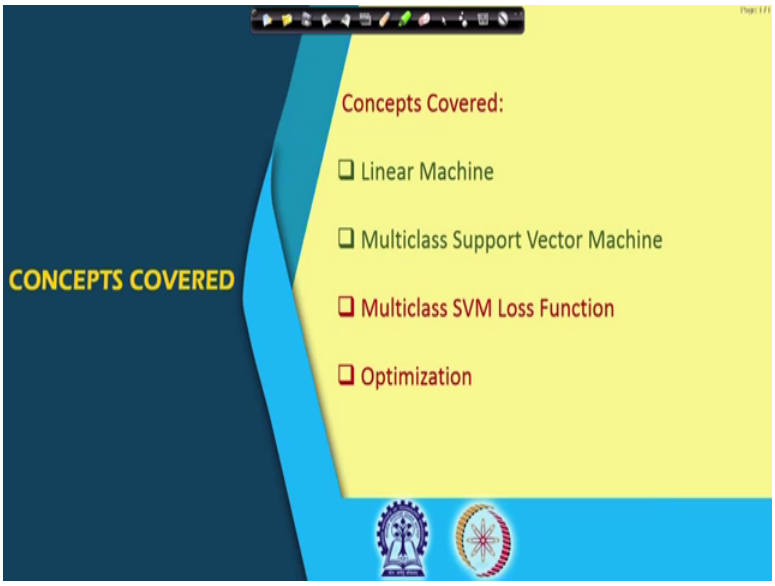
Deep Learning
Prof. Prabir Kumar Biswas
Department of Electronics and Electrical Communication Engineering
Indian institute of Technology, Kharagpur

Lecture - 15
Multiclass Support Vector Machine - II

Hello, welcome to the NPTEL online certification course on Deep Learning.

(Refer Slide Time: 00:34)



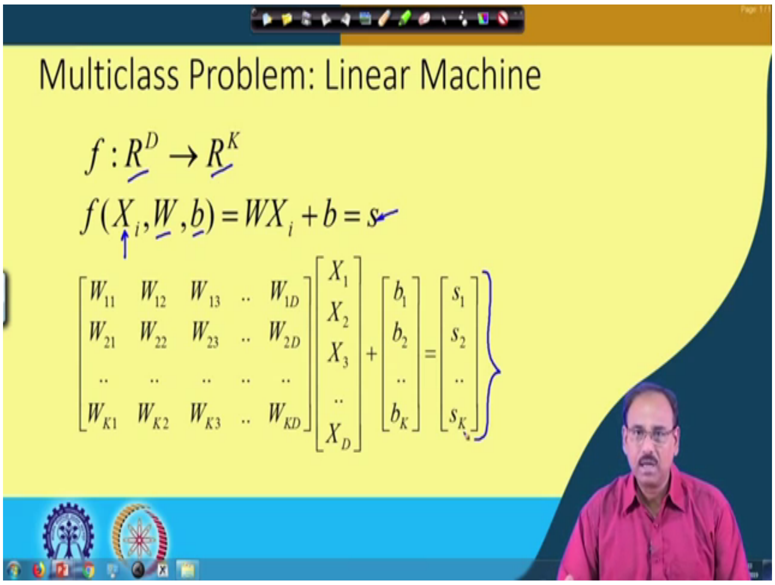
CONCEPTS COVERED

Concepts Covered:

- ❑ Linear Machine
- ❑ Multiclass Support Vector Machine
- ❑ Multiclass SVM Loss Function
- ❑ Optimization

In the previous class, we have talked about the linear machines and we have also discussed started our discussion on multi class support vector machine. Today's, lecture we will discuss on multi class support vector machine loss function and also the optimization techniques.

(Refer Slide Time: 00:55)



Multiclass Problem: Linear Machine

$$f : R^D \to R^K$$

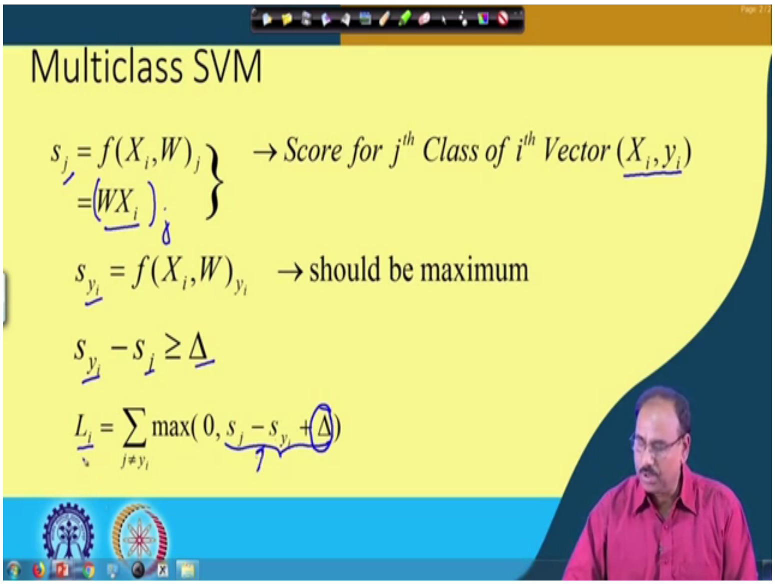
$$f(X_i, W, b) = WX_i + b = s$$

$$\begin{bmatrix} W_{11} & W_{12} & W_{13} & .. & W_{1D}\\ W_{21} & W_{22} & W_{23} & .. & W_{2D}\\ .. & .. & .. & .. & ..\\ W_{K1} & W_{K2} & W_{K3} & .. & W_{KD}\end{bmatrix}\begin{bmatrix}X_1\\X_2\\X_3\\..\\X_D\end{bmatrix}+\begin{bmatrix}b_1\\b_2\\..\\b_K\end{bmatrix}=\begin{bmatrix}s_1\\s_2\\..\\s_K\end{bmatrix}$$

So, what we have done in case of linear machine is that, we have seen that linear machine is a function that, transforms or that maps are D dimensional feature vector R D into a score function of dimension K. So, score function S that you get is also a vector of dimension K, where K is the number of categories or the number of classes.

So, if I expand this the function looks like this that f given an input vector X I f X i W b here W is the weight matrix and b is the bias vector is given by W X i plus b, which is equal to the score function S. And, we have seen that the score function S has got K number of components where K is the number of classes or the number of categories.

(Refer Slide Time: 02:00)



Multiclass SVM

$$\left.\begin{aligned} s_j &= f(X_i,W)_j \\ &= (WX_i)_j \end{aligned}\right\} \rightarrow \text{Score for } j^{th} \text{ Class of } i^{th} \text{ Vector } (X_i, y_i)$$

$$s_{y_i} = f(X_i,W)_{y_i} \quad \rightarrow \text{should be maximum}$$

$$s_{y_i} - s_j \geq \Delta$$

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + \Delta)$$

So, given this we have seen that the score for the j th class or the j th category is the j th component of the score function S, which in this case we have written as S j and this S j is nothing, but W X i the j th component of this.

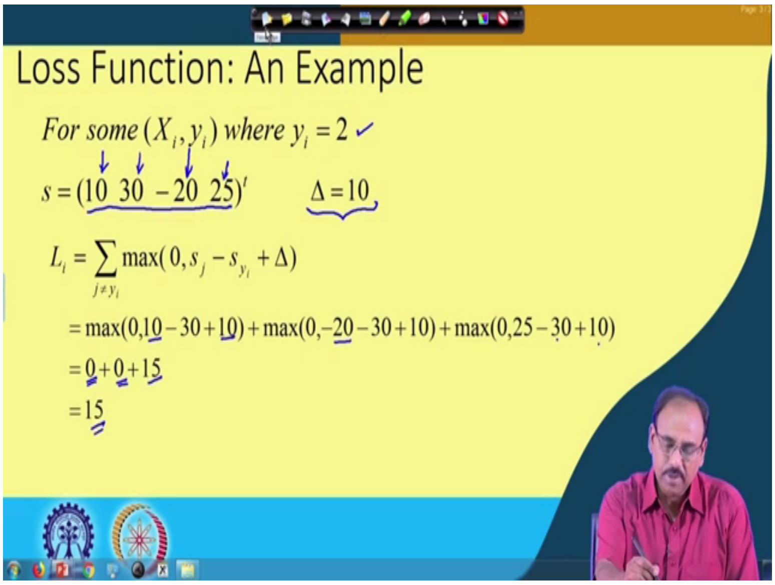
So, we can write this as f X i W the j th component. So, the j th component of this gives me the score for the j th class of the i th vector X i which belongs to class y i. So, these are the training vectors. And, because this X i the input vector belongs to class y i. So, the score function component y i must be maximum. So, when this linear machine gives you the score function, the score function the y i component of the score function must be maximum, because we have taken X i belonging to class y i.

And, we are not only satisfied with the score function to be maximum, we also want that the score function should be more than the score function of other classes by at least a threshold delta; that means, taken any other class S j our S of y i the score function of the class y i must be greater than S j, where j is any other category other than y i. So, this difference must be greater than some threshold delta.

So, accordingly for the i th component the loss function that we get L i, which is nothing, but maximum of 0 and S j minus S y i plus delta. So, you find that as long as S y i is greater than S j, then it is greater than S j by an amount delta. So, this amount S j minus S y i plus delta will be less than 0. Whereas, if S j is equal to S y i then this function will be equal to delta and if S j is greater than S y i this function will be greater than delta, where delta use a positive threshold.

So, in that case when you take max of 0 and this output will be this only if it is than 0. And, the output of this max function will be equal to 0 only when S y i is greater than S j by at least this threshold amount delta. And, you sum it over all j not equal to i, I get the last component S i. And, the overall loss that you get is given by sum of all these lost components.

(Refer Slide Time: 05:11)



Loss Function: An Example

For some (X_i, y_i) *where* $y_i = 2$ ✓

$s = (10\ \ 30\ \ -20\ \ 25)'$ $\qquad \Delta = 10$

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + \Delta)$$

$$= \max(0,10-30+10) + \max(0,-20-30+10) + \max(0,25-30+10)$$

$$= 0 + 0 + 15$$

$$= 15$$

So, to explain this we had taken an example, that suppose I have some X i y i write this y i is equal to 2; that means, this X i belongs to a category 2. And, the suppose the score function that is computed is given by this S equal to 10 30 minus 20 and 25 and let us assume that we have a threshold delta which is equal to 10.

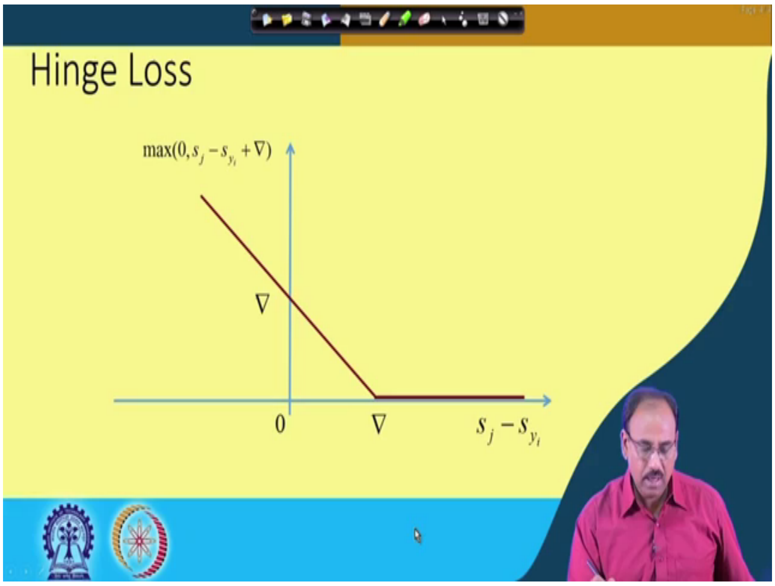
So, if i compute L i in this case i is going to 2, because we have taken X i from category 2. So, if i compute L i you find that it will have these 3 components; one is maximum of 0 10 minus 30 plus 10 this then is delta. The first 10 is the score function for category 1 and 30 is the score function for category 2.

So, this is 0 10 minus 30 plus 10 which becomes minus 30. So, the corresponding max function gives you an output 0. Similarly, for the second case it is max of minus 20, which is core function for category 3 minus 30, that is core function for category 2 plus

10 again this part becomes negative so, output is 0. And for the other one the score function is 25, which is the score function for category 4. So, it is 25 minus 30 plus 10 and that gives you an output 15. So, as a result L 2 is equal to 15.

So, when I compute these core functions as I said before that if I take summation of this core function of overall value of i for all value of j not equal to y i, I get on the overall score function.

(Refer Slide Time: 07:23)

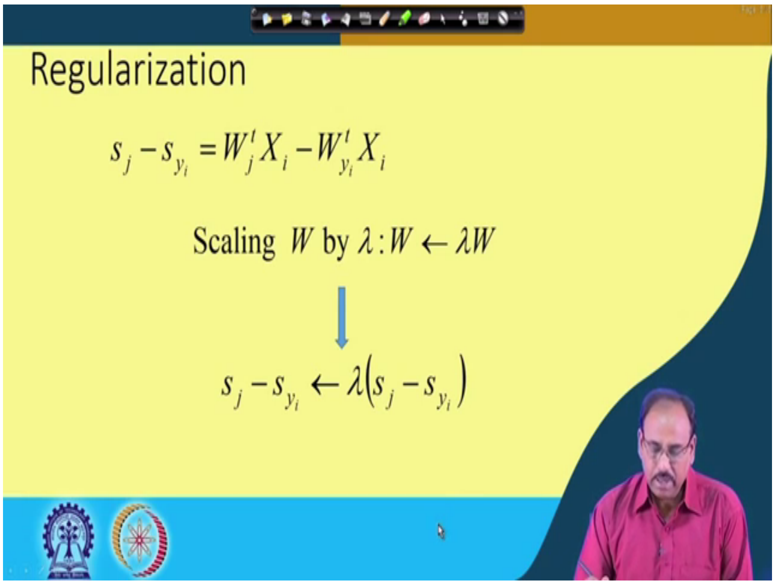


Hinge Loss

So, this is how we compute the score function. And, if you look at the nature of this core function you find that as long as this component S i S j minus S y i is positive ok. My score function will be equal to 0 only when this term S j minus S y i, as long as this term is less than delta my output will be 0. If, it is S y i minus S j this is greater than delta, then output will be equal to 0, because my classification is correct otherwise the output will be high.

So, if I plot this score function with respect to S j minus S y i the nature of the score function or the plot of the loss function will be something like this and this loss function is what is known as hinge loss.

(Refer Slide Time: 09:07)



Regularization

$$s_j - s_{y_i} = W_j^t X_i - W_{y_i}^t X_i$$

Scaling W by $\lambda : W \leftarrow \lambda W$

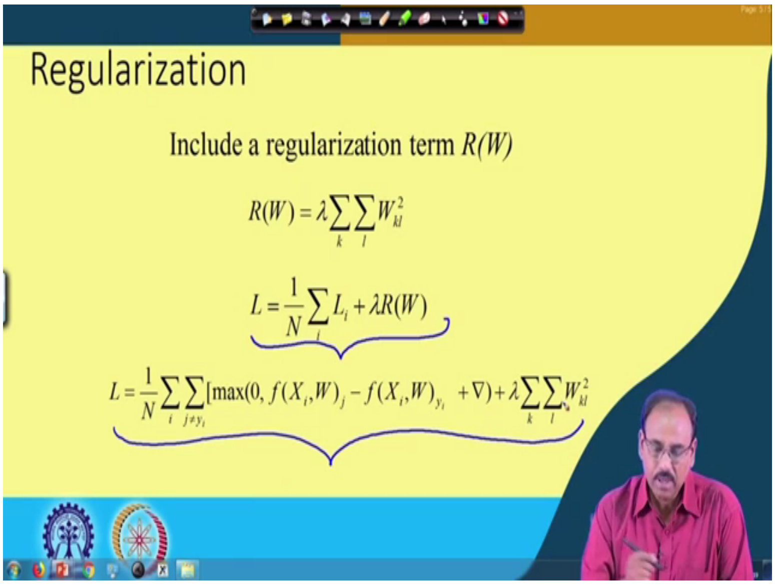
$$s_j - s_{y_i} \leftarrow \lambda(s_j - s_{y_i})$$

So, now we have also talked about a term called regularization, because you find that we decide about the classification to be correct or not or whether we are satisfied with the classification output or not, depending upon the difference S j minus S y i. And, which is nothing, but W j transpose that is W j is the j th row of the weight matrix W that transpose X i minus W y i transpose X i.

And, if you find you find that, if I scale up this W by a factor lambda, then the score the difference S j minus S y i will also be scaled up by the same factor lambda say. For example, for some W if the difference is j minus S y i is equal to 15 and if I multiply W scale up W by a factor 2, then the same difference S j minus S y i will be 30.

So, there are many possible values of W for which the value of the difference of S j minus S y i can be greater than delta right. So, what I need is I need to find out an optimum W or best value of W, which will satisfy all my requirement.

(Refer Slide Time: 10:39)



Regularization

Include a regularization term $R(W)$

$$R(W) = \lambda \sum_k \sum_l W_{kl}^2$$

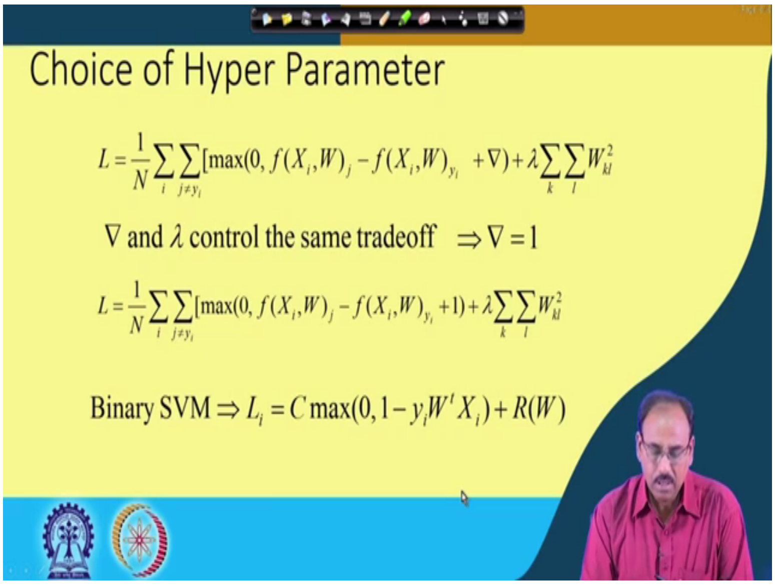
$$\underbrace{L = \frac{1}{N} \sum_i L_i + \lambda R(W)}$$

$$\underbrace{L = \frac{1}{N} \sum_i \sum_{j \neq y_i} [\max(0, f(X_i, W)_j - f(X_i, W)_{y_i} \ + \nabla) + \lambda \sum_k \sum_l W_{kl}^2}$$

So, as a result I have to include a regularization term, which is a function of W or from the weight matrix. And, this regularization term is usually taken to be a L 2 known. So, our regularization term L 2 R W becomes lambda times W K L squared, where you take the summation of W K L square or overall value of K and L. And, as a result our overall loss function becomes L is equal to sum of L i, that divided by N where N is the number of training samples we have plus lambda times our W or the overall loss function in the expanded form is given by this.

So, given this overall loss function, now what we need to do is we need to optimize this loss function or minimize this loss function.

(Refer Slide Time: 11:40)



Choice of Hyper Parameter

$$L = \frac{1}{N}\sum_i \sum_{j \neq y_i} [\max(0, f(X_i,W)_j - f(X_i,W)_{y_i} + \nabla) + \lambda \sum_k \sum_l W_{kl}^2$$

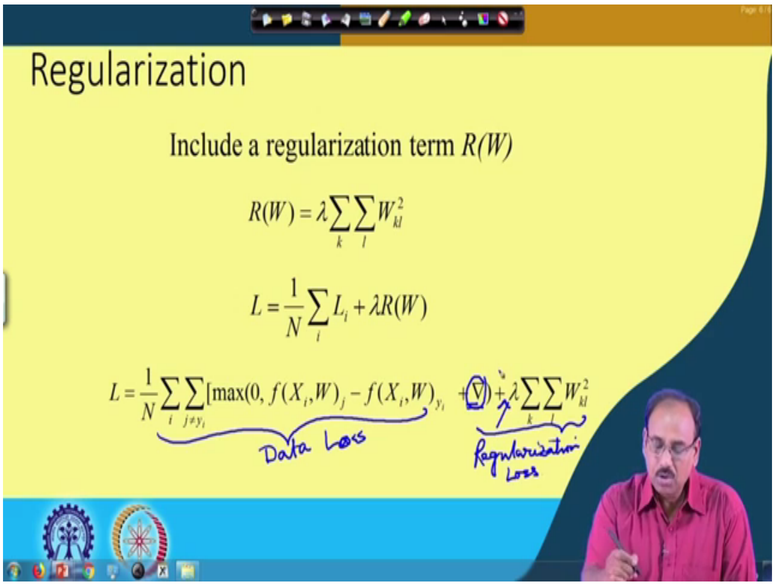
∇ and λ control the same tradeoff $\Rightarrow \nabla = 1$

$$L = \frac{1}{N}\sum_i \sum_{j \neq y_i} [\max(0, f(X_i,W)_j - f(X_i,W)_{y_i} + 1) + \lambda \sum_k \sum_l W_{kl}^2$$

Binary SVM $\Rightarrow L_i = C\max(0, 1 - y_i W' X_i) + R(W)$

And, in addition to that, what we need to do is we have to also choose the hyper parameters.

(Refer Slide Time: 11:52)



Regularization

Include a regularization term $R(W)$

$$R(W) = \lambda \sum_k \sum_l W_{kl}^2$$

$$L = \frac{1}{N}\sum_i L_i + \lambda R(W)$$

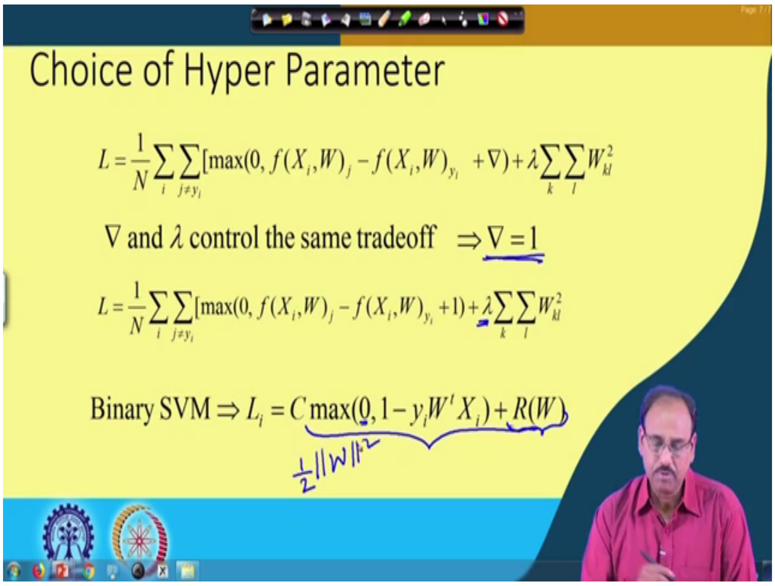
$$L = \underbrace{\frac{1}{N}\sum_i \sum_{j \neq y_i} [\max(0, f(X_i,W)_j - f(X_i,W)_{y_i}}_{\text{Data Loss}} + \boxed{\nabla}) + \underbrace{\lambda \sum_k \sum_l W_{kl}^2}_{\text{Regularization Loss}}$$

So, if you look at the previous expression you find that we have got two hyper parameters over here, one of the hyper parameter is delta, which is the threshold that we have used and other hyper parameter is lambda, which is in the regularization term.

So, if you remember this first term in this loss function we talked to this we defined this as data loss and the last one is what is known as a regularization loss.

So, you have a to hyperparameters over here, one is the threshold delta and other one is this lambda. So, I need to choose that what should be the proper values of these two hyperparameters.

(Refer Slide Time: 12:49)



Choice of Hyper Parameter

$$L=\frac{1}{N}\sum_{i}\sum_{j\neq y_i}[\max(0, f(X_i,W)_j - f(X_i,W)_{y_i} \ +\nabla)+\lambda\sum_{k}\sum_{l}W_{kl}^2$$

∇ and λ control the same tradeoff $\Rightarrow \underline{\nabla=1}$

$$L=\frac{1}{N}\sum_{i}\sum_{j\neq y_i}[\max(0, f(X_i,W)_j - f(X_i,W)_{y_i} +1)+\lambda\sum_{k}\sum_{l}W_{kl}^2$$

$$\text{Binary SVM} \Rightarrow L_i = C\underbrace{\max(0,1-y_iW'X_i)+R(W)}_{\frac{1}{2}\|W\|^2}$$

However, if you look carefully you find that both lambda and delta, they control the same trade off. That is if the lambda is more the difference of S j and S y will also be more, if lambda is less the difference of S j and S y i will also win this. So, accordingly I will have an effect of lambda. And, due to this we can safely choose the value of lambda is equal to 1.

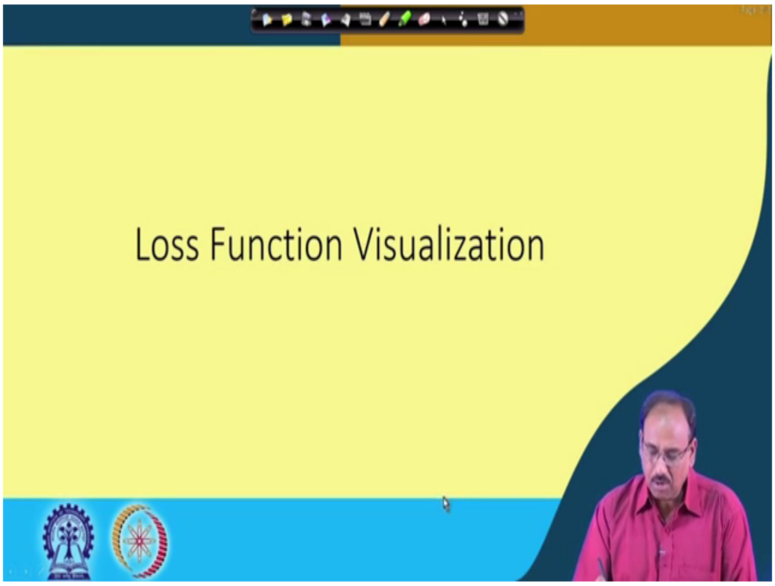
So, that is what we have done over here, the value of lambda is taken to be 1. And, accordingly when you go for minimization of this loss function the value of lambda will be chosen. So, we have taken value of delta equal to 1 and will do your minimization accordingly the value of lambda will be chosen.

And, you find that earlier we have talked about from the binary support vector machine where we have said, that given the separating plane between 2 classes omega 1 and omega 2. During training I will be satisfied only when we find that W transpose X i for our training vector X i is more than a normalized distance 1. So, accordingly a loss function for a binary SVM can be defined a like this it is max of 0 minus y i W transpose X i.

So, you find that as long as this W transpose X i is loss less than 1, that is the normalized distance from W transpose X equal to 0 that is the separating plane is less than 1 your loss function will be positive. If, it is greater than 1 then only loss function it is 0. And, the regularization term in case of 2 plus support vector machine, if you remember we have put this as half of W square, that was the regularization term in case of that two class support vector machine.

So, you find that this binary SVM or 2 class support vector machine is nothing, but a special case of a multi class support vector machine right.

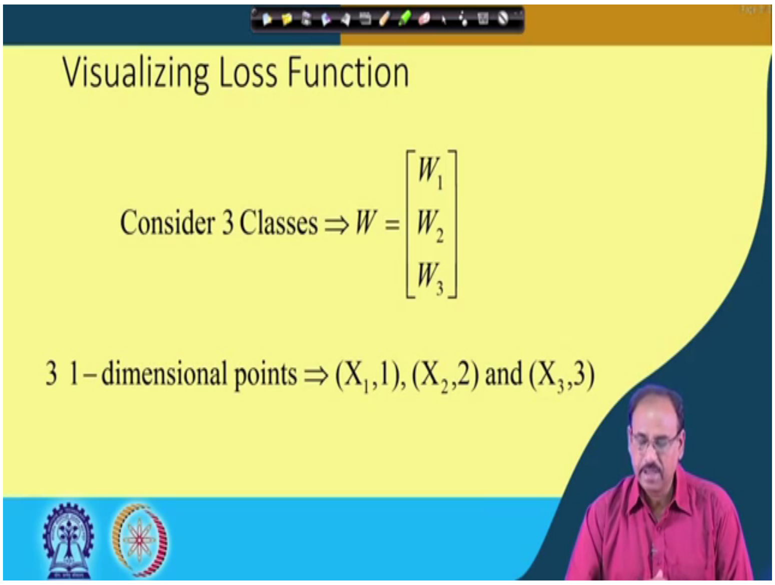
(Refer Slide Time: 15:01)



Loss Function Visualization

So, now let us go for how to see, how this loss function, what is the nature of this loss function?

(Refer Slide Time: 15:09)



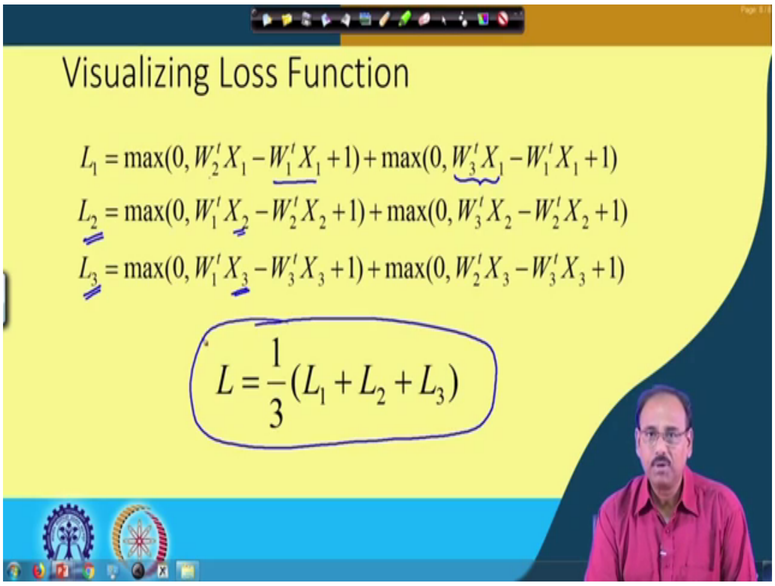
Visualizing Loss Function

Consider 3 Classes $\Rightarrow W = \begin{bmatrix} W_1 \\ W_2 \\ W_3 \end{bmatrix}$

3 1-dimensional points $\Rightarrow (X_1,1), (X_2,2)$ and $(X_3,3)$

So, for illustrating this I consider a 3 class problem and that is the vectors that I consider is 1 dimensional vector. So, suppose I have got 3 classes given by the weights W 1 W 2 and W 3 and as I said that, I am considering 1 dimensional vectors. So, each of this W 1 W 2 W 3 are scalars ok. And, I also take 3 1-dimensional training points X 1 taken from class 1, X 2 taken from class 2 and X 3 taken from class 3.

(Refer Slide Time: 15:49)



Visualizing Loss Function

$$L_1 = \max(0, W_2^t X_1 - \underline{W_1^t X_1} + 1) + \max(0, \underbrace{W_3^t X_1} - W_1^t X_1 + 1)$$

$$\underline{\underline{L_2}} = \max(0, W_1^t \underline{X_2} - W_2^t X_2 + 1) + \max(0, W_3^t X_2 - W_2^t X_2 + 1)$$

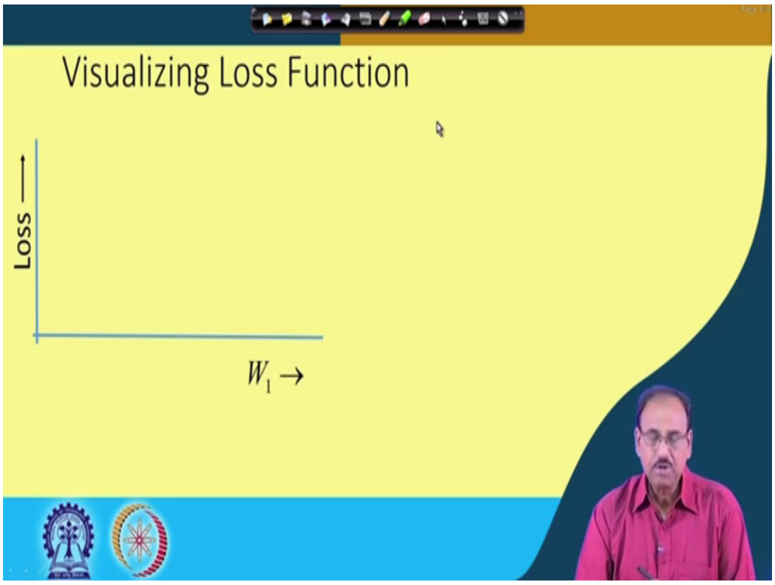
$$\underline{\underline{L_3}} = \max(0, W_1^t \underline{X_3} - W_3^t X_3 + 1) + \max(0, W_2^t X_3 - W_3^t X_3 + 1)$$

$$\boxed{L = \frac{1}{3}(L_1 + L_2 + L_3)}$$

So, given this situation you find that the loss functions that, we will get is given by L 1 is equal to max of 0 and W 2 transpose X 1. So, this W 2 transpose X 1 is nothing, but score of class 2 for this training vector X 1. And, this is the score of class 1 for training vector X 1. So, the loss function L 1 will be max of 0 W 2 transpose X 1 minus W 1 transpose X 1 plus 1 plus max of 0 and W 3 transpose X 1.

So, this W 3 transpose X 1 is the score of class 3 for training vector X 1 minus W 1 transpose X 1 plus 1. Similarly, we define loss function for plus 2 of L 2 for the support vector 2 we also define the loss function L 3 for support for the training vector X 3 and the overall loss is given by one-third of L 1 plus L 2 plus L 3.

(Refer Slide Time: 17:11)



Visualizing Loss Function

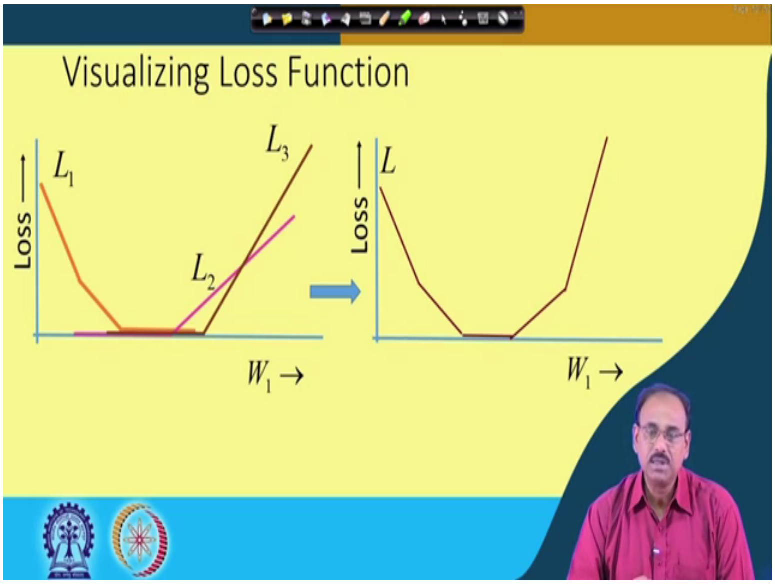
So, given this now, if we try to visualize how this loss function is. So, here you find that in case of L 1, if W 1 is very small, if this weight vector W 1 is very small, then W 2 transpose X 1 minus W 1 transpose X 1, this term will be positive assuming that W 1 transpose X 1 is less than W 2 transpose X 1. So, output loss function will be positive, which is max of 0.

And, this and this particular term will be 0 only when W 1 transpose X 1 is more than W 2 transpose X 1 at least by 1 ; that means, as long as W 1 is very small your loss function given by this value of loss is positive. Similarly, in this case as long as W 1 is less than W 3 here also it will be positive.

However, as W 1 goes on increasing the loss function gradually reduces and ultimately it becomes 0 and remains 0 for this component L 1. Similarly, for component L 2 you find that when W 1 is very small compared to this.

So, that this term becomes negative the output will be 0. And, as W 1 goes on increasing eventually this term becomes positive the output L 1 will also become positive and it will have a certain value, it is not 0 and same is for L 3. So, by this understanding, having this understanding now we can try to plot the different loss functions L 1, L 2, and L 3.

(Refer Slide Time: 19:04)



Visualizing Loss Function

So, if I plot L 1 with W 1 you find that as we said that when W 1 is very small the loss component L 1 is positive and is it goes on reducing as value of W 1 increases.

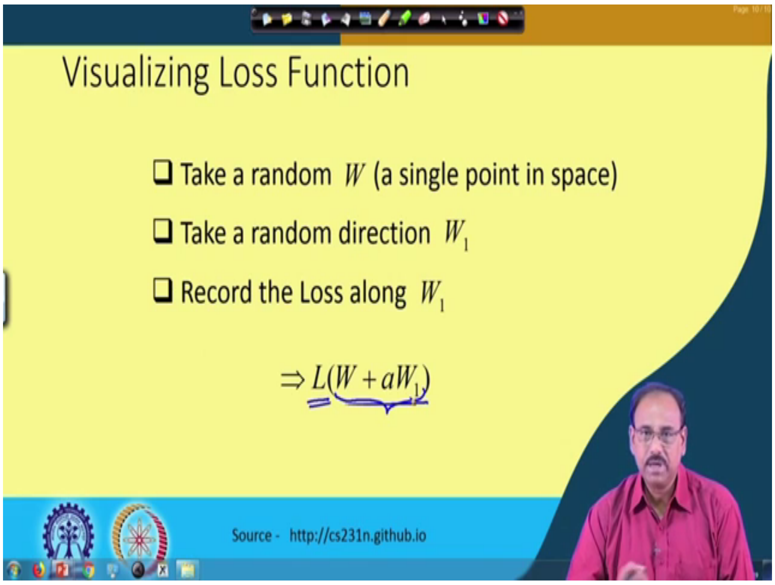
Similarly, the other component L 2 initially it remains 0 with respect to W 1, initially it remains 0 and when W 1 becomes very high in the sense that W 1 X 1 becomes more than W 2 X 2 by a factor 1 by the threshold 1, then the loss function becomes positive and it is like this. And, same is the case with component L 3. And, my overall loss function is average of all these 3 components in 1 L 2 and L 3 and the overall loss function is given by this.

So, by looking at this figure on the right which gives you the overall loss function, you find that the loss function is convex right.

So, this can be solved using convex optimization problems, because the loss function that will get is convex. Now, here the situation is very simple we can visualize it very easily, because I am considering W 1 to be a scalar or a 1 dimensional vector.

Now, what happens in case of multiple dimensions usually our weight vectors or the samples sample vectors they are of very very large dimension may be of the order of 1000s. So, the visualization of the loss function in such cases is very very difficult; however, we can try to visualize that section wise.

(Refer Slide Time: 20:55)



Visualizing Loss Function

- ❑ Take a random W (a single point in space)
- ❑ Take a random direction W_1
- ❑ Record the Loss along W_1

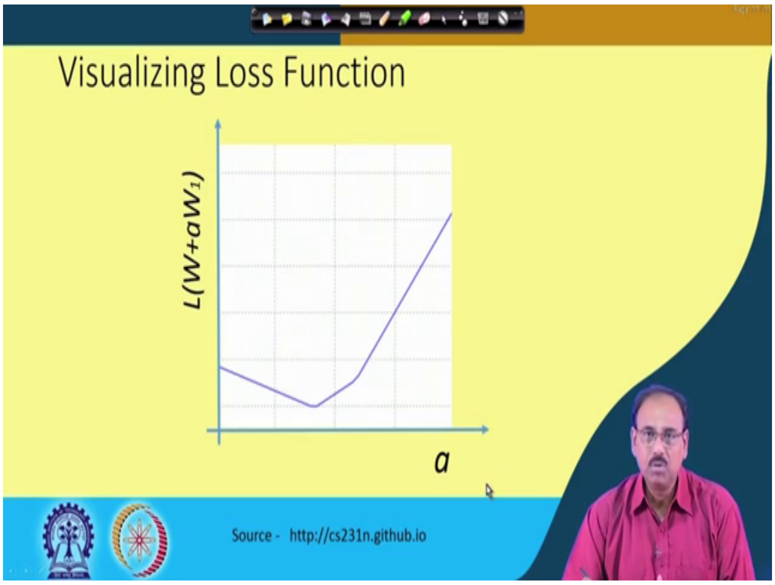
$$\Rightarrow \underline{L}(\underbrace{W + aW_1})$$

Source - http://cs231n.github.io

So, what I do is? Now, I know that my loss function is defined in high dimensional space. So, I can take a single point in that high dimensional space at random which is say W and I take a direction W 1 which is also at random. So, this W 1 I take as a direction passing through the selected point W. And, as we move along the direction of W on you go on recording the loss function, or effectively what you do is every point in the direction of W 1 passing to W is represented by this expression W plus a times W 1, where a indicates that what is the position of the point on the line W 1 passing to W.

And, we are taking the lost function L at those different points by varying a I get different points and you take the loss function W at all those different points.

(Refer Slide Time: 22:18)

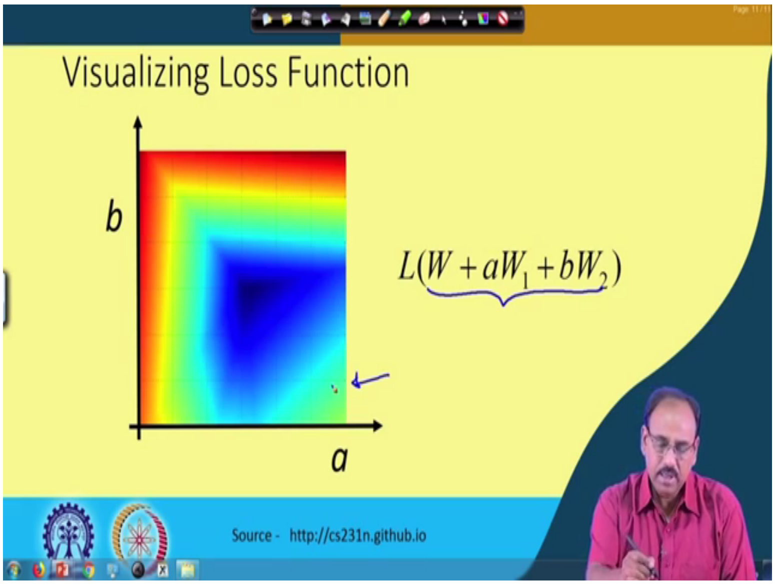


Visualizing Loss Function

Source - http://cs231n.github.io

So, what I can do is, I can now plot the loss function loss as a function of a or as a varies I get different points on the line, and now if I record the loss functions, I get a loss function, which is of this form, which is L W plus a W 1 a is the parameter, which defines which determines the points on line W 1 in the direction of W 1 passing through W. And, you will find that the loss function which will which will be of this form. I can also try to define loss function on a plane, if I take the section on a plane. So, in that case what I have to do is instead of taking just W 1 a single line, I had to take W 1 and W 2 as 2 different lines.

(Refer Slide Time: 22:57)



Visualizing Loss Function

$L(\underbrace{W+aW_1+bW_2})$

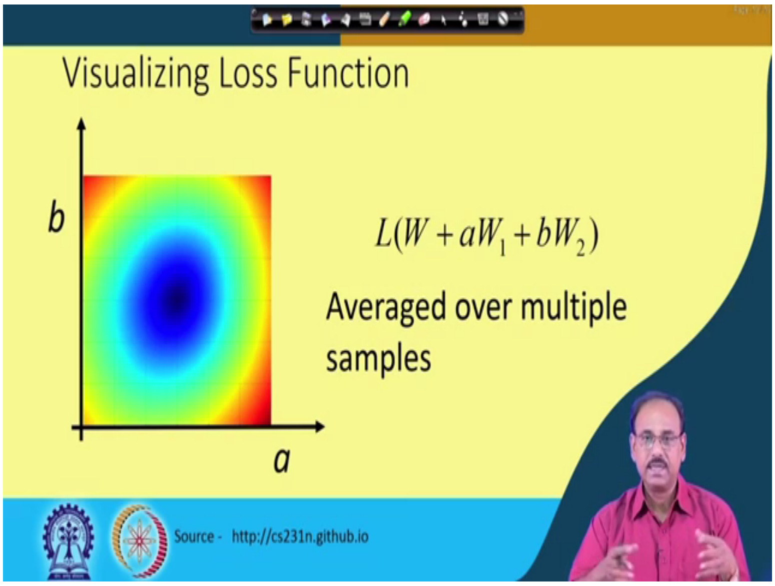
Source - http://cs231n.github.io

And, by giving that every point on that plane defined by these 2 directions W 1 and W 2, now can be determined by this expression W plus a times W 1 plus b times W 2. And, again if I record the loss value for different values of a and b, which are the parameters in this particular case I get a loss function which is given in this form. So, this is the plot of the loss function in 2 dimension.

So, in both the cases whether I take the previous one like this, but you will find that again the lost function is a convex function, it has a minimum somewhere over here or I take the next one, that is visualization of the loss function in a plane, you find that here again the loss function is a convex 1, where the blue that is in this particular case at the center, here a trace minimum and the red represents maximum loss function. So, I have the minimum of loss over here.

So, the loss function in 2 dimension, again shows that it is a complex 1. And, this is the plot that I get if I consider or if I plot the loss function only for a single sample or a single vector training vector. And, when I aware is this over multiple number of training vectors the loss function becomes something like this.

(Refer Slide Time: 24:32)



Visualizing Loss Function

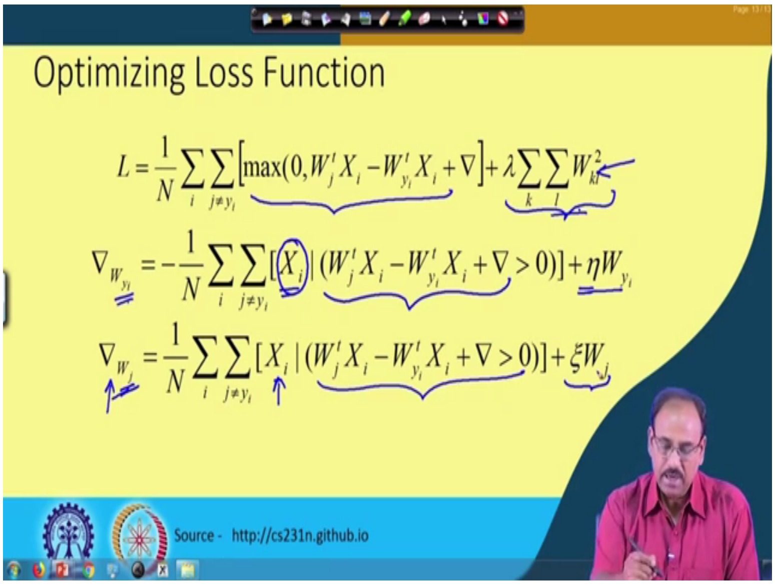
$L(W + aW_1 + bW_2)$

Averaged over multiple samples

Source - http://cs231n.github.io

So, you find that this averaging over all the training samples smoothes the nature of the loss function.

(Refer Slide Time: 24:49)



Optimizing Loss Function

$$L = \frac{1}{N}\sum_i \sum_{j \neq y_i}\left[\underbrace{\max(0, W_j' X_i - W_{y_i}' X_i + \nabla}\right] + \lambda \underbrace{\sum_k \sum_l W_{kl}^2}$$

$$\nabla_{W_{y_i}} = -\frac{1}{N}\sum_i \sum_{j \neq y_i}[\boxed{X_i} \mid (\underbrace{W_j' X_i - W_{y_i}' X_i + \nabla} > 0)] + \underline{\eta W_{y_i}}$$

$$\nabla_{W_j} = \frac{1}{N}\sum_i \sum_{j \neq y_i}[X_i \mid (\underbrace{W_j' X_i - W_{y_i}' X_i + \nabla > 0})] + \underbrace{\xi W_j}$$

Source - http://cs231n.github.io

So, once I have this loss function, next what I need to do is I have to optimize this loss function or I have to minimize the loss function. So, for minimization as we have done earlier we take the gradient descent approach. So, I have to take the gradient of the loss function that I have and a modified W in the direction of the negative gradient.

So, here what I have is as you have already said that the overall loss function is given by this where this is the data loss component and this is the regularization loss component if I take. So, I have to optimize this loss function in order to find out the value of W for which this loss function will be minimum. So, I take the gradient of this loss function with respect to W y i, I also take the gradient of this loss with respect to W j.

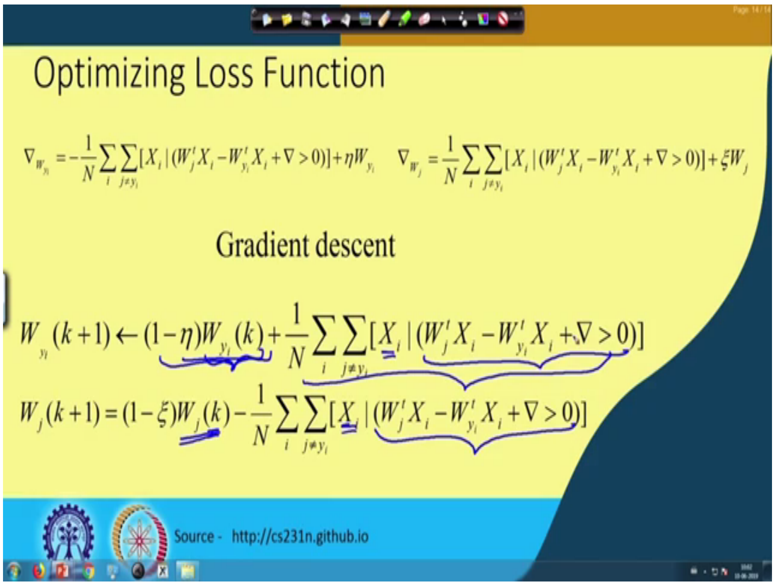
So, when you take the gradient of this loss function with respect to W y i you find that the expression of the gradient will be that it is sum of X i, that is gradient of loss function with respect to W y i is nothing, but sum of X i only in those cases where W j transpose X i minus W y i transpose X i plus delta is greater than 0.

And, because in all the cases so, at W transpose j transpose X i minus W y i transpose X i plus delta is less than 0, the loss function was 0. So, for those cases those X i S are correctly classified by or W. So, in such cases I need to not modify the weight matrix W. So, this gradient only takes the sum of all those X i all those training vector for which W j transpose X i minus W y i transpose X i plus delta is greater than 0; that means, these are the that X i leads to an error plus if you take the gradient of this term this gradient will be actually twice lambda times W y i.

So, in this case it has written with respect to another constant say eta, eta times W y i. In the same manner, if you take the gradient with respect to W j, then it also becomes sum of y i where W j transpose X i minus W y i transpose X i plus delta is greater than 0.

So, we will take the sum of only those training vectors for which in this condition is true we will not consider those training vectors for which this condition is not true; that means, those training vectors are correctly classified for the kind by the current W. And, plus from this regularization term when you take the derivative of this regularization term with respect to W j or take the gradient of this regularization term with respect to W j, the term that I get is zeta times W j.

(Refer Slide Time: 28:37)



Optimizing Loss Function

$$\nabla_{W_{y_i}} = -\frac{1}{N}\sum_i \sum_{j \neq y_i} [X_i \mid (W_j^t X_i - W_{y_i}^t X_i + \nabla > 0)] + \eta W_{y_i} \quad \nabla_{W_j} = \frac{1}{N}\sum_i \sum_{j \neq y_i} [X_i \mid (W_j^t X_i - W_{y_i}^t X_i + \nabla > 0)] + \xi W_j$$

Gradient descent

$$W_{y_i}(k+1) \leftarrow \underbrace{(1-\eta)W_{y_i}(k)} + \frac{1}{N}\underbrace{\sum_i \sum_{j \neq y_i} [X_i \mid \underbrace{(W_j^t X_i - W_{y_i}^t X_i + \nabla > 0)}]}$$

$$W_j(k+1) = (1-\xi)W_j(k) - \frac{1}{N}\sum_i \sum_{j \neq y_i} [X_i \mid \underbrace{(W_j^t X_i - W_{y_i}^t X_i + \nabla > 0)}]$$

Source - http://cs231n.github.io

So, these are the gradients and using these gradients we go for our optimization step, or gradient descent step, where we get the gradient descent as and the k th instant if my weight vector was W y i k, then the next iterated value of W y i is becomes W y i k plus 1, which is 1 minus eta times W y i K plus 1 over N sum of X i for all those X i, which satisfies this condition.

Similarly, the iterated value of W j at instant k plus 1 from instant k from the iterative state k is given by W j k plus 1 is equal to 1 minus zeta times W j k minus 1 over m sum of all those X i for which this condition is satisfied. So, when we will not consider those sample vectors, which it does not satisfy this condition, because we assume that those vectors are correctly classified by the W j at the k th instant ok.

So, if you look at these 2 gradient descent steps you find that what it indicates is first you are modifying W y i, which was there at the k th instant by the regularization term. So, this is a component which is coming from you are the regularization error part or regularization loss part. And, this is the component which comes from your data loss part.

So, that is how iteratively you go on optimizing or minimizing this loss function and when it converges the value of weight matrix that you get gives you the linear machine or the support vector machine for multiple classes. So, we will stop here today's lecture will come back in the next day.

Thank you.