we can now solve the dual of the optimization problem to get mu stars, okay?

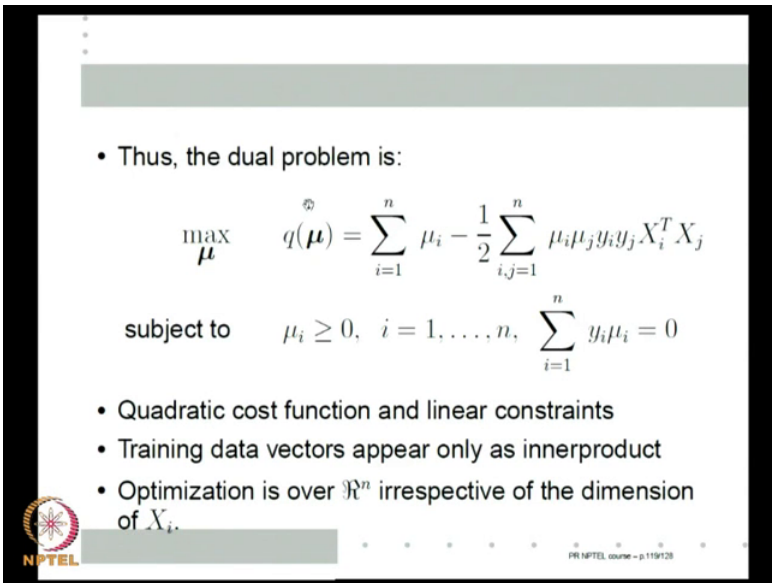
(Refer Slide Time: 55:36)



So, for this problem, what is my dual function q mu is infimum over W b of my Lagrangian is half W transpose W plus this all the constraint mu i into this. Now, if I look into this, I have a mu i y i b right summation i is equal to 1 to n mu i y i whole into b is 1 term minus. Now, I am taking infimum both with respect to W and b, so as you already know it is like minimizing 3 x. It is some b into some summation i is equal to 1 to n mu I, so if mu i if summation mu i y i is not equal to 0.

Then q mu is equal to minus infinity because infimum will be only minus infinity, which means ultimately because I want to maximize q. There is there is there is no I need to only look at those mu for which mu i y i is equal to 0. Now, how do I get infinity with respect to W? I want to minimize this with respect to W, so take the gradient with respect to W and equate to 0. That we know already know the solution infimum with respect to W is attained by taking W is equal to mu i y i x I, right?

(Refer Slide Time: 56:58)



So, essentially I get my dual function by substituting W is equal to mu i y i x i and imposing mu i y i is equal to 0. So, if we did this we will get the dual of the SVM optimization problem. We will we will look at the details once again next class, but right now I will show you the final expression. This turns out to be the dual of the SVM optimization problem, right? Once again in terms of mu this is a linear terms, this is a quadratic term. So, it is once again a quadratic cost function and the constraints there are so many inequality constraints and one equality constraints.

But all constraints are still linear, so it is the dual is once again a quadratic is a constrained optimization problem with quadratic cost function linear constraints. Another very important thing for us we will, I will once again stress it in the next class is that the training examples come only as inner products here. x i transpose x j that is the only way the training examples enter into this optimization problem, which which happens to be very useful to us later on. We will remember it because the dual problem the maximization with mu mu is the vector of Lagrange multipliers.

There are as many Lagrange multipliers as there are constraints in the primal problem. There is one constraint per example in the primal problem, so the optimization is over r n where n is a number of examples irrespective of the dimension of x. No matter what is the dimension of x is this optimization problem is only over x. So, this is what we will remember. So, we have introduced the optimization problem for the SVM. We looked at

the primal problem and we will we will look at the dual problem in the next class. Then see how to generalize it to a linearly non separable case.

Thank you.