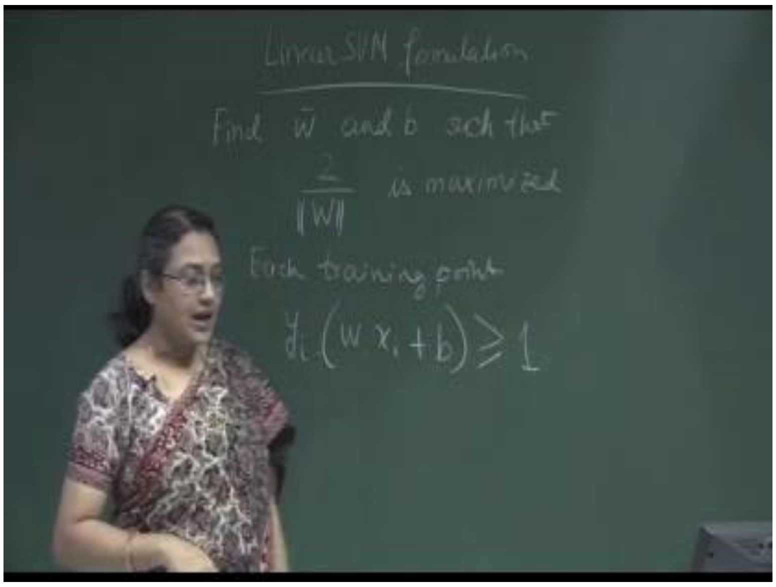
Introduction to Machine Learning

Prof. Sudeshna Sarkar
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture – 23
SVM: Maximum Margin with Noise

Good morning, today we will continue our lecture on Support Vector Machine. In the last class, we saw how to do work on linearly separable data set. Now, in many cases there will be noise in the data, so that a linear decision surface cannot separate the positive and negative points. We will see how support vector machine can deal with it.

(Refer Slide Time: 00:58)



Linear SVM formulation

Find $\bar{w}$ and b such that

$\frac{2}{\|W\|}$ is maximized

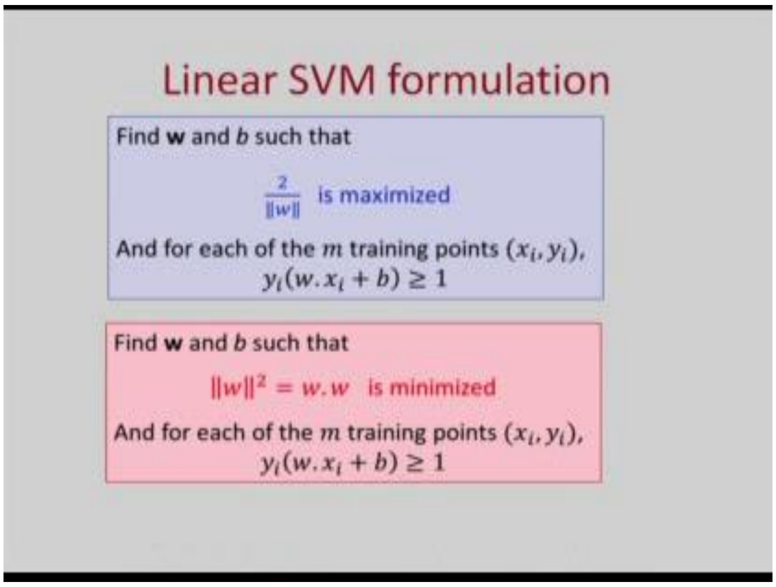
Each training point

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

If we look at the linear SVM formulation that we did in the last class, we had this optimization function which required us to find the weight vectors, w and b such that the margin width is maximised. So, the margin can be written like this 2 by w. So, this is maximised subject to m constraints corresponding to each training instance. So, for each training point i equal to 1 to m, we had y i w x i plus b is y i times w x i plus b is greater than equal to 1. This can also be written as instead of maximising 2 by w, we can maximise its reciprocal that is half w.

So, w is actually root over w dot w. So, we can minimise this, rather we can minimise w dot w. So, we can look upon it as a minimisation problem subject to m constraints corresponding to the m training examples.

(Refer Slide Time: 02:56)



Linear SVM formulation

Find **w** and b such that

$$\frac{2}{\|w\|} \text{ is maximized}$$

And for each of the m training points (x_i, y_i),
$$y_i(w.x_i + b) \geq 1$$

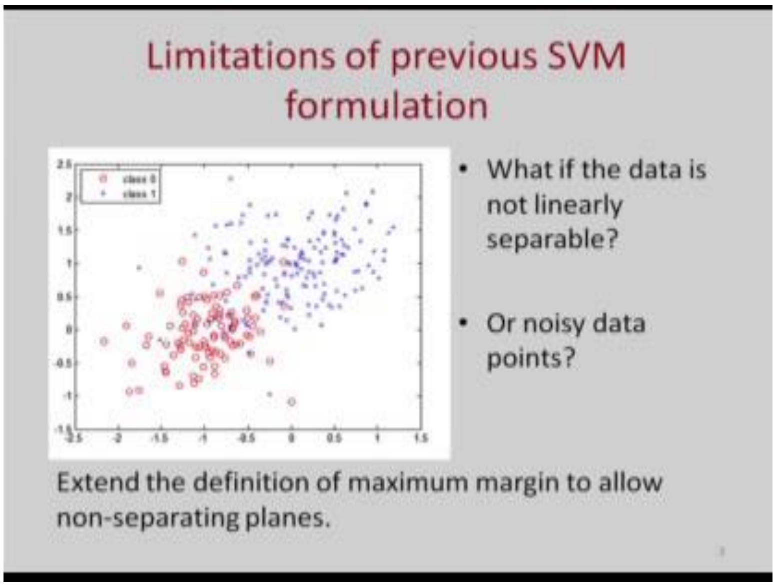
Find **w** and b such that

$$\|w\|^2 = w.w \text{ is minimized}$$

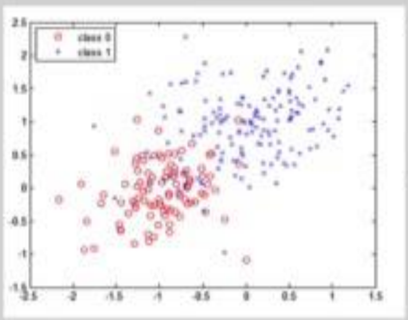
And for each of the m training points (x_i, y_i),
$$y_i(w.x_i + b) \geq 1$$

Now, this particular formulation enables us to deal with only those training instances which are linearly separable, but it may be that the data is not linearly separable.

(Refer Slide Time: 03:12)



Limitations of previous SVM formulation



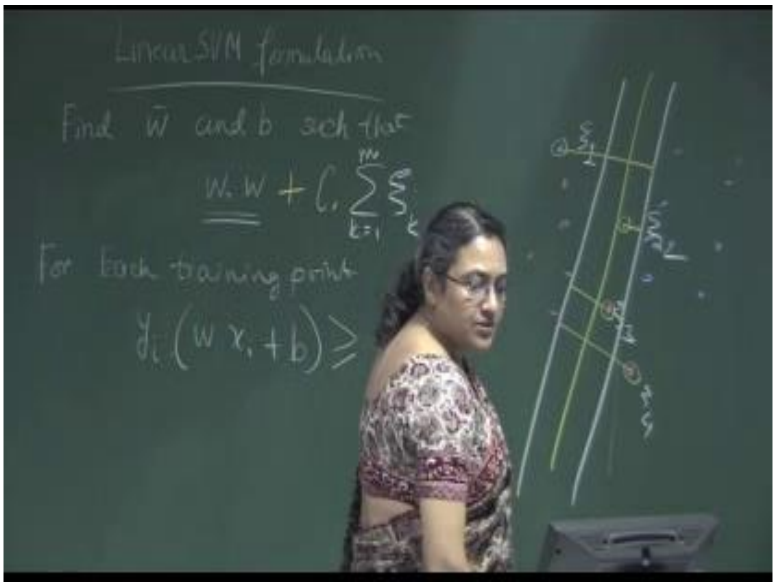
- What if the data is not linearly separable?
- Or noisy data points?

Extend the definition of maximum margin to allow non-separating planes.

If you look at the slide, we see that we have two classes blue and red and this data cannot be linearly separable. There could be a non-linear decision surface which separates the data or in this case it may be that the data points are noisy. So, that there is no clear separation of the points.

So, what we will like to do is we will like to extend the definition of maximum margin, so that we can allow these non-separable points. Now, we have to now redefine our objective function earlier.

(Refer Slide Time: 03:54)



Linear SVM formulation

Find $\bar{w}$ and b such that

$$\underline{\underline{w.w}} + C.\sum_{i=1}^{m} \xi_i$$

For each training point

$$y_i(w\,x_i + b) \geq$$

We were trying to minimise w dot w this is no longer sufficient. So, we have to be able to have a decision surface which makes some errors. So, if we have the points these are the red points and these are the blue points. We want to allow making decision surface which does not clearly separate the positive and negative points. So, that there is some error in this, but if we have this error we cannot we cannot deal with these constraints that we have looked at, we have to make some changes.

So, first of all we have to make changes to our optimisation function. So, one way of looking at it is we want a decision surface which looks at two things one is maximising the margin which corresponds to minimising w dot w, in fact we may say we want to reduce the number of miss classifications that is we want to minimise the miss classify training point or minimise the training error.

So, we may be tempt at to write this objective function as w dot w plus number of training errors and maybe we can use a parameter c to control how much importance we give to this part and this part the problem with this formulation is that it is no longer are quadratic objective functions, so that we cannot use a QP solver to solve this optimisation problem.

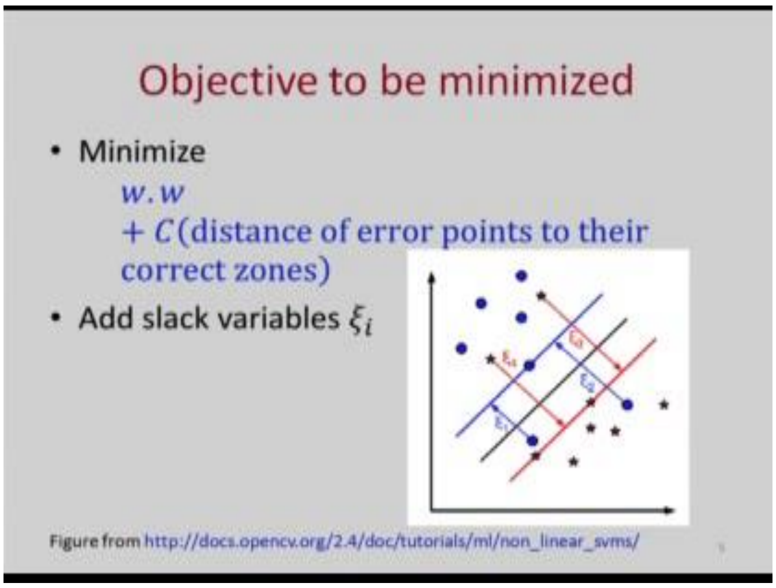
So, we have to look for an alternative formulation. In the alternative formulation that we look at, we do the following. We assume that given some points we are able to define a margin and then we try to; let me just redraw this. Suppose, this is my decision surface and this is a margin that we are looking for and these blue points may be here may be here where as the red point may be here. So, we can see that these point we have an error. So, this point the blue points ideally should be on this side of the margin. So, we can see that this blue point and this blue point are in error, where as this red point should be ideally on this side of the margin.

So, these red points are in error right corresponding to each of the points which are not correctly classified assuming a particular decision surface and the corresponding margin for each such point, we can give a penalty and the penalty can be given proportional to the how far it is from its correct position that is if we take this blue point what is the minimum amount by which I have to move this point, so that it is correctly classified. So, this can be obtained by dropping up a perpendicular from this point to this margin.

Similarly, for this blue point for this red point we have to move in this direction to get it correctly classified and for this we have to move it like this. So, what we will do is that with every miss classified point we will associate and error or penalty, which is the slack variable; suppose this is the point 1. So, we can associate is i 1 corresponding to this point z 2, corresponding to this point z 3, corresponding to this point z 4, corresponding to this point.

So, these are the penalty that we are associating with the points which are in the wrong position the z values associated with the other points will be equal to 0. Now, we can rewrite our objective function as w dot w plus c times or the some of this penalty functions. So, we can write c times sigma z k. So, k equal to 1 to m, where m is the number of training examples. So, if you look at the slides here.

(Refer Slide Time: 09:17)



Objective to be minimized

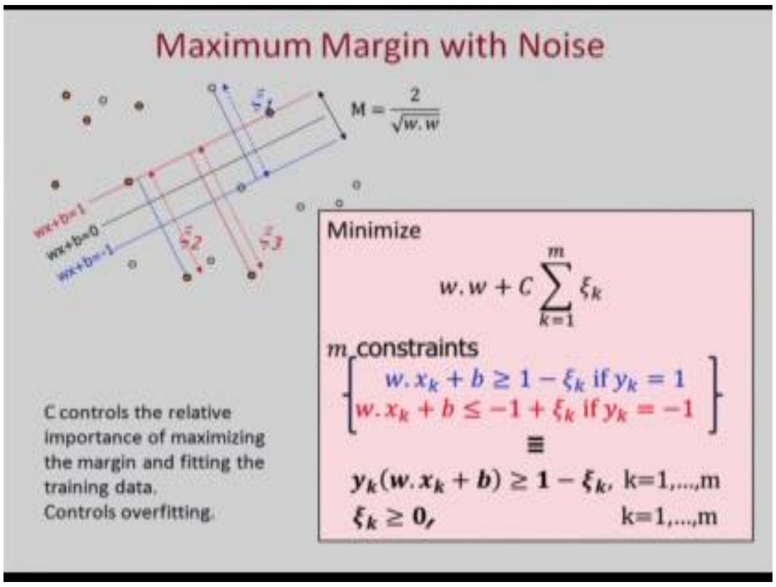
- Minimize
  $w.w$
  $+ C$(distance of error points to their correct zones)
- Add slack variables ξ_i

Figure from http://docs.opencv.org/2.4/doc/tutorials/ml/non_linear_svms/

ξ_1
ξ_2
ξ_3
ξ_4

We can rewrite our formulation as minimising w dot w, which corresponds to a making the margin looking at the size of the margin plus c times distance of error points to their correct zones and we achieve it by adding slack variables z i. So, for each point which is miss classified we find its shorter distance to the correct zone and this corresponds to the z for that variable for the other variables which are correctly classified by this particular decision surface i values are equal to c.

(Refer Slide Time: 09:57)



wx+b=1
wx+b=0
wx+b=-1
ξ_1
ξ_2
ξ_3

Maximum Margin with Noise

$$M = \frac{2}{\sqrt{w.w}}$$

Minimize

$$w.w + C\sum_{k=1}^{m} \xi_k$$

m constraints

$$\left\{\begin{array}{l} w.x_k + b \geq 1 - \xi_k \text{ if } y_k = 1 \\ w.x_k + b \leq -1 + \xi_k \text{ if } y_k = -1 \end{array}\right\}$$

$$\equiv$$

$$\boldsymbol{y_k(w.x_k + b) \geq 1 - \xi_k}, \quad k=1,\ldots,m$$

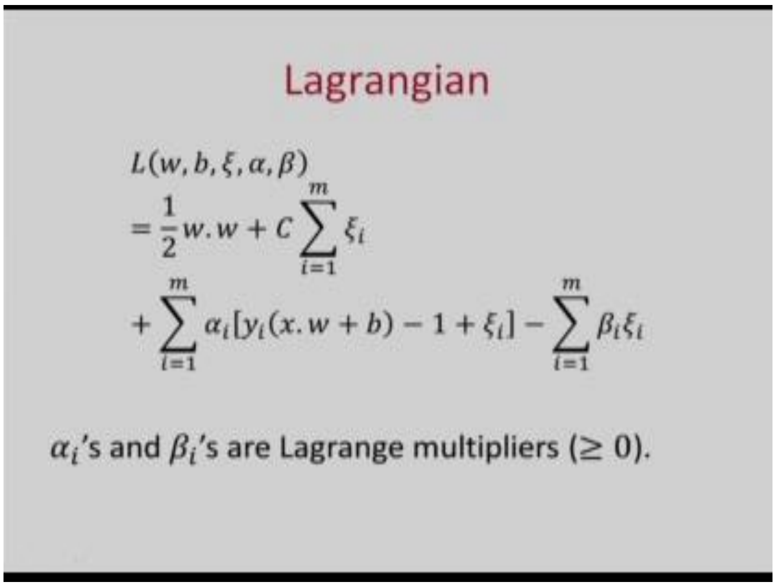
$$\boldsymbol{\xi_k \geq 0}, \quad k=1,\ldots,m$$

C controls the relative importance of maximizing the margin and fitting the training data.
Controls overfitting.

Now, let us look at the slide. So, we have this is our decision surface whose equation is given by w x plus b equal to 0 and upon normalization these are the two margin lines w x plus b equal to plus 1 and w x plus b equal to minus 1 and here are the three points which are wrongly classified and corresponding to each of them, we have the variables z 1, z 2, z 3, the margin width is given by 2 by root over w dot w, which we want to maximise and which corresponds to minimising w dot w adding the penalty due to this slack variables we get the minimisation function to be w dot w plus c times sigma k equal to 1 to m z k. So, this is what we want to minimise.

Earlier the constraints that we had is y i w dot x i plus b greater than equal to 1. Now, you see that this constraint will no longer hold for those points which are in the wrong zones. So, we can rewrite this constraint as y k y i w x i plus b greater than equal to 1 minus z k right for i equal to 1 to m actually for the blue points, which are wrongly classified we can write w dot x i plus b greater than equal to 1 minus z k and for the red points we can write w x i plus b less than equal to minus 1 plus z k and we can write it together as y i into w x i plus b greater than equal to 1 minus z k for i equal to 1 to m. So, this is the formulation of the maximum margin classifier, when we wish to handle noise. Now, in order to solve this equation we can find the Lagrangian of this quantity.

(Refer Slide Time: 12:17)



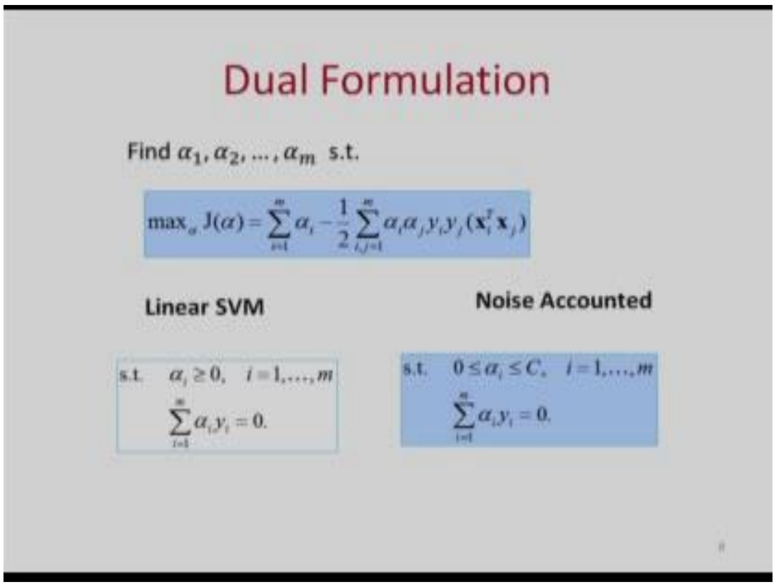
Lagrangian

$$L(w, b, \xi, \alpha, \beta)$$
$$= \frac{1}{2} w.w + C \sum_{i=1}^{m} \xi_i$$
$$+ \sum_{i=1}^{m} \alpha_i [y_i(x.w + b) - 1 + \xi_i] - \sum_{i=1}^{m} \beta_i \xi_i$$

α_i's and β_i's are Lagrange multipliers (≥ 0).

So, the lagrangian with parameters w b z alpha beta and we get it as half w dot w plus c sigma z i plus sigma i equal to 1 to m alpha i y i into x dot w plus b minus 1 plus z i minus sigma i equal to one to m beta i z i. So, actually there is another set of constraints. In this case, we have a set of m non-linear constraints and we also have m constraints z i greater than equal to 0 for i equal to 1 to m. So, we have two m constraints m constraints of this type and m constraints of this type and therefore, in the slide we can see the lagrangian of this quantity has half w dot w plus c sigma z i plus, please look at the slide, sigma i equal to 1 to m alpha i times y i x dot w plus b minus 1 plus z i minus i equal to 1 to m beta i z i alpha i s and beta i s are the lagrangian multipliers and they are greater than equal to 0.

(Refer Slide Time: 13:44)



Dual Formulation

Find $\alpha_1, \alpha_2, \ldots, \alpha_m$ s.t.

$$\max_\alpha J(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i^T \mathbf{x}_j)$$

Linear SVM

$$\text{s.t.} \quad \alpha_i \geq 0, \quad i = 1, \ldots, m \qquad \sum_{i=1}^{m} \alpha_i y_i = 0.$$

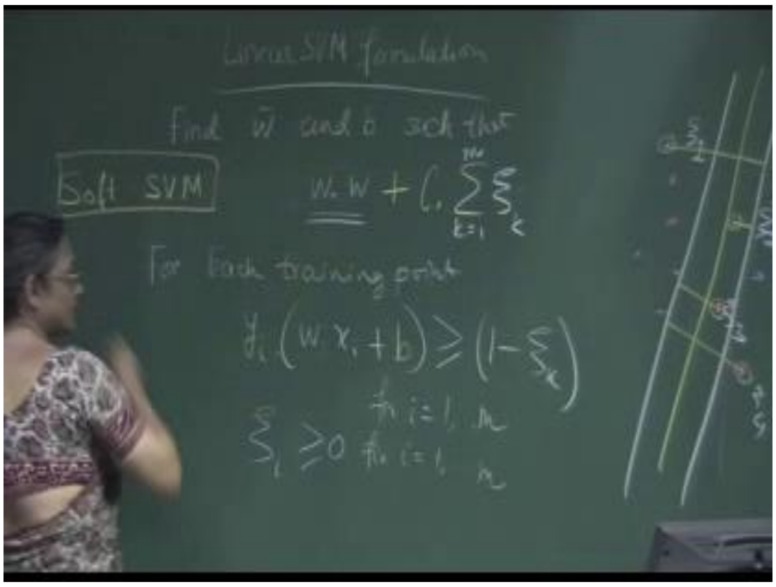
Noise Accounted

$$\text{s.t.} \quad 0 \leq \alpha_i \leq C, \quad i = 1, \ldots, m \qquad \sum_{i=1}^{m} \alpha_i y_i = 0.$$

Now, given this lagrangian we can look at the dual formulation corresponding to this formulation. So, in the linear SVM, we wanted to find alpha 1, alpha 2, alpha m such that this quantity sigma alpha i minus half alpha i alpha j y i y j x i dot x j is a maximised and such that the constraints where alpha i greater than equal to 0, for i equal to 1 to m and sigma i equal to 1 to m alpha i y i equal to 0. In the case of noisy SVM this quantity remains the same, but the constraints change. So, earlier we had alpha i greater than equal to 0, we have now alpha i is between 0 and c for i equal to 1 to m, the second constraint remains the same. So, the dual formulation of this noise accounted SVM this constraint is changing alpha i is now lying between 0 and c the rest of the formulation is the same.

Now, given this dual formulation we can find the solution to the classifier. So, this is called noisy linear SVM or most commonly this is called soft SVM because we do not have a hard classifier or a hard decision surface that clearly separates the positive and negative points.

(Refer Slide Time: 15:22)


Linear SVM formulation

Find $\bar{w}$ and b such that

Soft SVM

$$\underline{w.w} + C.\sum_{i=1}^{m}\xi_i$$

For each training point

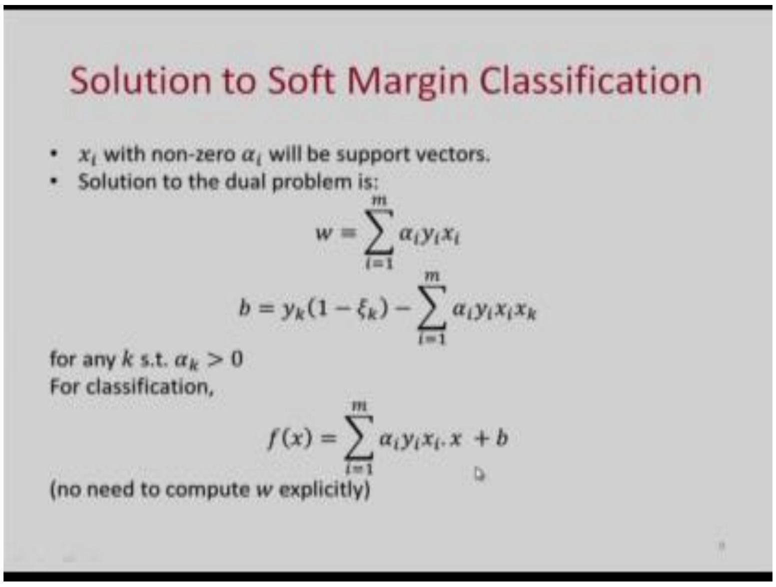
$$y_i(w.x_i + b) \geq (1 - \xi_i)$$

for $i = 1, m$

$$\xi_i \geq 0 \text{ for } i = 1, m$$

And the solution of this soft margin classification will be similar to the solution that we got earlier.

(Refer Slide Time: 15:34)


Solution to Soft Margin Classification

- x_i with non-zero α_i will be support vectors.
- Solution to the dual problem is:

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

$$b = y_k(1 - \xi_k) - \sum_{i=1}^{m}\alpha_i y_i x_i x_k$$

for any k s.t. $\alpha_k > 0$

For classification,

$$f(x) = \sum_{i=1}^{m}\alpha_i y_i x_i . x \ + b$$

(no need to compute w explicitly)

For example, those data point whose, which have non-zero corresponding. Secondly, multipliers alpha i they will be the support vector and as you can see the support vectors will be those points which lie at the margin as well as those points which lie in the incorrect zone, and the solution to the dual problem is given by w equal to sigma i equal to 1 to m alpha i y i x i alpha i is non-zero only for those points which are support vectors and once we find the values of alpha i we can find the value of b from any one of those points. So, b equal to y k into 1 minus z k minus sigma alpha i y i x i x k.

So, for any of this k we can find b after we are found the alpha i’s. For classification, now once we have found the solution of this equation we can use it for classification for classification again, we use the similar formula f (x) equal to sigma i equal to 1 to m alpha i y i x i dot x j x plus b. So, x is the test point and i corresponds to you know. Secondly, because to those point for which alpha i is non-zero, which are exactly the support vectors.

So, again we see that in order to find the classification of a point, we actually do not need to compute w explicitly, but we can just apply this formula and in this formula we need to find the dot product of the test point with the x value of those points which are support vectors and multiply by the corresponding alpha i y i values sum it up and add b to this right. So, the formulation is pretty straight forward and quite simple and easy to apply. So, this is how we deal with soft classification or soft SVM. However, if the decision surface, we are able to account for noise and come up with classifiers which do not exactly separate the positive and negative points.

However, the classifier decision surface is still linear and cannot handle those cases where the decision surface is actually non-linear, how to handle that in some cases we will see in the next class.

Thank you.