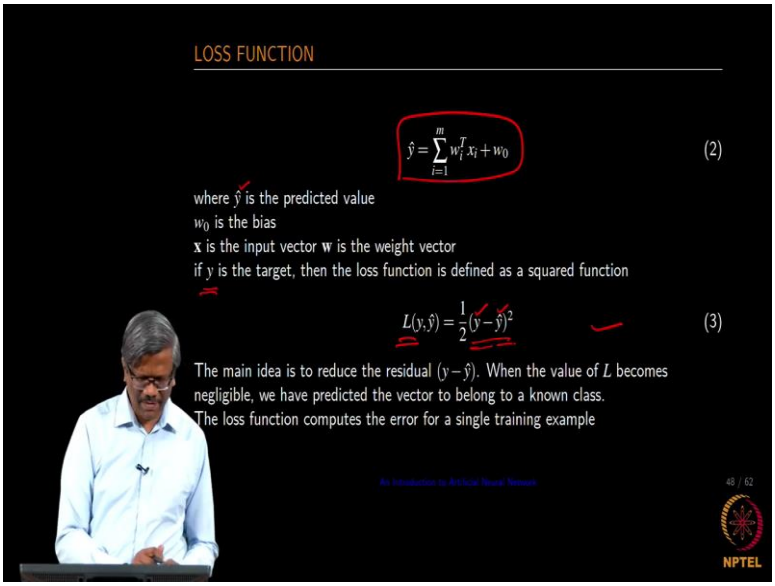
**Applied Natural Language Processing**
**Prof. Ramaseshan Ramachandran**
**Department of Computer Science and Engineering**
**Indian Institute of Technology, Madras**

**Lecture - 37**
**Gradient Descent**

(Refer Slide Time: 00:15)




So, we saw this set of activation functions, and now we will also talk about what is a loss function and a cost function. In many cases, you would notice that the laws and costs are interchangeably used, but I would like to separate this out a little bit, ok. The last function is computed based on the mean square error you know what is the output that you are computing and what is the target, ok.

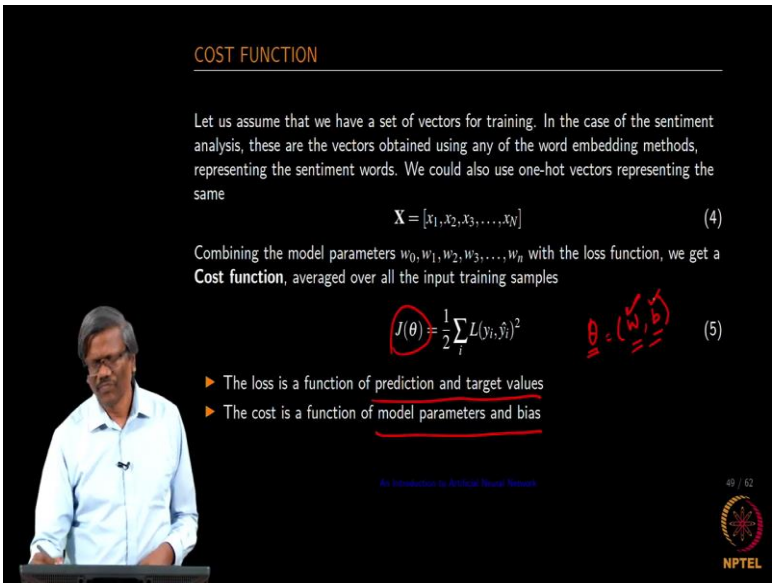
$$\hat{y} = \sum_{i=1}^{m} w_i^t \; x_i + w_o$$

$$X=(x_1, x_2 \ldots \ldots x_n)$$

So, here this you know now very well, right. So, we have the predicted value computed using equation 2. And w naught is your bias and x is the input vector and w is the weight vector and the if y is the target that we are looking at. The loss is computed is in the

mean square equation as follows. The idea is to minimize this loss, correct. So, when it becomes negligible we can say that we can stop, we can tell this system that stops the iteration, ok.

(Refer Slide Time: 01:25)



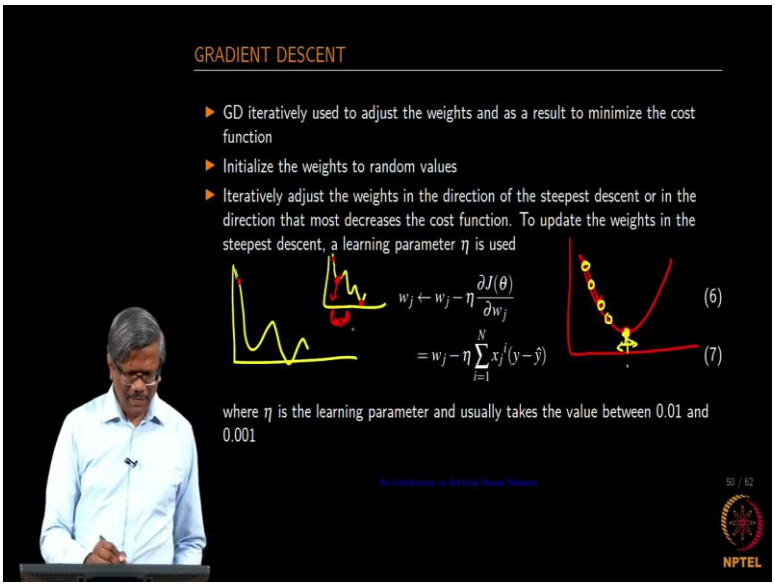

The cost function is a little different, if you look at the difference here let us talk about the loss function is a prediction and the is a function of prediction and target values, right.is shown below

$$L(y, \hat{y}) = \frac{1}{2}(y - \hat{y})^2$$

In the cost function, what we do is we just call this as the function of model parameters and bias. This theta represents your w and the bias. This is the parameter that these are the parameters we are trying to estimate, right, in the model. So, this going to represent the parameters that are going to be estimating. So, it is usually represented as J theta, ok.

It is very similar to what we saw earlier, but in this case this will represent only the; the computation is very similar to what we have done in the last function, ok, but there is a slight change in the representation here.

(Refer Slide Time: 02:38)




And another one, I think I spoke about this little bit earlier we would be using gradient descent in order to find the local minima. I will be talking about the local minimum most of the time. In many cases you know the though ideally, this is what we expect and then our minimization problem comes down in the negative slope and then finally, reaches the endpoint here, the endpoint, ok.

$$w_j \leftarrow w_j - n\,\mathrm{J}(\theta)$$

$$\mathrm{J}(\theta) = \frac{1}{2}\sum_i L\,(y - \hat{y})^2$$

But in the, so you may also get something like this or you may find the error function going like this. Most cases you know when you follow the gradient descent ideally I want this to come down the slope and then finally when it reaches the minimum there will be no more change that is when we know the value of the gradient will become very small or be equal to 0. We say that, I reached the end of my weight computation, ok.

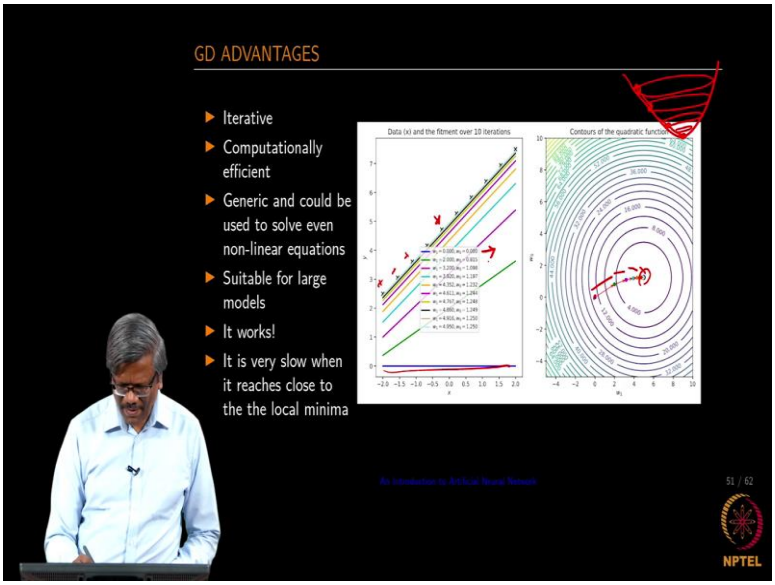
So that means, I can stop my iteration. We do not get into that many times. We will come here and then get to this point and think that this is my minima, but my global minima are somewhere here. Since, the algorithm will start finding the increase in the error early after some point you think that it is not doing good, let me stop here and we call this as

the local minima, ok. So, we will not worry about this at this point in time. I just thought I would tell you about what is local minima and global minima.

Gradient descent is usually used to get to the minimum value here. So, how do we get to that value? Using the iterative process where we keep changed the value of the weight based on the cost function, ok. And then we have seen that eta is also your learning parameter which is adjusted to either 0.1, sometimes it could be even 0.1, 0.01, 0.001 depending on your application.

We normally start with the initialization using random values. All the weights are just are to have random values and then the weights are adjusted in the direction of steepest descent or in the direction that decreases the cost function and then we know that how to compute this. We will talk about this when we go into the artificial neural network where we have multiple layers, ok. And assume that for now that we are going to be using gradient descent algorithm, that is going to take us to the minima or let us call it as the minimum of this function, ok.

(Refer Slide Time: 06:28)




Again you will see the representation of the same thing in different ways. This is a one-dimensional representation. You will also see the quadratic contours. It is seen from the top, ok, so what we have is a side view of this function. This is seen from the top, you will see the alternative process moving in from our sliding into the minima from here. So, it is like this, ok. And when you see from the top you would see the contours in this

fashion, and then we are coming down from here to here to here and that is what is shown here in this fashion.

So, it is an iterative process. And also you can see that the decision function also is moving slowly and steadily closer to the actual one which is provided as plus corresponds to the original data point and that could be obtained by an iterative process starting from this. So, initially, you have some random values, in this case, it is 0, I think we spoke about that earlier.