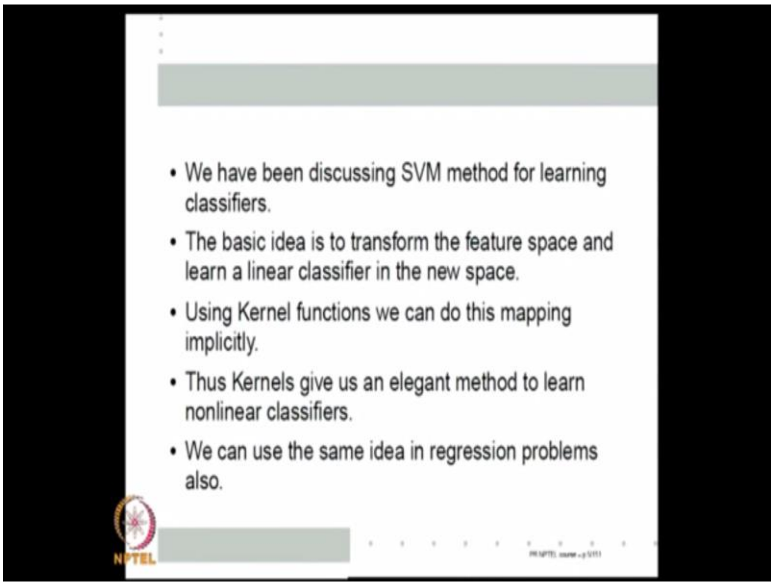
**Pattern Recognition**
**Prof. P. S. Sastry**
**Department of Electronics and Communication Engineering**
**Indian Institute of Science, Bangalore**

**Lecture - 35**
**Support Vector Regression and E-insensitive**
**Loss function, examples of SVM learning**

Hello and welcome to this next lecture in pattern recognition. We will continue with the support vector machine method. We saw last class the basic support vector classifier for two class case and we have also look at the kernel functions this class. We look at how to poss the regression problem also in the same way. So, we derive a support vector regression algorithm also and then discuss a few issues about how actually we solve for the support vector machine. So, just to briefly recapitulate, we have been discussing the SVM method for learning classifiers.

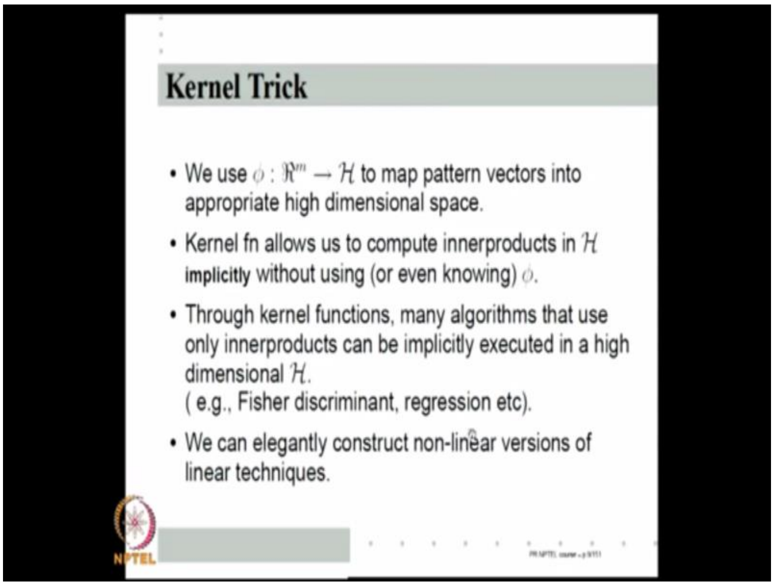
(Refer Slide Time: 01:01)




The basic idea of the SVM method is that we transform the feature space that is with we map the feature vectors into some other space and then learn a linear classifier in the new space. So, basic idea is what is a linear classifier in this new space would be a nonlinear classifier in the old space. So, we can actually learn a nonlinear classifier using techniques of learning a linear classifier. The Kernel functions allow us to do this mapping implicitly and thus Kernels given elegant method to learn nonlinear classifiers.

So, essentially what is a nonlinear classifier in the original space could be a linear classifier in the transform space and the Kernel function allow us to do this mapping implicitly. So, Kernels give us a elegant way to learn nonlinear classifiers the same idea can be use in regression and many other problems. So, let us basically recapitulate what the Kernel Trick is.

(Refer Slide Time: 01:55)

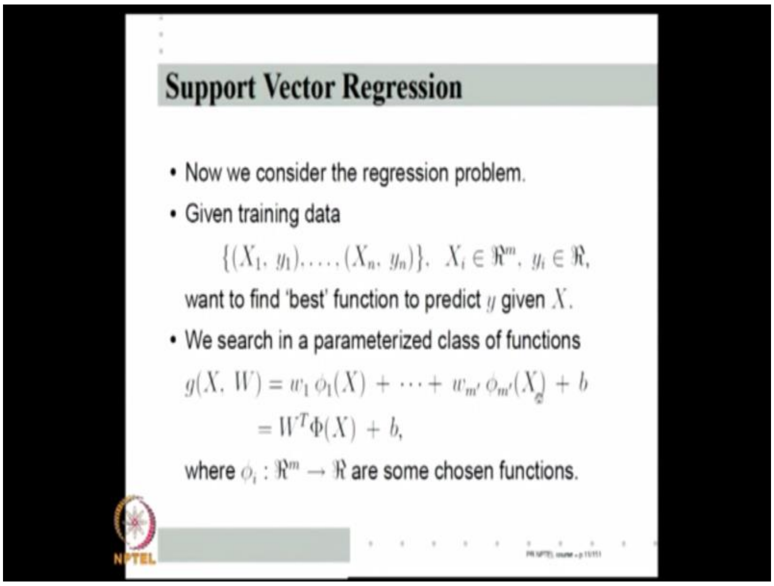



Kernel Trick is if originally the feature space is m dimensional let say R m we are mapping the feature vectors from R m into some other space I called it H it could be R m prime most of the time or just called it H which is some appropriate high dimensional space a vector space in which there is an inner product define. What the Kernel function allows us is to compute inner products in the space H. So, if I have two vectors in R m say x and x prime I want to calculate the inner product between phi x and phi x prime I want to calculate phi x transpose phi x prime there is inner product in the H. I can compute them as a function of just x and x prime.

So, Kernel function allows us to compute this inner products implicitly without using phi or without even knowing what the pie is given x and x prime. I can derive a function of x and x prime which happens to be the inner product of phi x and phi x prime. So, kernels allow us to do the computation of inner products in the H space implicitly and with no knowledge of the function phi. So, what it means is by use of kernel functions many algorithm that essentially use only inner products can be implicitly execute a high dimensional space for example, if I look at Fisher discriminant the discriminant which is w transpose x as well as the way I learn it using generalize diagonal value problem all of them only use inner products.

So, I can actually if I think of Fisher discriminant as the finding a linear transform on to one dimensional space where I get the best separation between classes I can ask can I find the nonlinear transform. So, I can have a Kernel Fisher discriminant and so on. So, any algorithm that uses essentially inner product can be implicitly execute in a high dimensional space that is the basic idea of the Kernel Trick it can be using Fisher discriminant and it could be using regression it can be using many other similar technique algorithms. So, basically what it means is using kernels we can elegantly construct nonlinear versions of linear techniques. Now, we going to look at this in the context to regression. So, far you looked at it the context of classification now we will look at it in the context of regression.

(Refer Slide Time: 04:17)



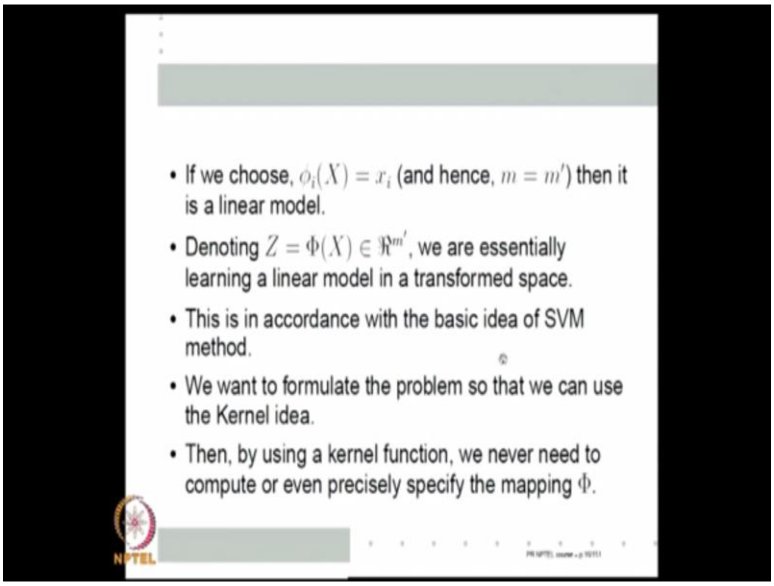

What is the regression problem were given training data X 1 y 1 X n y n and training samples where each X i is in R m is of like our feature vector only thing the targets now y 1 y i are not plus one minus one but, they are the real value targets. So, given this training data I want to learn the best function to predict y given an X we want to blend a function f such that y y had is equal to f of X is a very good prediction for y.

As you seen earlier essentially we can construct such functions in a parameterized manner X w 1 phi 1 X w 2 phi 2 x up to w m prime phi m prime X plus b where m prime is some arbitrary number. lets write this phi 1 phi 2 phi m prime as the vector capital phi and the rest of it as w. So, I can write as W transpose phi X plus b with a bit of use of notation I am putting the parameterization in the left hand side only as W even thou W can as you already seen we can put b inside or outside depending on our convenience.

So, even thou I have a b here I am think of the parameterization is W but, minor point apart this is what we seen when we want when we used the linear least square regression we said as long as the functions phi 1 phi 2 phi m prime are fixed this is like linear regression. I can always fix some functions I would have choose what the functions should be and I would have choose what I am trying would be then I can do this. So, phi are as in general R m to R functions.

Now, essentially if phi i X is equal to X i that will give me w 1 X one plus w 2 X 2. So, one of course, which will mean m prime is equal to m. Then, it is a linear model where simply W transpose X plus b. In general, of we have seen this as a generalize linear model earlier but, now we can think of this as if I think of phi 1 phi m prime phi 1 X phi 2 x phi m prime X as an m prime dimensional vector represent that as capital phi of X which is a transform regression of X.

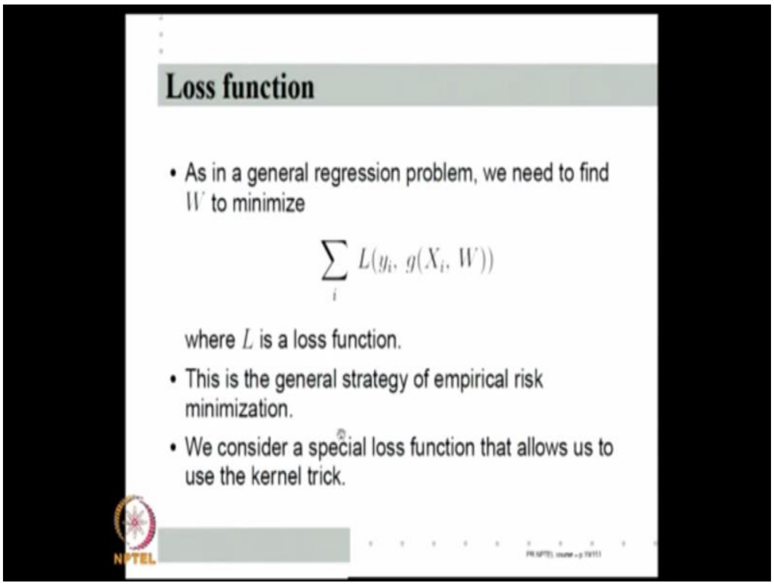
(Refer Slide Time: 06:32)



So, I can think of Z is equal to capital phi of X which is in R m prime and then we are essentially learning a linear model in a transformed space. This is basically the basic idea of SVM is but, even in our earlier least square regression we said if we fix the function phi 1 phi 2 phi m for example, we will see how to learn polynomials in 1 variable simply as a linear regression problem. If I fix the functions phi 1 phi 2 phi m prime its no difference from linear regression the main thing that you want to gain now is even if I choose the functions phi 1 to phi m prime I have to work in the m prime dimensional space I have to first transform all the X's to Z's and then do a linearly squares regression in a Z space that is not what we want to do. We want to do this problem formulating such a way that you can use the Kernel idea.

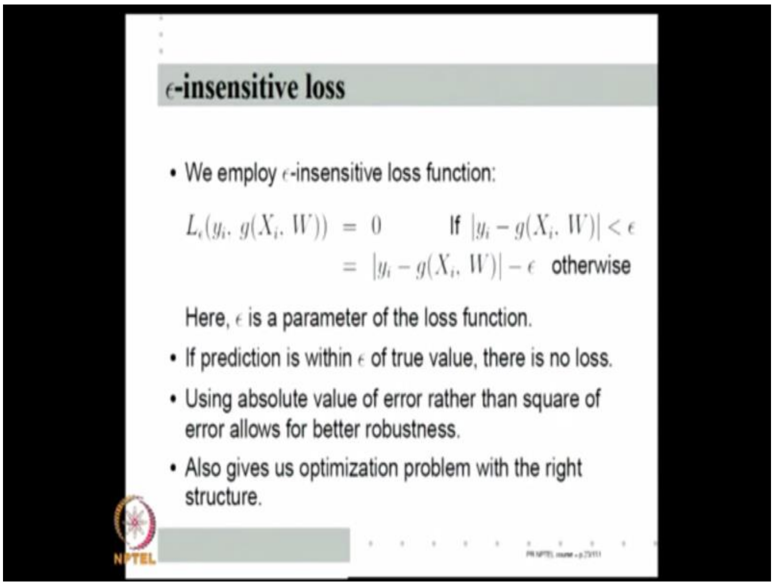
So, once again it the same kind of problem as in the classification case. So, I may have a hundred dimensional space in each I want to learn quadratic functions but, if I want to learn quadratic function by explicitly specifying phi 1 phi 2 phi m prime I will have ten thousand dimensional linear least square regression problem we do not want to do that we want to use a Kernel function. So, that we do the simplicity we do not even have to precisely specify the mapping phi we only specific the function. So, we want to formulate the problem such a way that just like in the classification case we will use the Kernel Trick. So, that we do not have to actually get into the m prime dimensional space we do not have to suffer the computational complexity of calculating this phi's and solving the linear least square problem in the m prime dimensional space.

(Refer Slide Time: 08:17)



How do we do this in any general regression problem as you already seen like in classification we essentially minimize m empirical loss something like the empirical loss. So, if I am learning a model g X comma W. So, L of y i comma g X i comma W is the loss. So, given an X i I would say g X i comma W the prediction that is given to me in the training data as y i. So, L of y i comma g X i W is the loss. So, I summed over to all i that is the empirical risk minimization of course, I can put a one by n in front but, that does not makes no difference. So, we have been doing. So, far by choosing and appropriate loss function we essentially minimize an empirical list correct this. So, what we are going to do as we already discuss this in last class we are going to use a special loss function that allows us to use the Kernel Trick.

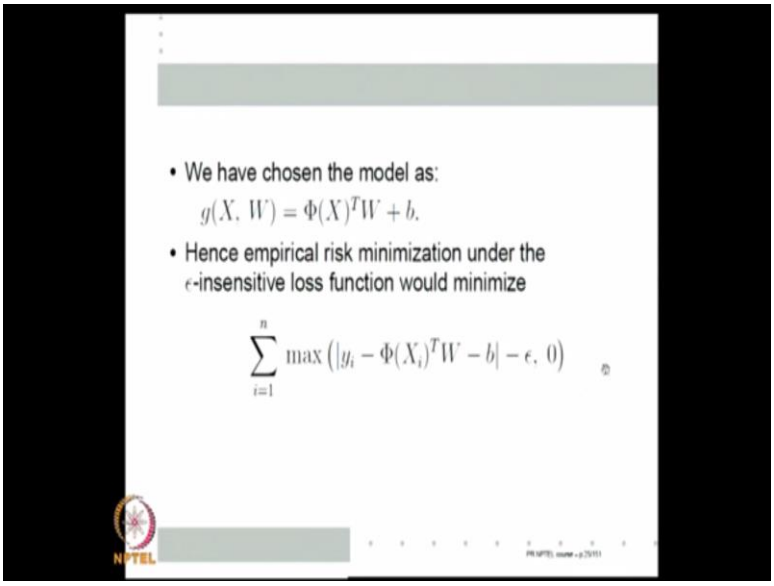
(Refer Slide Time: 09:19)




What is a special loss function is the epsilon insensitive loss function we seen this earlier when we did the formulation of risk minimization empirical risk minimization and we discussed loss functions many different loss functions are listed and this is one of them and also briefly discussed it last class anyway. So, on an example X i y i for the moral g X comma W the loss will be 0 if y i minus g X i comma W absolute value less than epsilon otherwise it is absolute value minus subscript what is that mean if the prediction is within epsilon of the true value my prediction is g X i comma W y i is the true value on the example. So, if prediction is within epsilon of the true value than there is no loss epsilon of course, a parameter of the loss function that is why I put a epsilon as a subscript here.

So, given as specific value of epsilon if the prediction is within epsilon of the true value there is no loss otherwise the absolute value of the difference on over and above epsilon is the loss. Using absolute value of error rather than square of error earlier we just we are not using any epsilon we just using square of the error and minimizing it using absolute value rather than square of the error allows a better robustness. In the sense a few out layers do not give me do not influence the final fit to much but, more importantly the reason why you use this formula is that this particular loss function when we want to do empirical risk minimization of this class function that gives us optimization problem which has the structure for us the structure is one where I can employ the kernel function alright.

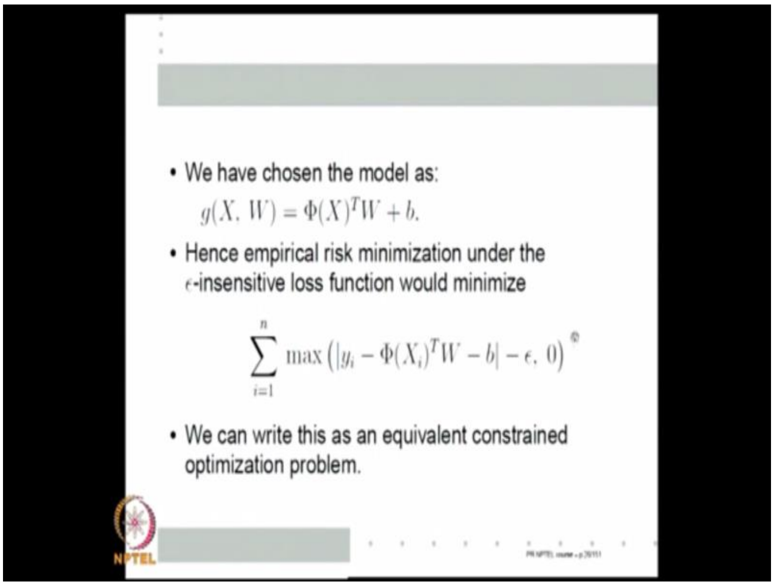
(Refer Slide Time: 11:14)




So, what is it I have to do? this is my model g X comma W phi X transpose W plus b once again W and b both are included in this W and left hand side essentially I have to learn W on b. By minimizing the empirical risk. So, my empirical risk is for a for each example X i y i my prediction is phi X i transpose W plus b the two prediction is y i. So, I take the difference between them absolute value the difference minus epsilon if that quantity is greater than 0 that is the loss is that quantity less than 0 loss is 0. So, this is my empirical risk.

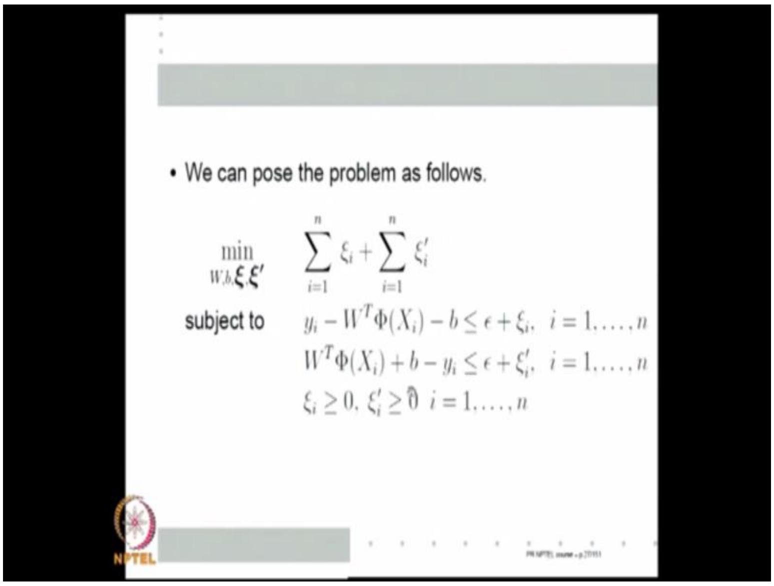
That is my epsilon insensitive loss function is the difference is less than epsilon the loss is 0 otherwise loss is this minus epsilon. So, loss is actually this minus epsilon comma 0 whichever is maximum.

(Refer Slide Time: 12:03)




So, that is what i have to maximize I can put one by n to make it actually empirical risk I have to put one by n but, that does not really make any difference and I am minimizing this. So, this what I want to minimize. What we are going to now do is we will write this unconstrained minimization into an equivalent constrained optimization problem because that will allow us to get into the structure we want. So, this is what I want to minimize I just remember this. So, basically given a W on b ideally what I want either y i minus phi X i transpose W plus b less than epsilon I mean which or phi X i transpose W plus b minus phi i less than epsilon whichever is positive actually I want both of them to be less than epsilon because one of them will be negative.

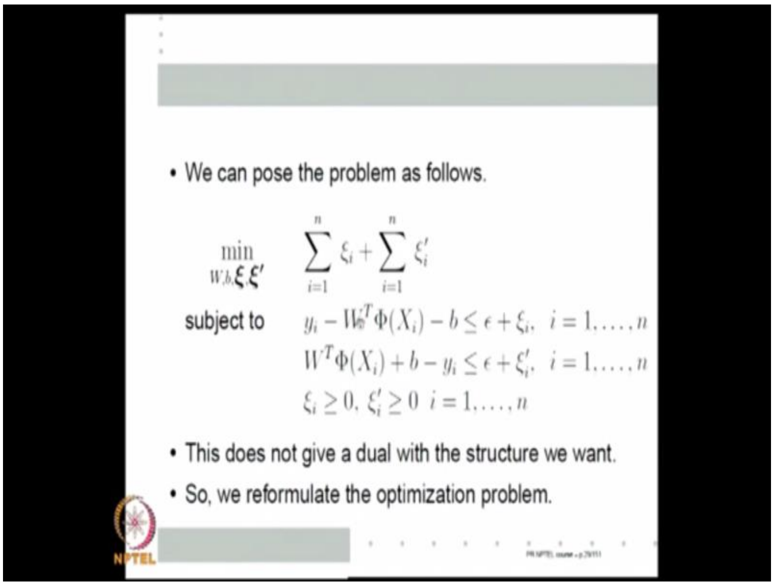
(Refer Slide Time: 12:57)




So, thus what have we write. So, the absolute value of this difference less than epsilon can be written as y i minus W transpose phi X i minus b I wanted to be less than epsilon I want W transpose phi X i plus b minus phi i also to be less than epsilon. If one of them will be negative. So, only one of these constraints is really useful if that is less than epsilon I am done if is not less than epsilon I am using a slack variable xi th. So, I need 2 level sets as likewise xi and xi prime that tells me by how much the difference between y i W transpose phi X i plus b differs from epsilon. Note that, for any given i only one of xi or xi prime is greater than 0 both cannot be greater than 0 because one of these left had sides is negative it really does not matter which one so but, one of them is the operational 1 for any given i and that xi prime tells me that xi or xi prime gives me the actual absolute value of y i minus W transpose phi X i minus b minus epsilon.

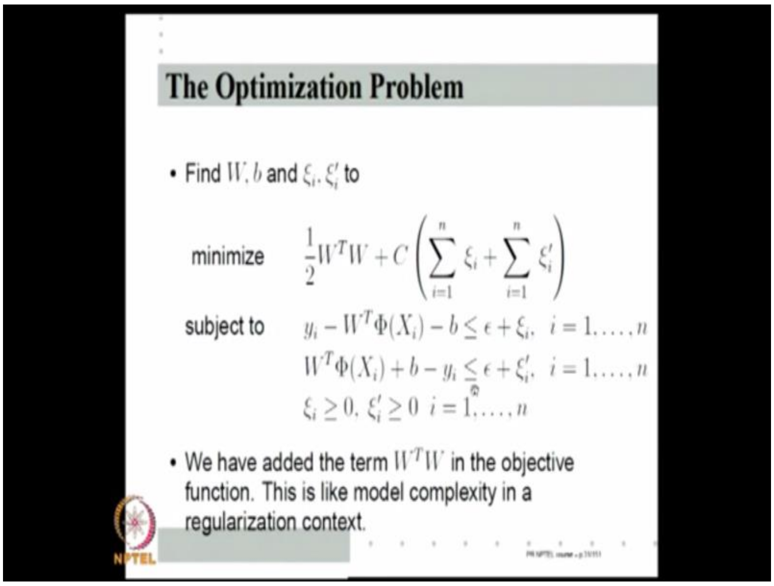
So, essentially what is it saying is saying xi one of xi or xi prime will be greater than or equal to absolute value of y i minus W transpose phi X i minus b minus epsilon and is also greater than equal to 0. So, if I just add this xi's this xi and xi prime is nothing but, max of this minus epsilon. So, I can always say such an unconstrained maximization problem as a constrained unconstrained minimization problem as a constrained minimization problem of this. So, of course, I am trying to find W b xi and xi prime to minimize this while satisfying all this. This is nice problem but, which exactly is same as minimizing empirical risk as easy to see.

(Refer Slide Time: 14:57)




Minimizing this empirical risk is equivalent to solving this constrained optimization problem. So, the problem is nice, but I have we have a problem we have a problem in the sense this problem is nice but, it does not solve our purpose this does not give a dual with the structure we want. So, its slightly reformulate this what do we do keep the constrained same.

(Refer Slide Time: 15:15)

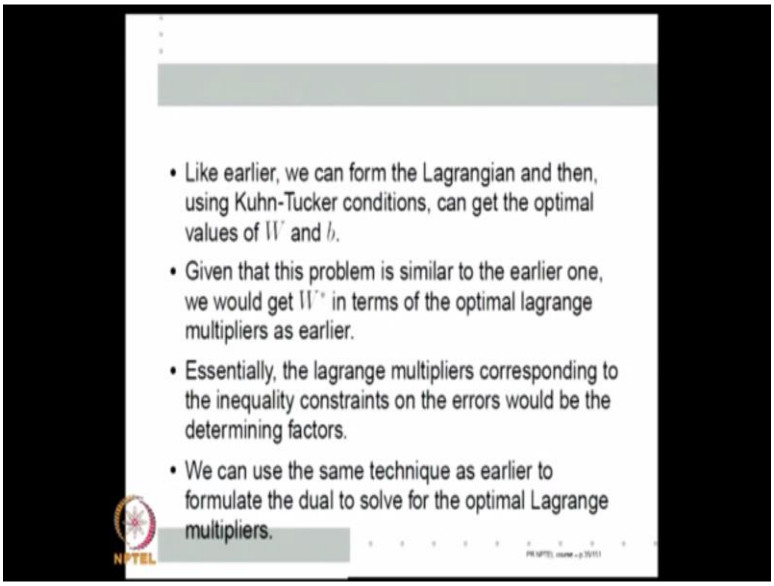



But, add a half W transpose W just like. So, that it looks like you know what we solve for the classification why should I had this half W transpose W well we will come back to that little later in today's class I will explain you why we want to add half W transpose W now let us say. So, if I had only this term the constraints and this term than is exactly like minimizing the empirical risk but, I am adding this. So, what we can think of this is with along with these constraints, this is the empirical risk. So, this in some senses like a model complexity.

So, this is the data error as you have already seen this term captures the empirical risk under the epsilon insensitive loss function. So, this is like some term which is modelled complexity. So, I can think of this as a regularized risk minimization we already seen in linear lease squares regression that we want to we may want to put a regularization term. So, that we learn a smooth model instead of learning instead of completely putting all our attention in minimizing the data error that is minimizing the empirical risk and we only minimizing empirical risk we may not learn as nice a function as if we also pay a little attention to the complexity of the model there also we just mentioned that an often use model complexity term for linear models is the norm of W. So, the same norm of W we are using here as a regularization term as I promise to by before the end of today's class we will come back and ask why is this a good way to characterize the smoothness of the model where learning.

So, for now we simply say that earlier if I did not have that W transpose W as exactly equal to the empirical risk. So, in addition to minimizing empirical risk I add a model complexity term. So, that this problem now corresponds to a regularized empirical risk minimization.Now, we are in business this now is very easy to see except that instead of 1 xi I have xi and xi prime I essentially have similar kind of constrained and similar kind of problem structure as espion. So, I would expect similar kinds of things to happen.

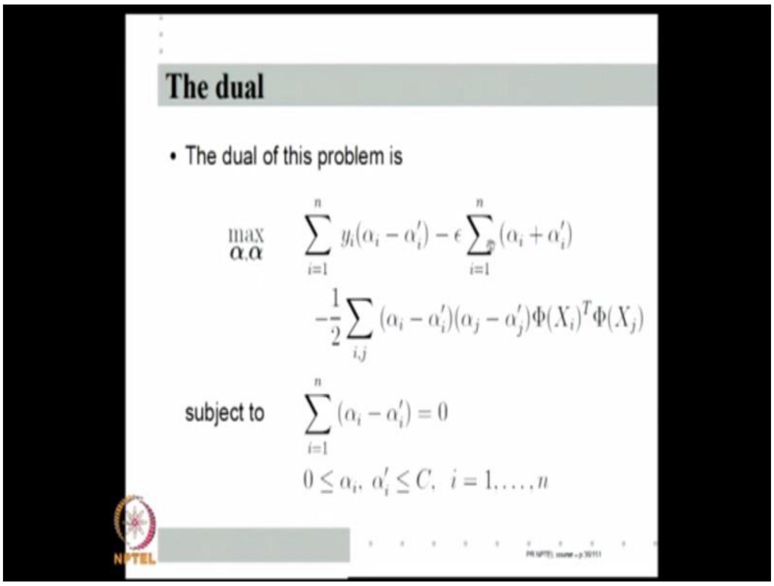
(Refer Slide Time: 17:33)




So, like earlier we can form the Lagrangian this is objective function this is constrained form the Lagrangian put in the Kuhn-Tucker conditions. That will give you the expressions for optimization values of W and b. Given this structural similarity we will expect the optimal values of W on b to be similar to the earlier expression. So, it means we can get W star and b star in terms of the optimal Lagrange multipliers what Lagrange multiplier essentially the Lagrange multipliers corresponding to the inequality constraints on the errors just like there also the Lagrange constraints and xi's are not important towards the constraints xi greater than equal to 0 Lagrange multiplier corresponding those constraints not important. Basically inequality constraints that on the errors in the data would be the determining factors and we can always use the same technique as earlier that is we formulate the dual to solve this.

We can once again this is a this is as you can see a quadratic cost function a convex cost function and linear constraints. So, I you can use the same technique as you use earlier for deriving the dual using the dual function.

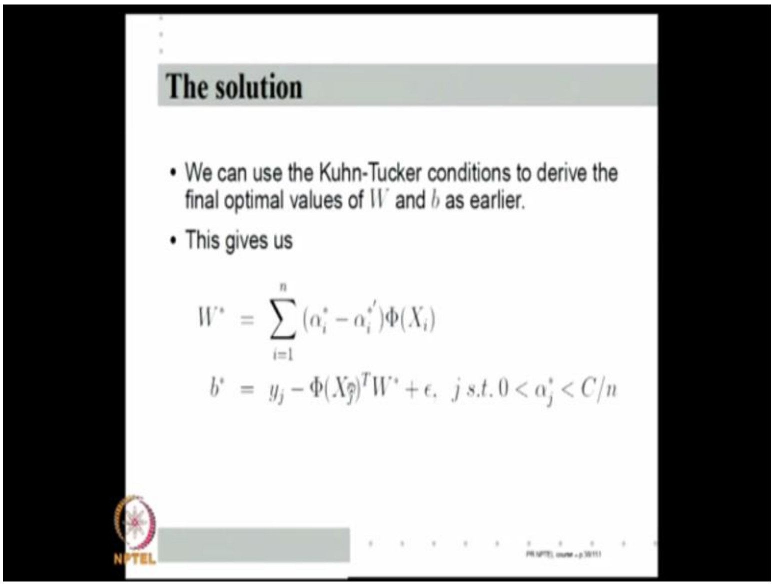
(Refer Slide Time: 18:53)




And if you do all that the dual transform to be this we will just omit the details look at the derivations quite similar. So, now we have two Lagrange multipliers alpha and alpha prime. So, basically there 4 constraints 1,2,3,4. So, the constraints for this lets say this as the these are the xi i constraints these are the xi i prime constraints. So, Lagrange multiplier corresponds with xi constraints are alpha i Lagrange multiplier correspond with xi prime constraints are alpha I prime. These constraints are not important just like in the SVM case Lagrange multiplier correspond to these constraints would not enter into the dual. So, dual is in terms of alpha and alpha prime, I have we have alpha i i is equal to 1 to n and alpha i prime i is equal to 1 to n those 2 n variables are the Lagrange multiplier corresponding to these 2 n constraints.

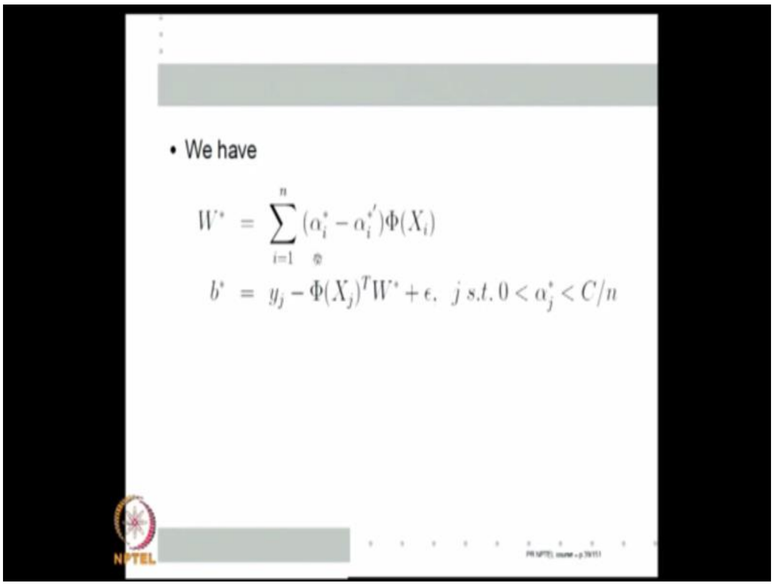
So, that turns out to be the objective function and these half turn out to be constraints once again like earlier the all the Lagrange multipliers have to between 0 and C and there is one equality constraints and then the objective function the dual is also quadratic in both alpha i and alpha i prime and the training data themselves appear only as inner products phi X i transpose phi X j. So, all the structure that we had for the SVM dual still is there the details do not really matters to us the actual terms at different here compare to the SVM but, the overall structure is same it's a quadratic cost function one in a one linear equality constraint bound constraints and variables and the bound is between 0 and C where C is the same penalty constraints and the training data appear only as inner products in the objective function phi X i transpose phi X j.

(Refer Slide Time: 20:54)




Using the Kuhn-Tucker conditions as earlier we can derive the optimal values of W and b and it trans out to be the same W star is sum all alpha i star minus alpha i star prime phi X i. So, as I said alpha i's are correspond with the first constraints alpha i prime say correspond second constraints there. So, this happens to be the W star and b star.

(Refer Slide Time: 21:18)

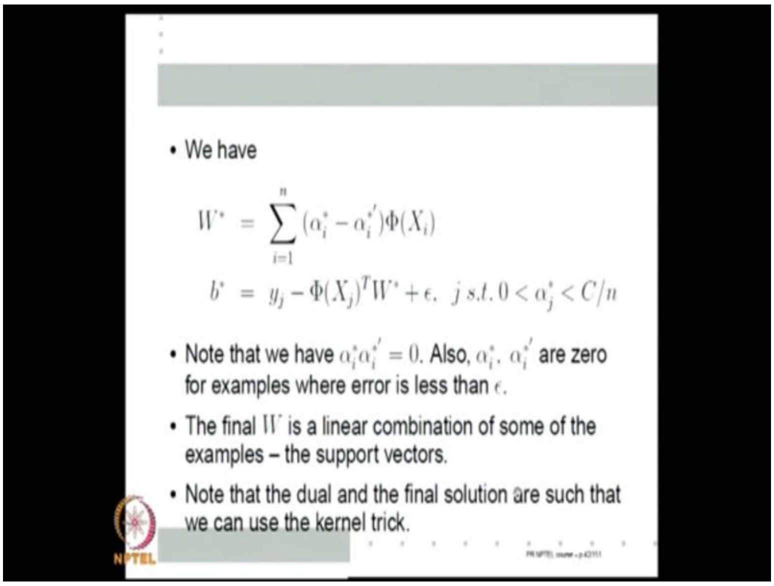



Let us look at this expression.

First let us note certain thing about this primal problem. As you already seen, only 1 of xi and xi prime is non0 the other because on the LHS one of the terms is negative. So, for any given i only if at all only 1 of this constraints can be X i only they will be only one xi i which strictly greater than 0 which means because Lagrange multipliers are alpha i here and alpha i prime here the complementary slackness shows us that both alpha i and alpha i prime cannot be simultaneously greater than 0 for them to be phi arithmetic greater than equal to greater than 0 both inequality should be satisfy by equality that is not possible. So, only one of alpha i and alpha i prime are would be greater than 1 the optimal at the optimality point and similarly, if y i suppose this is positive.

So, y i minus W transpose phi X i minus b if it is actually strictly less than epsilon than xi will be 0. So, once again even among that particular all i belong the alpha i is that only one of alpha i and alpha i prime are greater than 1 as a set for each i in addition not all alpha i's are alpha i prime will be if I look at all the strictly greater than 0 Lagrange multipliers they may not be enough them the most probably will not be enough them like the support vectors in the classification case basically only when this inequality has to be strictly satisfied only that is why. So, that xi i strictly greater than 0 only than the corresponding alpha i will be positive.

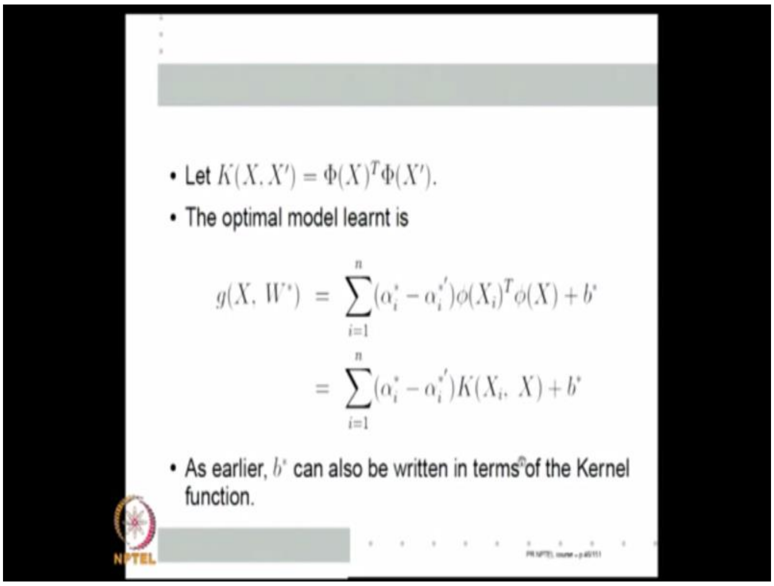
(Refer Slide Time: 20:19)




So, which means we know by looking at the primal problem that only 1 of alpha i and alpha i star prime is non0 and both of them will be 0 for example, where error is strictly less than epsilon. Now, when I with that in my mind if I look at the W star essentially W star is nothing but, a linear combination of some of your data phi X i and the coefficients are the Lagrange multipliers because in this term is either alpha i star or is minus alpha i star prime. So, coefficient are simply Lagrange multipliers. So, the final W is a linear combination of some of the examples what which are these examples the support vectors where ever the corresponding Lagrange multipliers strictly positive and the dual and the final solution both are such that the Kernel Trick is visible. The dual as we already seen the data phi X i if we are only as inner products.

So, I can use the kernel there and the W star is a linear combination of phi X i. So, once again I can use the Kernel Trick and the b star comes as phi X transpose W star. So, once again, I can use the Kernel Trick.

(Refer Slide Time: 24:40)



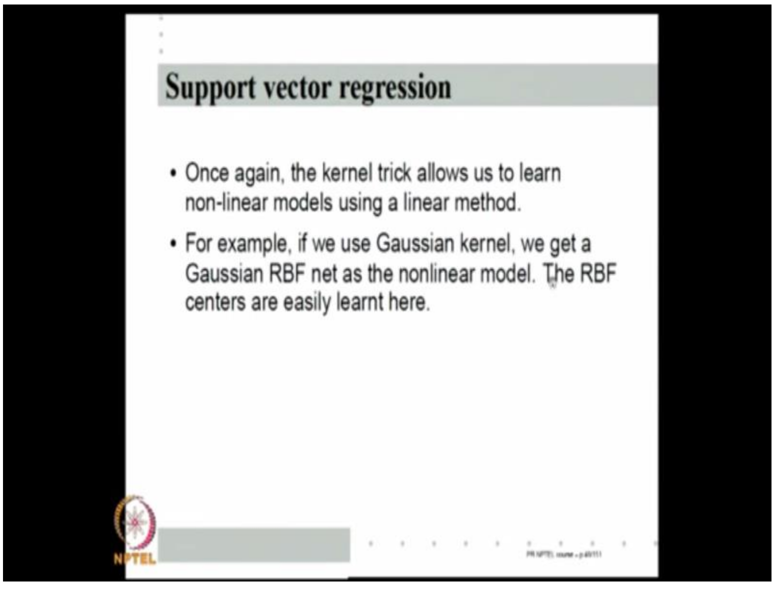

Specifically, suppose we have a kernel function than what is mu optimal model g X comma W star is all this is my expression for W. So, W transpose phi X plus b star. So, this can be canalized this can be does K X i X.

And b star and b star can also be canalized because b star is phi X j transpose W star. W star is linear in phi X. So, this term also involves only phi X i transpose phi X i. So, once again this can be canalized. So, this can be canalized and as earlier b star can also be written in terms of the kernel function. So, the way we formulated the regression problem let us go back to our formulation.

This is how we formulate the regression problem. We put in this extra regularization context and otherwise it is like the SVM formulation this is actually minimization of empirical risk under epsilon in sensitive loss function along with a regularization context. And for that problem that happens to be the dual dual has the vary large structure.

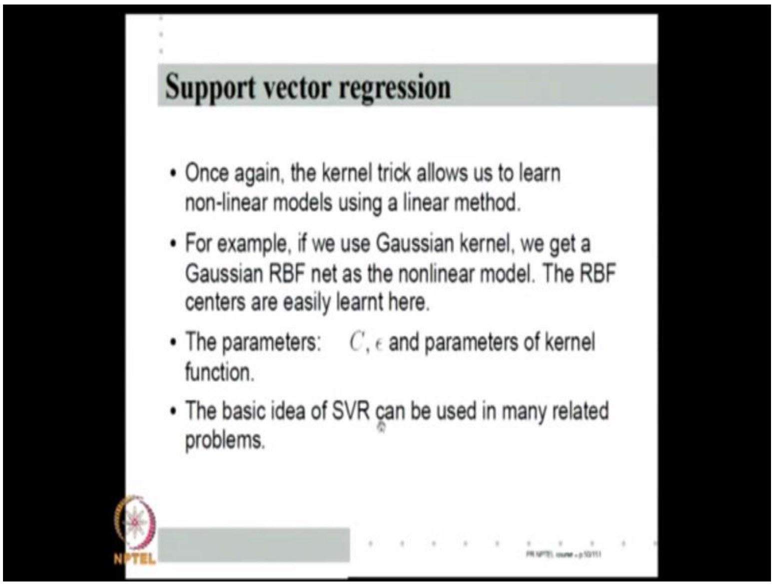
And with that dual I get my solution in the way. So, that the solution allows me the Kernel Trick specifically, I can write my final function like this where b star can also be obtained through the kernels. So, essentially now I do not have to calculate phi's, I only stay in the m dimensional space of the original space of the X i's and still learn nonlinear regression functions.

(Refer Slide Time: 26:42)



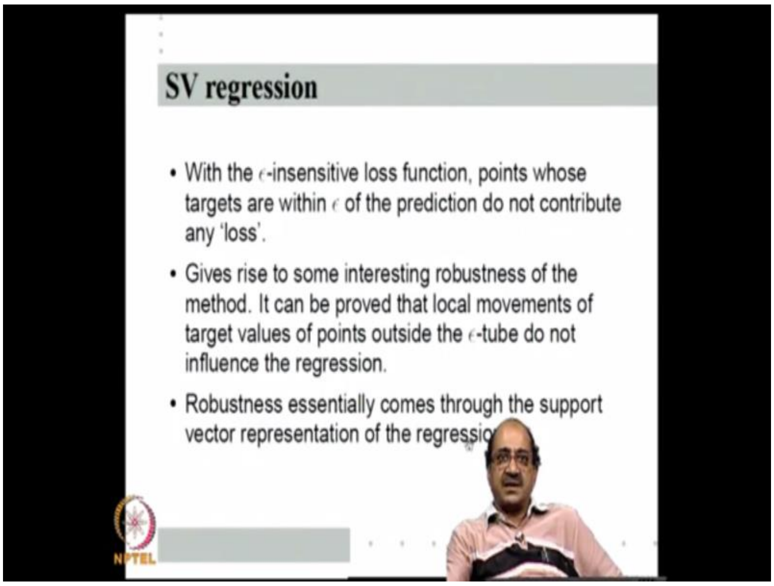

So, once again the Kernel Trick allows us to learn nonlinear models using a linear method for example, if we use Gaussian kernels we get a Gaussian RBF net. So, if this is Gaussian K exponential K X i X is exponential minus norm of X i minus X whole square than this is nothing but, linear combination of Gaussian real basis functions. So, this is actually the output of an RBF network the stand RBF network Gaussian RBF network with a Gaussian real base function hidden notes and linear output note.

(Refer Slide Time: 27:23)




So, the kernels automatically give you this. So, if you use a Gaussian kernel we get a Gaussian RBF net just like in the SVM case and just like in SVM case RBF centers are easily learnt here they happened to be the support vectors. There are parameters, of course, any algorithm as parameters, we have the penalty contrast C which is like the regularization constraints epsilon where choose the epsilon in the epsilon insensitive loss function and any parameter the kernel function for example, if I am using a Gaussian kernel I may use a sigma. So, these parameters have to be choosing properly, but, if I choose the parameters, I get fairly good performance and this basic idea of supported to regression is also used in many related problems.

(Refer Slide Time: 28:13)

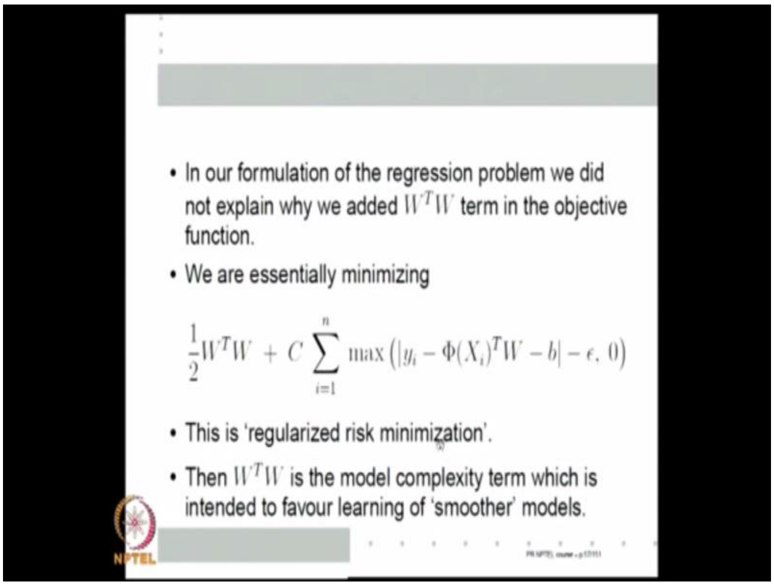



Basically, what happened was by using the epsilon insensitive loss function point whose targets are within epsilon of the prediction do not contribute to loss that as what as given rise to the nice structure in the problem and it also gives us some interesting like this the support vectors as we as we know in the general classification problem the support vectors are very good by product as we said they represent all the patterns which are closest to the mount debutante clauses. So, and once we solve a problem and get all the support vectors if am I training data set I remove all the things for the support vector i will still get the same SVM because those are the closest to the training class boundary.

So, if I can separate them we will anyway separate the rest of them. So, the SVM on the entire training data will be same as SVM reruns on the support vectors. So, essentially same way the support vectors represent the regression function two for example, the substant because the epsilon loss function will can show that local movement of target values of target values of points outside the epsilon tube that is the epsilon tube is like the W transpose X plus b is equal to plus one or minus one through two parallel hyper planes in our classifier here given our predictor function W transpose phi X any prediction that within epsilon of this does not contribute any loss.

So, I can put an epsilon tube around this prediction function and we can show that if you move targets outside this that does not much influence a regression. So, robustness comes through this support vector representation of the regression.

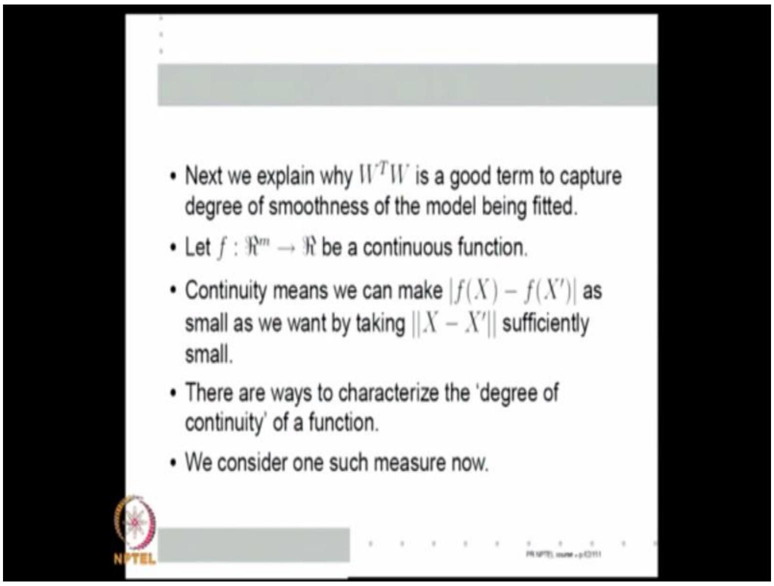
(Refer Slide Time: 30:13)




Now, let us get back to our regularization term as I promise you I said I will tell you why it is All to add W transpose W, we chosen an epsilon insensitive loss function we want to minimize empirical insensitive loss function and the rest of the formulation is fine except the term W transpose W and we said it is like a model complexity term. So, basically this is my empirical risk i is equal to 1 to n max y i minus phi X i transpose W minus b minus epsilon 0.

This is what is captured by summation xi i plus summation xi i prime subject to those constraints in addition I added this. So, this is essentially what I am minimizing. So, this is the empirical risk this the model complexity and that is what I minimize. So, this is regularized a risk minimization the only thing we have to show is why is this a good regularization term. So, W transpose W is a model complexity term which is intended to favor learning of smooth models why do we use regularization why do we use a model complex term just because I am minimizing my data error is not enough as we discuss when we discuss statistical learning theory we need to learns simpler models the simpler the model the better it is. So, essentially we want more smoother models. So, W the W transpose W term is a complexity term which intended to favor smoother model.

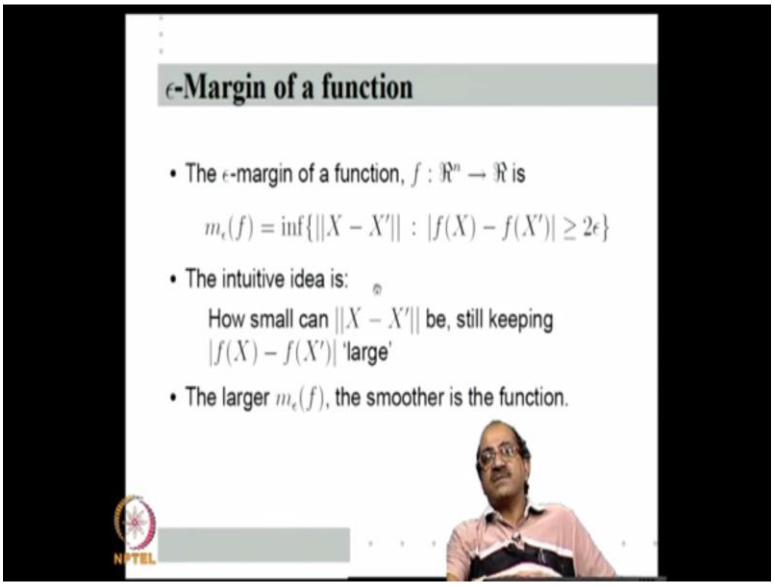
(Refer Slide Time: 31:40)




So, we have to ask why is this a good term to capture degree of smoothness of a function being fitted smoothness is being nicely continues there many continuities a continues function is kind of a continuity that bind the property a function either continues or not continues but, all continues functions are not same or a say f R m to R is a continues function what does continuity mean I can make f X minus f X prime the different between f X and f X prime as small as a one by taking the norm of the different X minus X prime sufficiently small but, if f is even if f is continues if is varying rapidly than to make the difference between f X and f X prime small by the same epsilon I may have to make the different between the X and X prime very very small.

So, for a given difference between f X and f X prime how small should I have to make X minus X prime is a kind of one kind of measure to say how fastly or slowly wearing the function is and which is one measure of how smooth the function is. So, there are ways to characterize what can be called degree of continuity a degree of smoothness a function were going to look at just one such measure we just look at one such measure how to characterize the degree of continuity function.

(Refer Slide Time: 33:06)

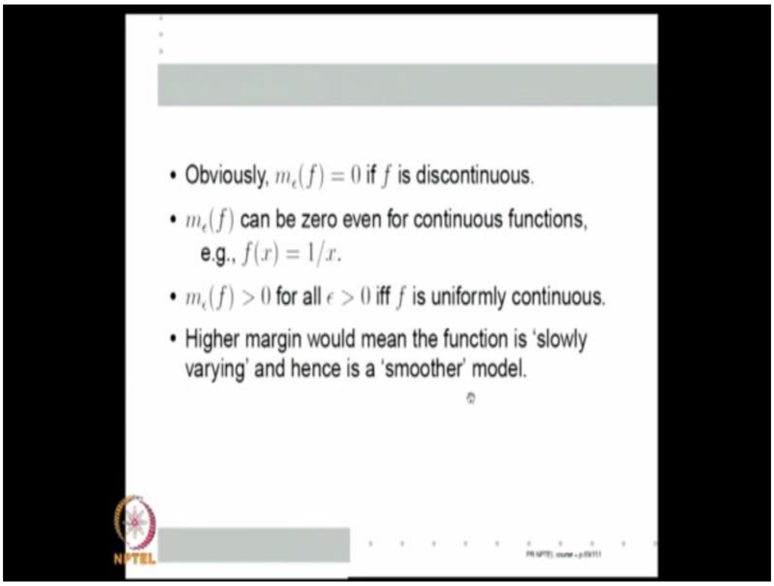



The measure we look at is what is known as an epsilon margin of a function given a function f R n into R and some epsilon greater than 0 we for that epsilon the epsilon margin of the function we generate is m epsilon of f is infimum over all the numbers norm X minus X prime such that f X minus f X prime is greater than 2 epsilon let say given an epsilon I want to keep f X minus f X prime at least two epsilon where f X minus f X prime has to be large. So, do say f X minus f X prime has to be greater than 2 epsilon I am asking of all X minus X prime that will allow f X minus f X prime to be greater than 2 epsilon what is this smallest of that. So, how small X and X and how close X and X prime be still keeping f X minus f X prime large.

So, if m epsilon f is small that means X and X prime even very small difference between X and X prime can make f X minus f X prime large than is not very good on the other hand if m epsilon is large than its good the larger m epsilon f the smoothly the function m epsilon f the epsilon margin being large means what the smallest of norm X minus X prime which ensures f X minus f X prime is greater than 2 epsilon is still large. So, even if I want to make f X minus f X prime more than 2 epsilon I can still keep I do not have to keep my X and X prime very close. So, that is what the basic idea here is. So, the larger the epsilon margin the smoother is the function.

Now, it is easy to see that for suppose f is discontinuous if f is discontinuous than epsilon margin will be 0 because around the discontinuity at the point of discontinuity, no matter how small I make X minus X prime this can still be greater than 2 epsilon because f X minus f X prime is strictly greater than 0. So, they will be an epsilon such that this always greater than 2 epsilon no matter how small I make X minus X prime.

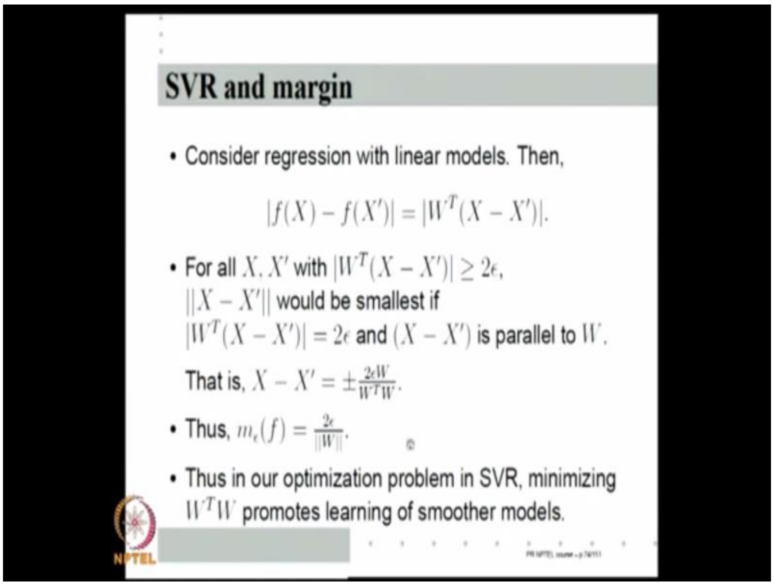
(Refer Slide Time: 35:26)




So, this is easy to see that the epsilon margin is 0 the function discontinuous but, interesting the function the epsilon margin can be 0 even if the function continuous if you look at a function one-xth obviously is continuous but, is x comes closer to 0 it is varying fast and faster because its rate of change is one by of the order one by x square. So, x becomes closer to 0 its varying fast and faster. So, its not a really smooth function because its varies very very fast. So, they can be a continuous function like this for which also epsilon margin is 0. One can show that for every epsilon greater than 0 if the epsilon margin has to be greater than 0 then f is to be uniformly continuous do not worry if you do not know what uniform continuous we really do not need it.

Essentially, higher the margin the smoother the function is that the function is slowly varying and hence it is smoother that is basically what this is. We have defined this as the epsilon margin and is easy to see that higher the epsilon margin the smoother the function.

(Refer Slide Time: 36:39)




So, now let us calculate the epsilon margin for linear models. So, we have a linear model than f X minus f X prime because f X is W transpose X f X minus f X prime even if f X is W transpose X plus b than f X minus f X prime will be W transpose X minus X prime. So, to find epsilon margin whatever to do given X and X prime such that w transpose X minus X prime is greater than 2 epsilon find all X and X prime which satisfy W transpose into X minus X minus greater than 2 epsilon among all such spites calculate norm X minus X prime and ask for which pair is norm X minus X prime smallest that is what is infimum in the definition?

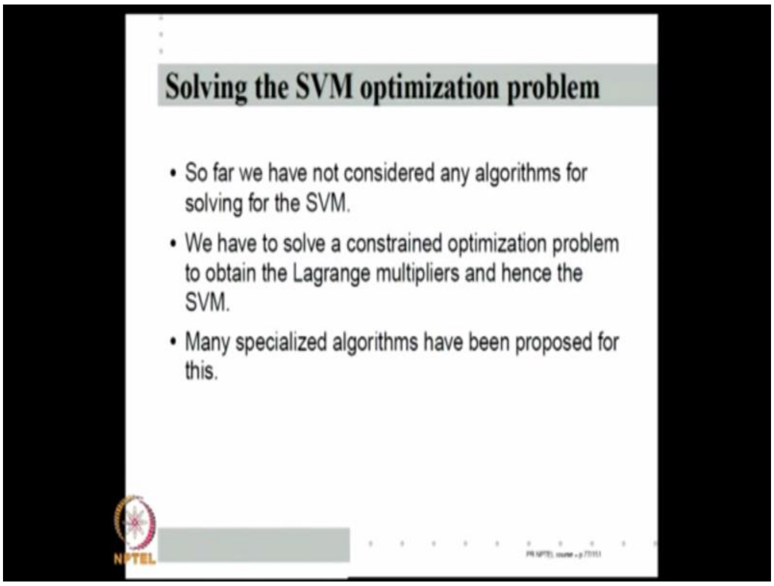
So, we are asking when would norm of X minus X prime be smallest if we have to make W transpose into X minus X prime greater than 2 epsilon firstly the concern is greater than 2 epsilon because I want norm of X minus X prime to be smallest and this inner product is nothing but, norm W norm X minus X prime into some cost of angel. So, do say its better to take it to be 2 epsilon. So, W transpose X minus X prime equal to 2 epsilon would be satisfied by X and X prime which have smaller norms than W transpose X minus X prime greater than 2 epsilon and among all X and X prime that will satisfy W transpose X minus X prime is equal to 2 epsilon.

The one that we have the smallest norm if that pair such that the vector X minus X prime is parallel to W, here we know this a norm W norm X minus X prime into some factor which is less than 1 that factor becomes one of the two vectors are parallel. So, 2 vectors are not parallel because that factor is less than one corresponding a norm has to be larger. So, that is still 2 epsilon. So, the smallest the pair X and X prime with the smallest norm X minus X prime which still satisfies this is such the W transpose X minus X prime is equal to 2 epsilon and the vector X minus X prime is parallel to W what is that mean X minus X prime should be parallel to W. So, it should be k times W and W terms X minus X prime should be 2 epsilon.

So, if is k times w k time W transpose W should be equal to 2 epsilon there is an absolute value. So, k can be only plus minus 2 epsilon by W transpose W. So, X minus X prime is either plus 2 epsilon by W transpose W into W or minus of that. So, my epsilon margin is nothing but, the dorm of this smallest X minus X prime if I take norm of this that will be 2 epsilon by W transpose W into root of W transpose W. So, that will give me 2 epsilon by norm W. So, for linear models the epsilon margin is nothing but, 2 epsilon by norm W this is what we call the margin of the hyper plane the classifier.

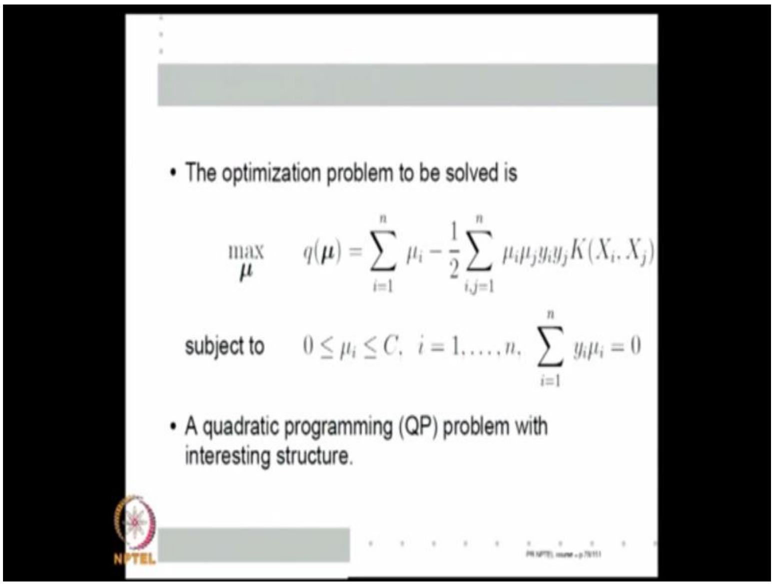
So, essentially when I added a term to minimize norm W I am promoting learning of the smoother. So, for all linear models because the epsilon margin of the function is inversely proportional to norm W or norm W square is a good regulation term to add we are done it regularized least squares also earlier without explaining here is the reason why norm W square is a good regularization. So, this completes our discussion on regression.

(Refer Slide Time: 40:18)



So, far we looked at support vector machines for classification regression but, than both cases we just said solve the dual we never said how do I solve the dual dual is a constrained optimization problem were not really given any algorithm for solving the dual we have to solve a constrained optimization problem to obtained the lagrange multipliers and once you get the Lagrange multipliers we can get the SVM. So, there is one optimization problem namely the dual which is a quadratic programming problem which needs to be solve we are not looked at any algorithm. So, far for this dual is many some interesting structure. So, that many specialized algorithm proposed for this.

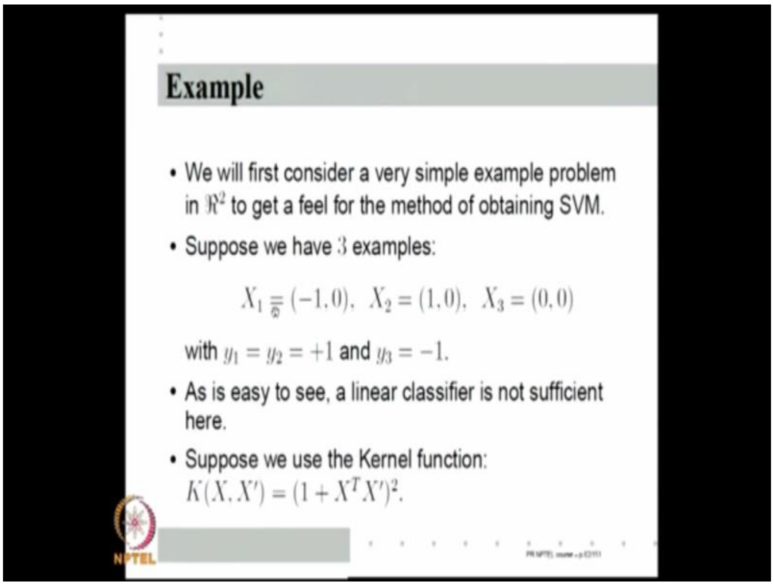
(Refer Slide Time: 41:00)




So, this is the dual. Let us remember this is the dual we want to maximize over mu some objective function which I called q mu which is equal to summation i is equal to one to n mu i minus half mu i mu j y i y j K X i X j this like a quadratic form of the K X i X j's. So, if I got X one to X n as the data vectors and I make a matrix whose i j'th element is K X a X j mu i mu j K X i X j is nothing but, the quadratic form of the matrix with respect to the vectors mu the vector mu and we have to also of course, multiply this matrix by y i y j the i j'th element has to multiply by either plus one minus one remember y i y i are plus one minus one. So, the product is either plus or minus depending on whether both X i and X j in the same class or different classes.

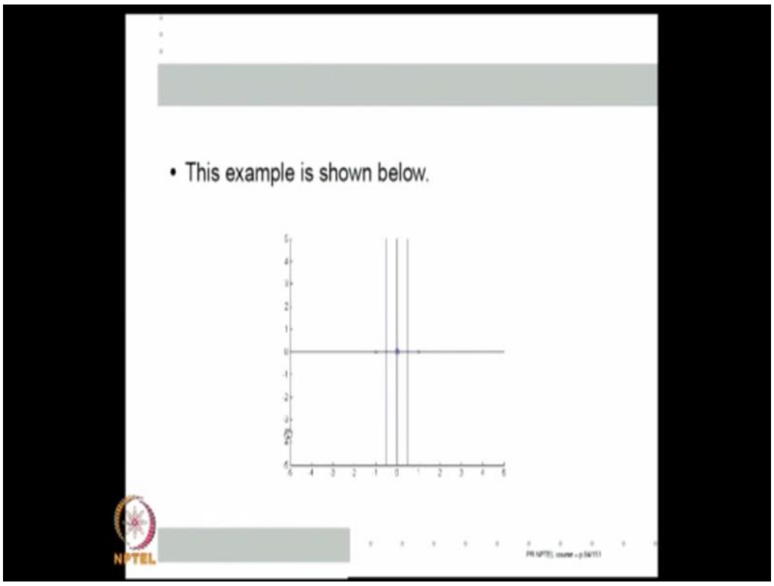
So, this essentially a quadratic form. So, this a quadratic cost function subject to one linear inequality constraints and bound constraints and variables. So, quadratic programming problem with very interesting structure and hence there are many special algorithms. So, we would not consider any algorithm details I will just mention where this specialized algorithms comes from but, before going there we will solve one simple problem just to get a feel for how one solves for an SVM.

(Refer Slide Time: 42:17)




So, we will first consider a very simple example in r 2 let us say only 3 examples in R 2 minus 1,0,1,0,0,0 minus 1,0 and 1,0 are in class plus 1,0,0 is in class minus matter of fact they are all of them are y component 0's is actually one dimensional problem. So, we call it 2 dimensional problem. So, is a very simple thing for all that is simple a linear classifier still not sufficient.

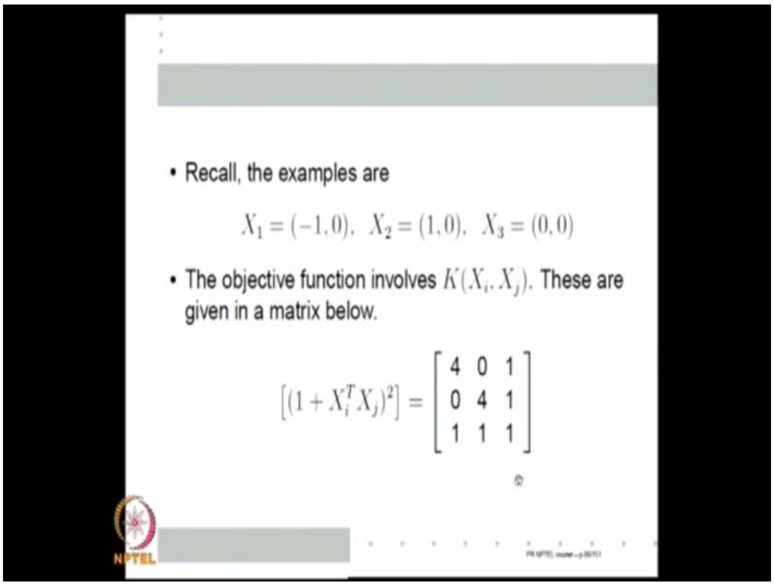
(Refer Slide Time: 42:48)




Because, see this my problem. So, this is 1 class the 2 dotes or 2 stars are at the same class and the center circle is a different class. So, this is not linearly separable there in a line as you already seen 3 points which are collinear cannot be shattered by a hyperplane is the middle point is one class so, I have chosen that so, the middle point is 1 class the outer 2 other 2 classes. So, for all that is a very simple three point thing it is still not solvable by linear classifier.

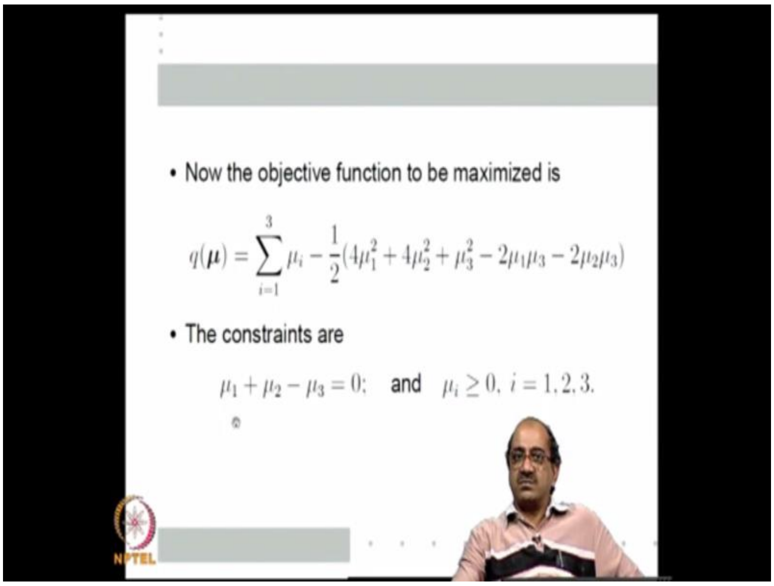
So, let us say we chose a kernel function of this kind which essentially give us second degree polynomial in the original space so, this is my thing for example, I know one classifier is this I just put now the origin at that 0,0,3 in just to once again see. So, basically if I have some line parallel to y axis on either side. So, the thing is in between the lines is one class outside the lines other class so, it is like a pair of lines kind of a classifier. So, essentially if I square the x coordinate than this 2 will become same. So, I can actually get a threshold on the x coordinate square as a classifier. So, its a quadratic classifier but, of course, we know this but, my SVM has to automatically fine instead of I just chose a kernel function and go ahead and solve for the SVM.

(Refer Slide Time: 44:20)




So, how do I solve for the SVM and it know the to write the dual as K X i X j n a. So, let us first calculate K X i X j K X i X j is 1 plus X i transpose X j whole square we can write that as a matrix notation these are the X i's. So, 1 plus X 1 transpose X 1 whole square X 1 transpose X 1 is 1. So, 1 plus 1 is 2 square is 4 X 1 transpose X 2 is 0 sorry is minus 1. So, minus 1 plus 1 is 0. So, K 1 plus X 1 transpose X 2 is 0 X 1 transpose X 3 is 1,0. So, this becomes 1 and similarly, for X 2 transpose X 2 is once again 1. So, the Kernel function is 4 and X 2 transpose X 3,0. So, the Kernel function is 1 and finally, X j transpose X is 0. So, this Kernel function is 1 by this obviously this is a symmetric matrix.

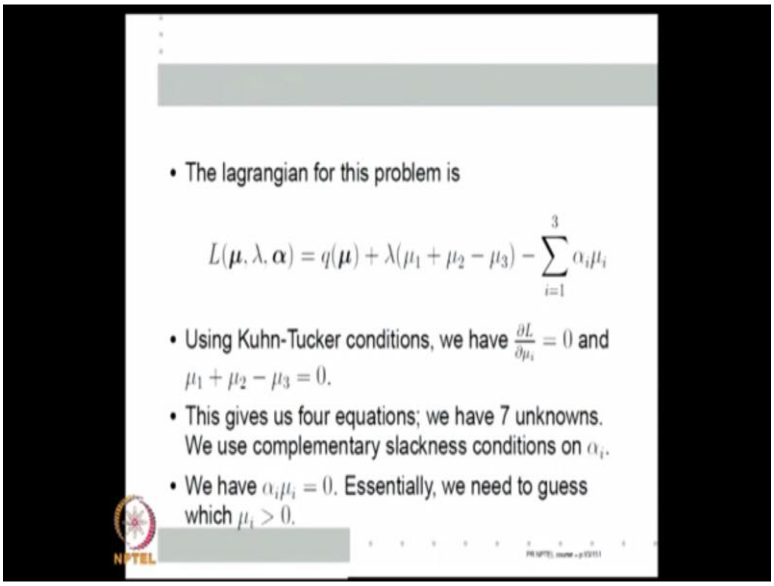
(Refer Slide Time: 45:24)




So, what will be mu dual now? My dual is q mu of the objective function is i is equal to 1 3 mu i plus the quadratic form on this. So, 4 mu 1 square plus 4 mu 2 square plus mu 3 square than I have all the class terms mu 1 mu 2 has coefficient 0 of course, a way mu 1 mu 3 is 2 mu 2 mu 3 is also 2 but, both mu 1 mu 3 and mu 2 mu 3 will be minus terms because 1 and 3 are a different classes and 2 and 3 also in different classes. So, it will be 4 mu 1 square 4 mu 2 square mu 3 square minus 2 mu 1 mu 3 minus 2 mu 2 mu 3 this is my objective function. What am my constraints? mu i y i is equal to 0. So, that is mu 1 plus mu 2 minus mu 3 mu 1 and mu 2 are in class plus x y 1 and y 2 is plus 1 y 3 is minus 1. So, mu 1 plus mu 2 minus mu 3 is equal to 0 and mu's are positive this is my problem to be solve. So, let us form.

So, this is the dual but, does not really matter to us this is just a constraint optimization problem with this at the objective function these are the constraints. So, I can once again use Kuhn-Tucker conditions. So, I have to form the Lagrangian. Lagrangian is this q mu plus Lagrange multipliers when constraints allow one mag Lagrange multiplier for this constraint. Let us call that lambda. I have some 3 Lagrange multipliers I have to write these constraints as minus mu i less than equal to 0.

(Refer Slide Time: 46:54)

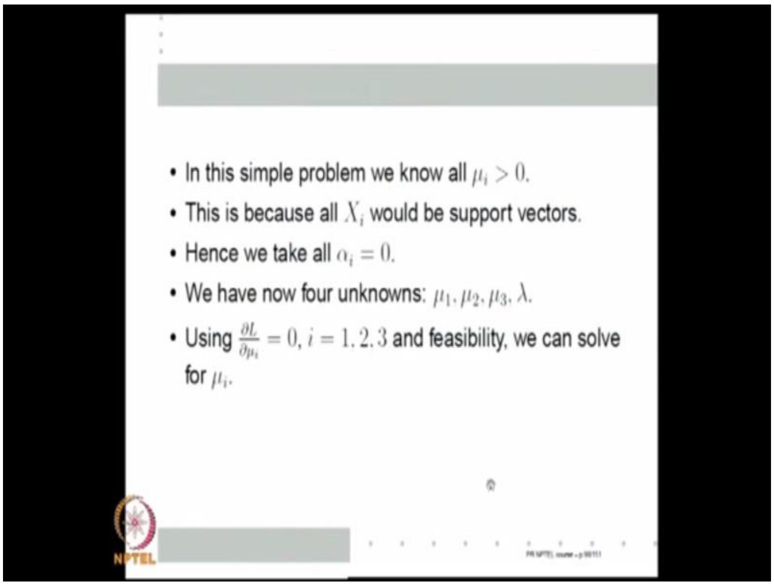



So, ultimately I get that as my Lagrangian q mu plus lambda into 1 constraint minus alpha i mu i where alpha i's are the 3 Lagrangians multipliers for these 3 constraints that is my Lagrangian. Now, to solve for solve this using Kuhn-Ttucker condition. what are the Kuhn-Tucker condition? I have my optimization variables are mu 1 mu 2 mu 3. So, I can take partial derivatives of l with respect to mu 1 mu 2 mu 3 is equal to 0 that gives me 3 equations I know mu's have to be feasible.

So, they have to satisfy all the constraints. So, mu 1 plus mu 2 minus mu 3 should be equal to 0 thats a constraints. So, that i get 4 so, this gives us 4 equations how many unknown I have mu 1 mu 2 mu 3 lambda and 3 alphas I got 7 unknowns. So, you need 3 more equations have somewhere where do we get the extra equations I have to use the inequality constraints and we can use the complementary slackness on alpha i's. What does complimentary slackness alpha? i say alpha i's correspond to the constraints mu i greater than equal to 0. So, the complimentary slackness tells us that alpha i mu i is equal to 0 which means if mu i is strictly greater than 0 than alpha i is 0 otherwise mu i equal to 0. So, mu i is equal to 0 i do not need to know mu i and if mu i strictly greater than 0 alpha i is equal to 0 i do not need to know alpha i.

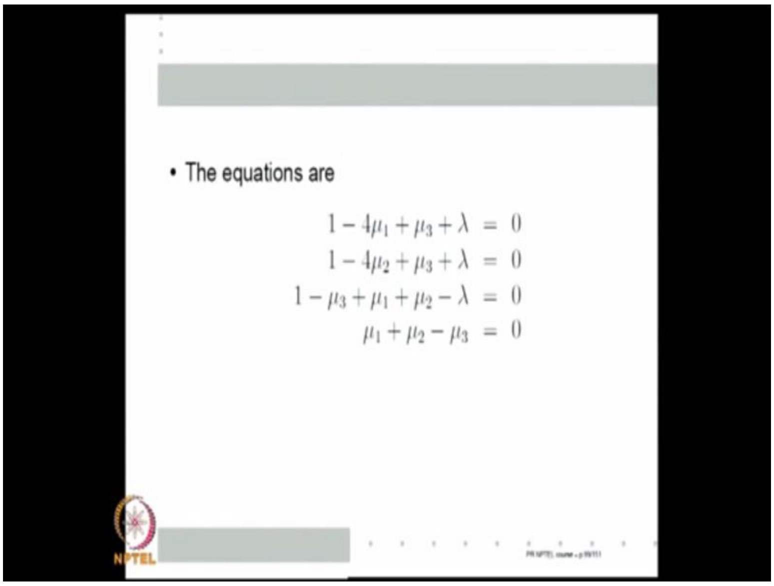
So, because that 3 complementary slackness is essentially they give me the remaining three equations essentially in the sense they will tell me which alpha i's are mu i's are 0 but, I really do not know which are 0 I have to actually in a general constrained optimization problem like this I have to make a guess work and which mu i's are greater than 0. So, they can take the corresponding alpha i to be 0 and solve this what 1 normally does if you are solving a constrained optimization problem in pen and paper is to make this various guesses like this and then see which guess gives rise to a consistent solution in SVM’s in general in this particular type problem but, in SVM’s in general this is very simple to make because we know that mu i greater than 0 has a physical interpretation a mu i is greater than 0 the corresponding X i is a support vector that means is closest to the separating bound. So, I can actually geometrically think of which excess have to at least in r 2 I can do this in this problem of course, we know all the 3 are support vectors.

(Refer Slide Time: 49:27)



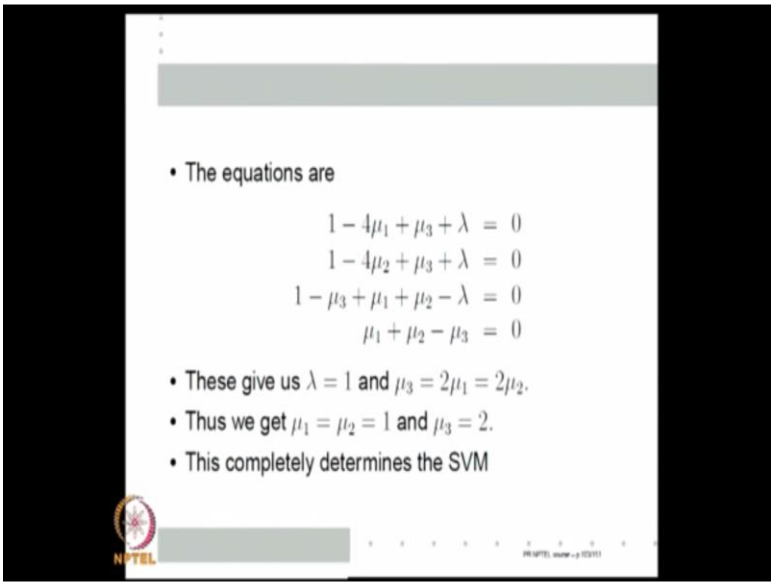

So, we know that all the mu i will be greater than 0. So, we can take all alpha i to be 0 that means we have only 4 unknowns and we got 4 equations. So, we can solve for it.

(Refer Slide Time: 49:42)




So, if I put this 4 equations down that will happened to be the 4 equations it easy enough to see. Essentially, I am differentiating this if I differentiate this could to mu see when differentiate q mu i get 1 from this because in y and a mu term from this the square and then the mu 2 or mu 3 term from the cross terms. And then, from here I will get a lambda term alphas are 0.

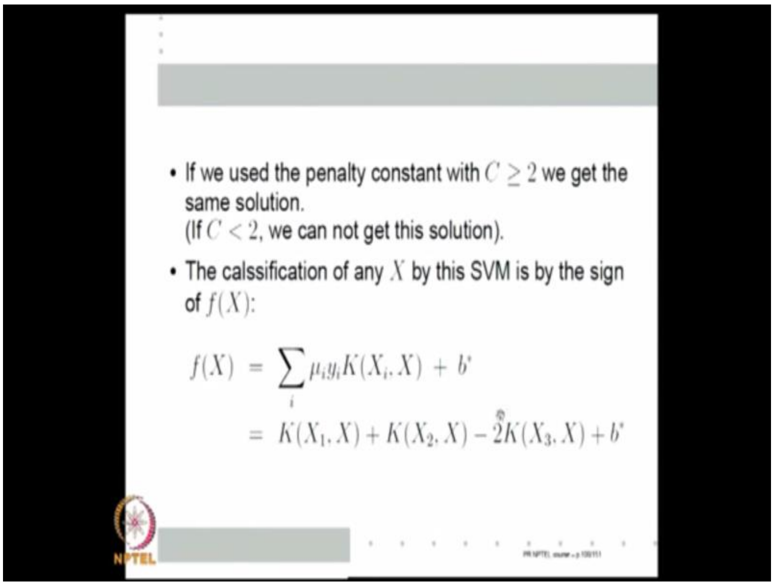
(Refer Slide Time: 50:23)




So, roughly one can see that that the general structure of the equations it that just happened because you can differentiate and see now 1 and this particular case is chosen.

So, the equations are very simple to solve we already can see that see mu 1 plus mu 2 minus mu 3 is equal to 0 i got mu 1 plus mu 2 minus mu 3 here if I put that equal to 0. I get lambda equal to 1 and looking at these 2 equations. if I subtract 1 from the other I know mu 1 is equal to mu 2 and from here I know mu 3 is equal to mu 1 plus mu 2. So, I know mu 3 is equal to 2 mu 1 is equal to 2 mu 2. So, mu 1 is equal to mu 2 and mu 3 is twice of them once I put that in I its absolutely straightforward to see that the solution is mu 1 is equal to mu 2 is equal to 1 and mu 3 is equal to 2. This completely does in the SVM that is all the SVM is I have all the non0 Lagrange multipliers and the corresponding support vectors.

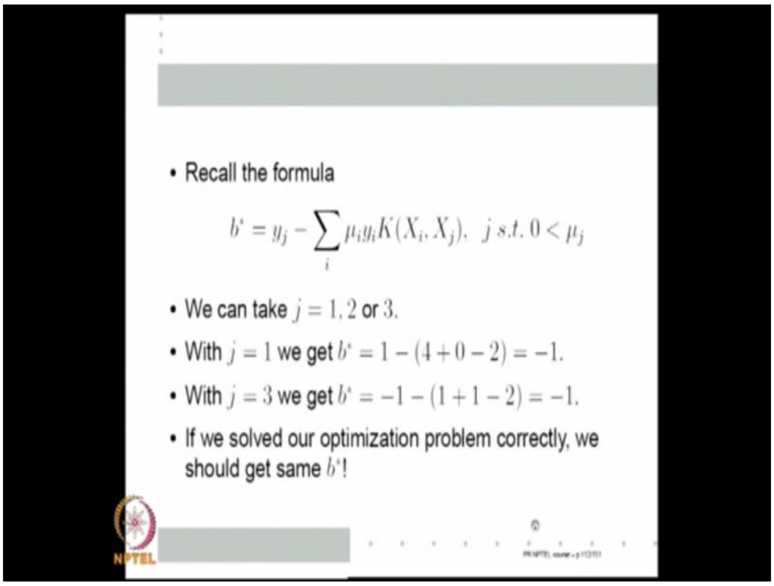
(Refer Slide Time: 51:15)




Just one thing I am not uses the penalty constant in this formulation because the high problem but, if I use the penalty constant with greater than 2. I would get the same solution because using the penalty constant simply means that in the I have an extra constraint that says that all mu's have to between 0 and 2 even with unconstrained my mu's are utmost 2. So, if C is greater than or equal to 2 the penalty constant would not make a difference the other hand by mistake we change should be less than 2 we will not get the solution we may get a bad solution will lets not go there. So, the classification by this SVM for any new vector x is determines the sign of f X where mu i y i K X i X plus b star lets first calculate b star to understand what this is. So, this is what mu i y i K X i X is K X 1 X mu 1 is 1 y 1 is 1 K X 2 X mu 2 is one-second is 1 minus 2 K X 3 X because mu 3 is 2 and y 3 is minus 1 plus b star i because i know X 1 X 2 and X 3. If you give me

X i can calculate all this.

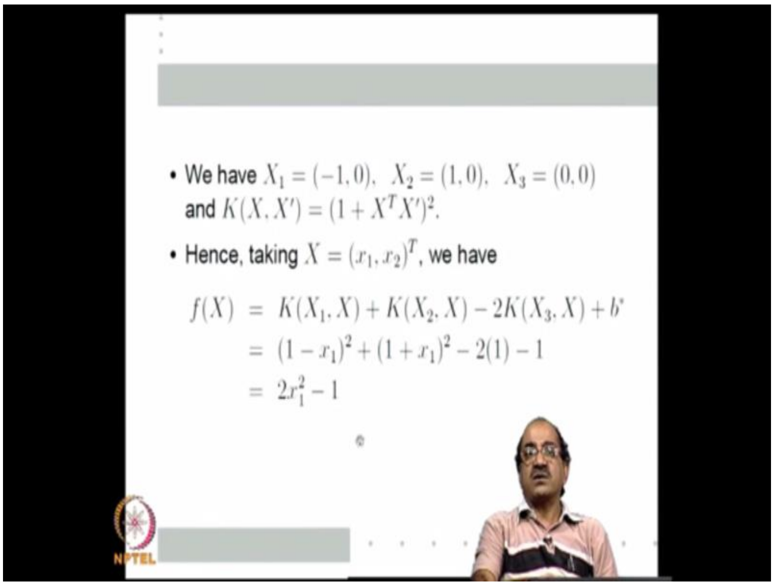
(Refer Slide Time: 52:23)




So, let us first calculate b star you remember b star is given by this y j minus mu i y i j X i X j for any j's such that mu j strictly greater than 0 in this case all three mu j's strictly greater than 0. So, if I put j is equal to 1 what do I get b star is y 1 y 1 is 1 minus that 3 terms here first 2 terms will be positive second term negative because y 1 y 2 is positive y 3 is negative mu 1 is 1K X 1 X 1 is 4 that is 4 K X X 2 X 1 1 is 0 and the other one is 1 I get a 2 from mu 3 and minus from. So, that gives me minus 1.

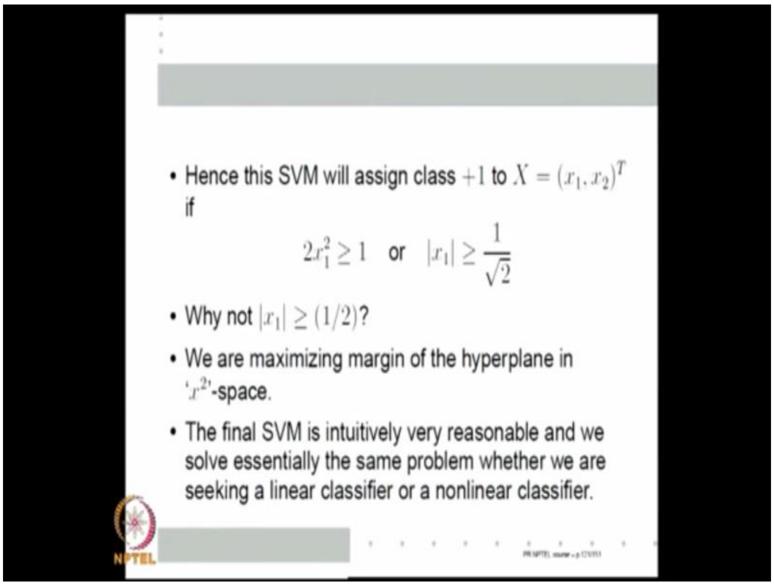
I do not have to chose j is equal to 1 any j will do if i 2 j is equal to 3 y 3 is minus 1 with respect to that I once again put that in this I get minus from it obviously basically if we solved our optimization problem correctly no matter which j such that mu j star mu j is greater than 0 I chose I should get the same b star matter of fact this is one way I can check whether we have solve the optimization problem correctly but, when we do it numerically this may not work. So, we may take b star to be the average of all b star obtained with different j's the minor issue.

(Refer Slide Time: 53:37)




So, we have X 1 X 2 X 3 is this and we know X K of X X comma X prime is 1 plus X transpose X prime whole square. So, if I take X to be X 1 X 2 I know my function is K of X 1 X K of X plus K of X two X minus 2 K of X 3 X plus b star. What will be X 1 X X 1 is this. So, this is 1 minus X 1 whole square. So, 1 minus X 1 whole square X 2 is this. So, it will be 1 plus X 1 whole square X 3 is this. So, X 3 transpose X is always 0. So, 2 into 1 b star is minus 1. So, 1 minus X 1 whole square plus 1 minus X 1 whole square will give me 2 X 1 square plus 2 that plus 2 will cancel this minus 2 I get minus 1. So, my f X is 2 X 1 square minus 1.

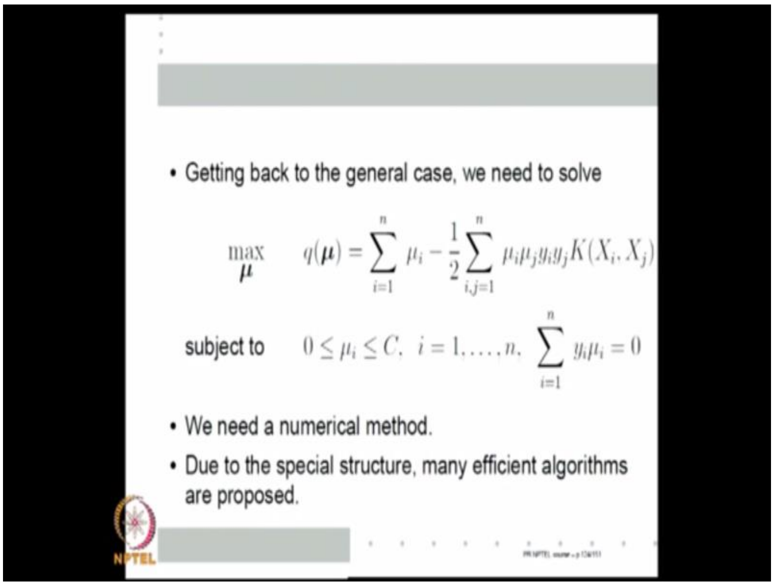
(Refer Slide Time: 54:31)



Which mean, this SVM will assign class plus 1 to X any given X X 1 X 2 if 2 X 1 square is greater than 1 or X 1 is greater than root 2 this exactly what we as we said the square of the first thing should be greater than some threshold that is the SVM that is magically come over we did nothing we just chose a Kernel function. Now, you may say it should not the classifier be mode X 1 greater than half. So, that i i put it exactly between the 0 and minus 1 or 0 1 plus 1 that is what we are asking by maximizing margin well, we shall understand that we are maximizing because we are using a kernel function where maximizing margin in the quadratic space in the X 1 square X 1 square space.

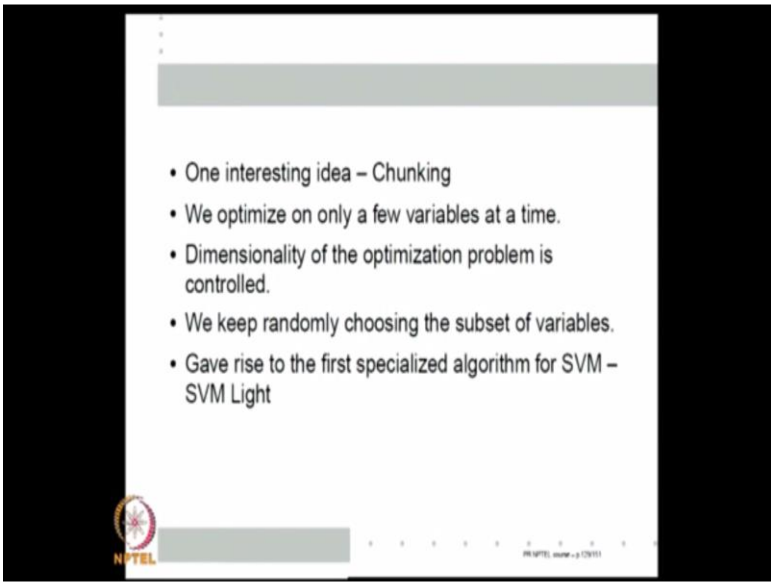
So, in the X 1 square space I wanted between 0 1 plus 1 that is at half. So, in the X 1 space will be 1 by root 2 does we maximizing margin in the transforms space. So, the final SVM is intuitively very reasonable and we solve exactly the same problem whether we are seeking a linear classifier or a nonlinear classifier. So, this is a good example, so we understand how to solve SVM's.

(Refer Slide Time: 55:47)



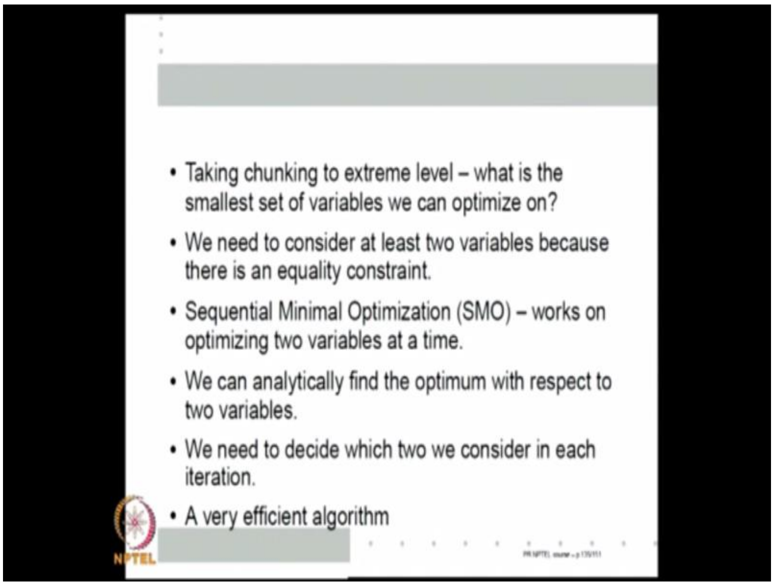

Now, let us move on and look at the general case this is what you want to solve and we need a numerical method as I said due to the special structure there are many efficient algorithms.

(Refer Slide Time: 56:00)




We essentially look at only one structure here what is called Chunking. Chunking is simply that when you have a optimization of many variables we do optimization a few variables at a time something that all of you might remember is what is called coordinate rise minimization if you have function of X 1 X 2 X n. We first keep X 2 to X n fix and minimize with respect X 1 as one dimensional problem than minimize with respect to X 2 than minimize with respect to X 3 and cyclically keep doing it. Lets has simple idea of Chunking we optimize on only a few variables at a time which allows the dimensionality of each optimization problem to be small and at each iteration we randomly choose a subset of variables of course, we have to use some proper heuristic to decide how to choose as long as all variables keep coming into the set being optimized again and again essentially chunking works. This basic idea gave rise to the first specialized and very efficient algorithm for solving SVM which is called SVM light.

(Refer Slide Time: 56:59)




And then, I can take the I want to take Chunking to the extreme level SVM light takes 5 or 10 or 15 variables at a time but, I want to take chunking to the extreme level I should ask what is the smallest chunk that I can optimize on because SVM has one equality constraint I can optimize at most on 2 variables at a time I cannot do change one variable keeping all the others fix because if I met a feasible point and change only one variable I go to infeasible point because there is a equality constraint.

So, there is a special algorithm call sequential minimal optimization which works on this principle of optimizing two variables at a time basically this very interesting algorithm because if I keep rest of them fix And I will say i will optimize only mu 1 mu 2 than by analytically manipulating the expression I can calculate the optimum with respect with respect to mu 1 mu 2 which means I do not have to do any numerical optimization at all I spend all my time deciding which two variable which 2 Lagrange multipliers I should do optimization next time. So, using a variety of heuristic this algorithm does a very good job of solving the SVM problem is a very efficient algorithm. So, I will just re talk about s this SMO and if and a couple of other issues in solving the SVM problem in the next class.

Thank you.