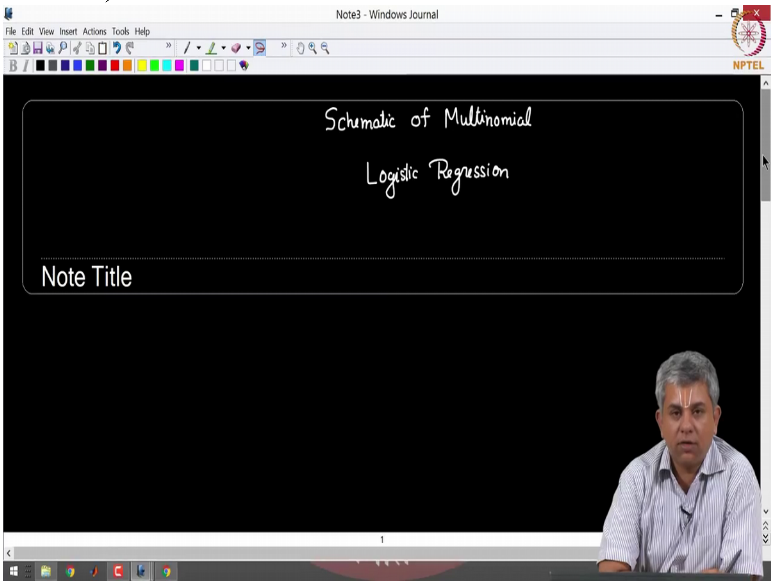
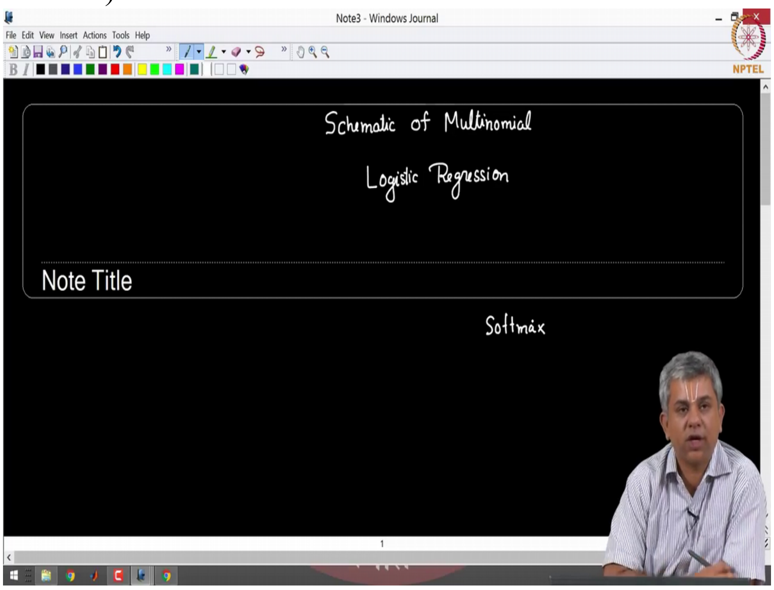
Machine Learning for Engineering and Science Applications
Professor Doctor Balaji Srinivasan
Department of Mechanical Engineering
Indian Institute of Technology Madras
Schematic of Multinomial logistic regression

(Refer Slide Time: 00:12)



Welcome back. In the previous videos we had seen how to use logistic regression for multiclass problems. We had done that using a Softmax function if you remember. We had also looked at what the corresponding

(Refer Slide Time: 00:34)

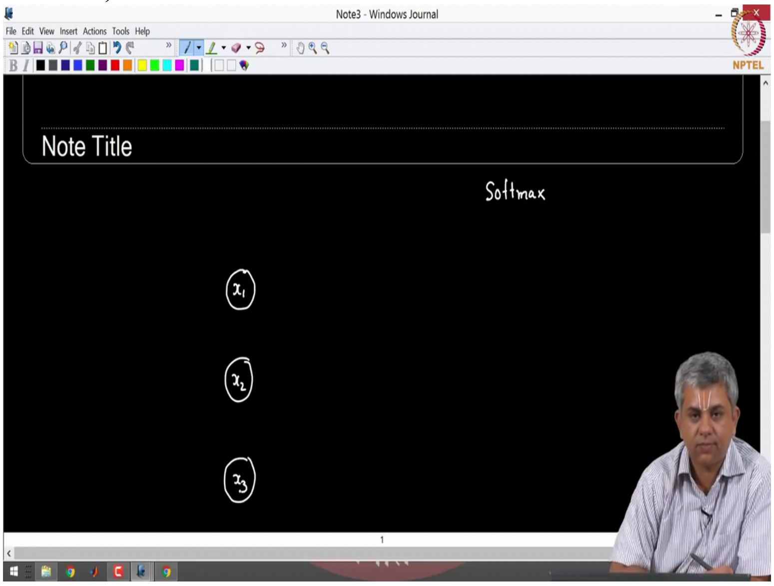


loss function was, etc.

In this video I want you to see a simple schematic which will also tell you how exactly a matrix comes when you deal with weights with multiple classes, when we have multiple classes in multinomial logistic regression.

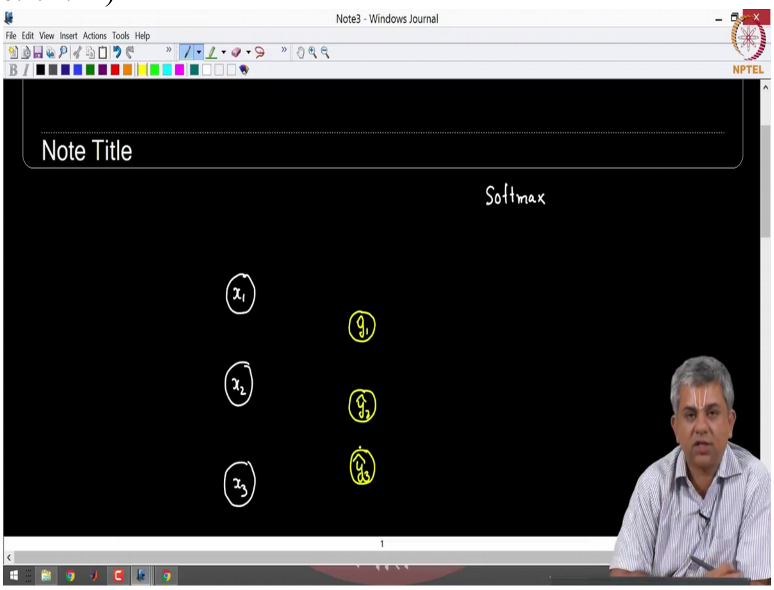
So let us consider a simple example. Let us say I have 3 input features.

(Refer Slide Time: 01:07)



And let us say I have 3 output features also, Ok, y 1 hat, y 2 hat, y 3 hat. So

(Refer Slide Time: 01:22)

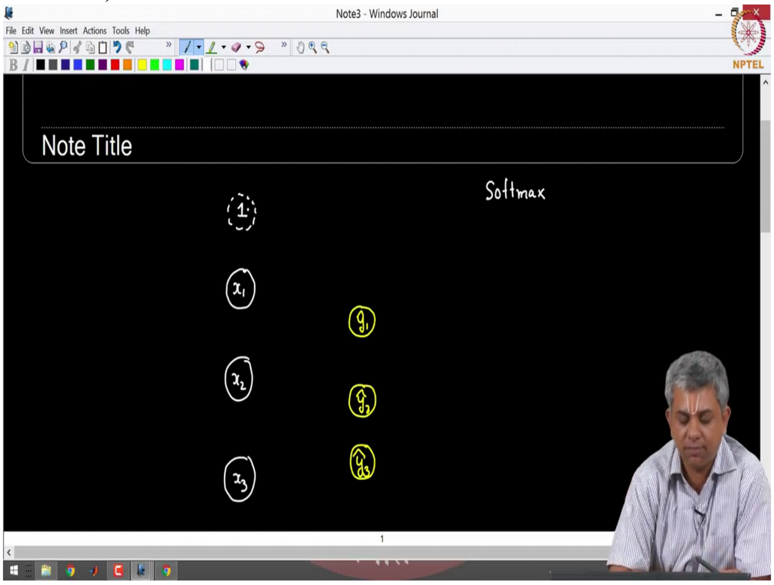


let us say this is a 3 classification problem. You can think of multiple examples for this.

For example if I give height, weight and age, suppose you want to find out whether this person has no probability of heart disease or low probability of heart disease, medium probability of heart disease or high probability of heart disease, this is not quite a classification problem but just as an example I can give you this.

We will look at several examples or at least a few examples in the examples week which will be around week 9 or so. So you can think of any convenient example for yourself. And now let us introduce our usual bias unit which is 1 or x naught

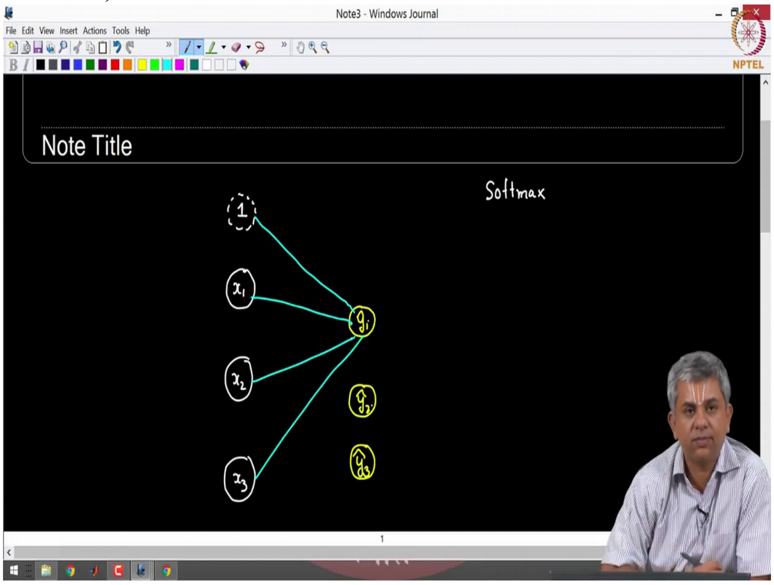
(Refer Slide Time: 02:04)



and now we want to find out what is y 1 hat, y 2 hat, y 3 hat.

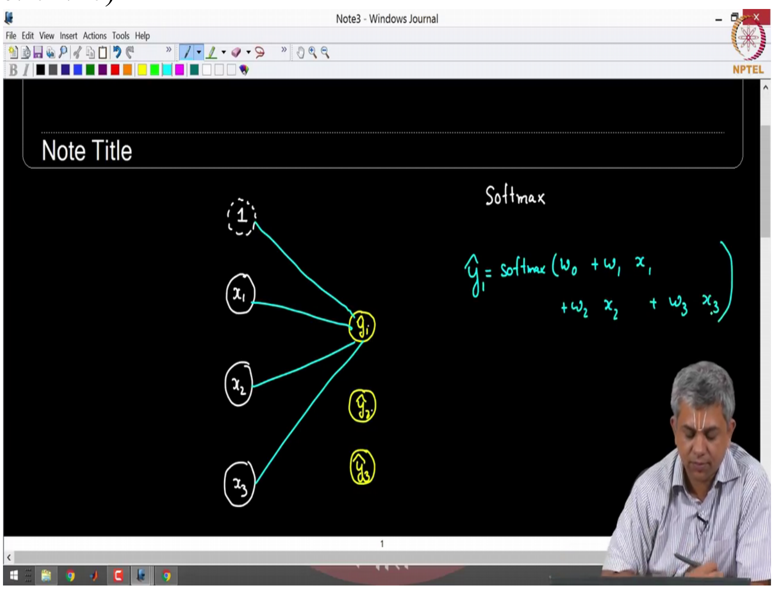
So the portion that we are doing right now is the forward model. So as usual y 1 hat

(Refer Slide Time: 02:19)



is equal to Softmax of the linear combinations of this, w 0 plus w 1 x 1 plus w 2 x 2 plus w 3 x 3, Ok.

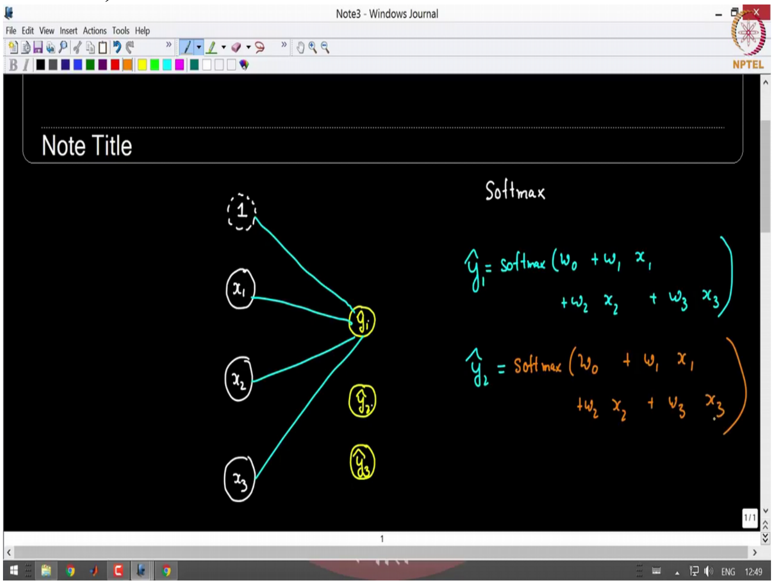
(Refer Slide Time: 02:40)



That would be y 1 hat. Now suppose I have y 2 hat.

Now y 2 hat is also Softmax of some linear combination, w naught let us say plus w 1 x 1 plus w 2 x 2 plus w 3 x 3. Now suppose

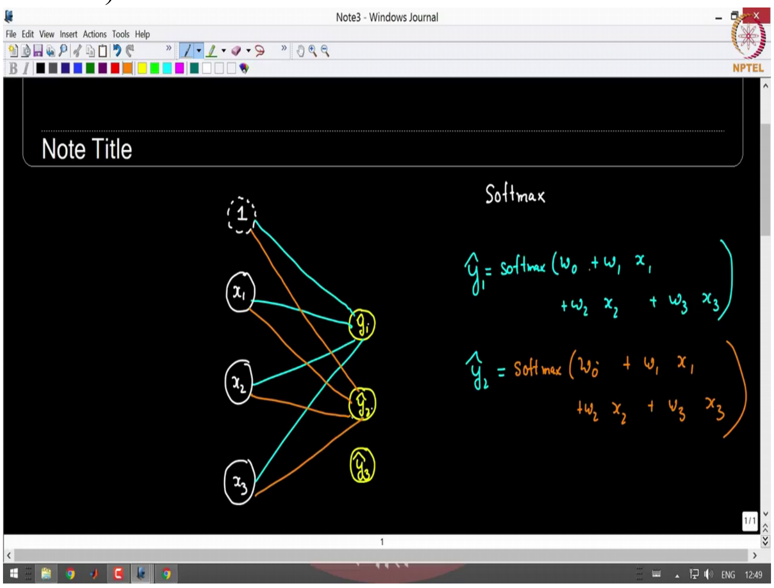
(Refer Slide Time: 03:03)



this w naught, w 1, w 2, w 3 were the same in both these cases, obviously you are going to get the same y 1 hat as well as y 2 hat. Because otherwise the functions are identical.

So this is not a good idea. So you need different weights. So we are going to use different weights here, Ok.

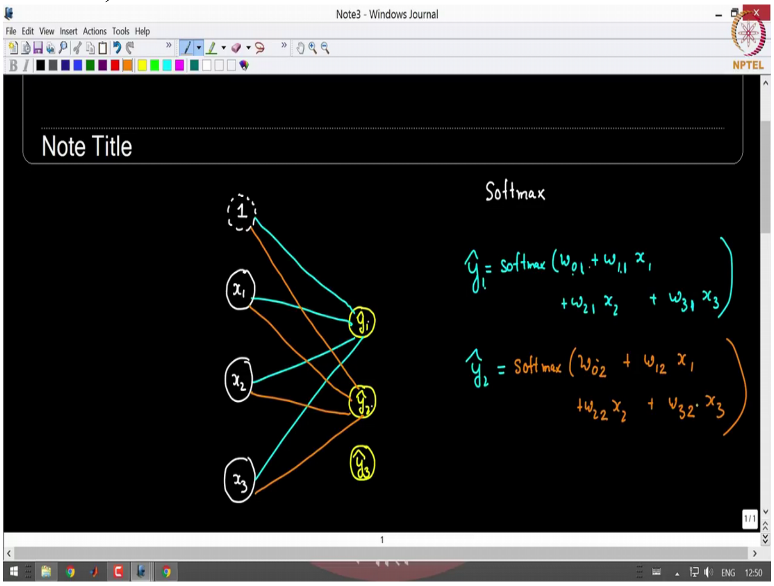
(Refer Slide Time: 03:31)



So we need some terminology in order to distinguish these two weights. So I will call it w naught 1, w 1 1, w 2 1, w 3 1 where the 1 stands for the output and the 0, 1, 2, 3 actually stand for the input.

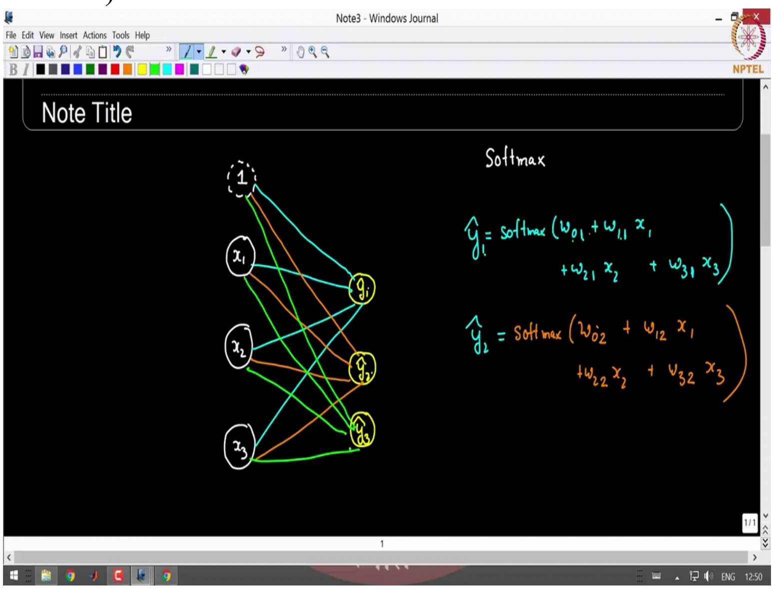
Similarly you can easily see that now this should be w naught 2, w 1 2, w 2 2 and w 3 2.

(Refer Slide Time: 03:56)



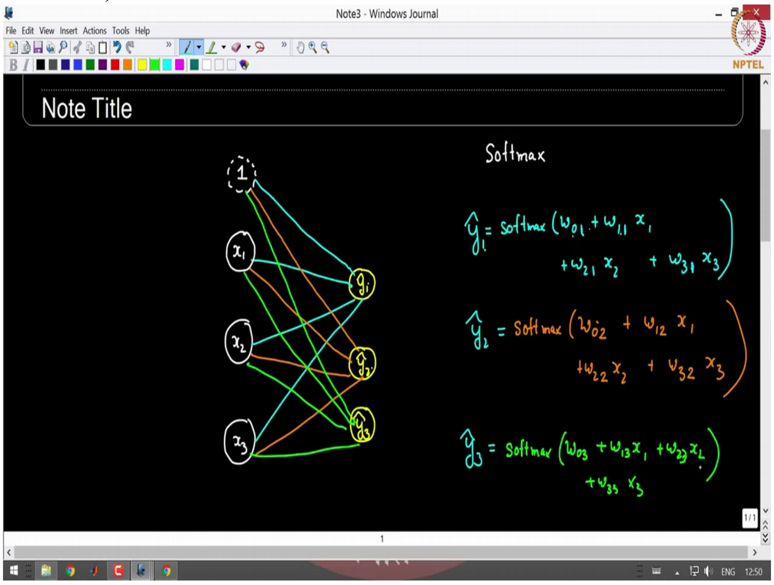
Finally if we come here, I need another set of weights.

(Refer Slide Time: 04:08)



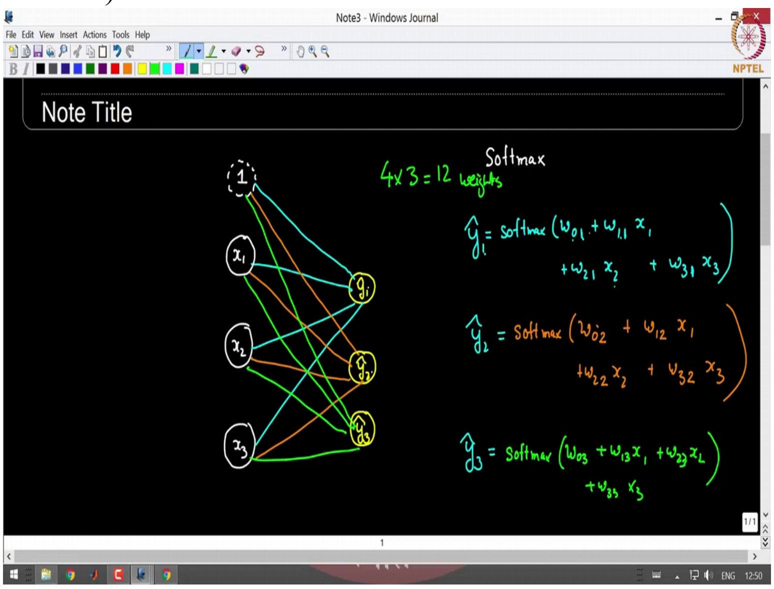
So y hat 3 would be Softmax of w naught 3, w 1 3 x 1, w 2 3 x 2 plus w 3 3 x 3.

(Refer Slide Time: 04:32)



So how many weights do we have? 4 unique weights in each one of these, so you have 4 into 3,

(Refer Slide Time: 04:43)

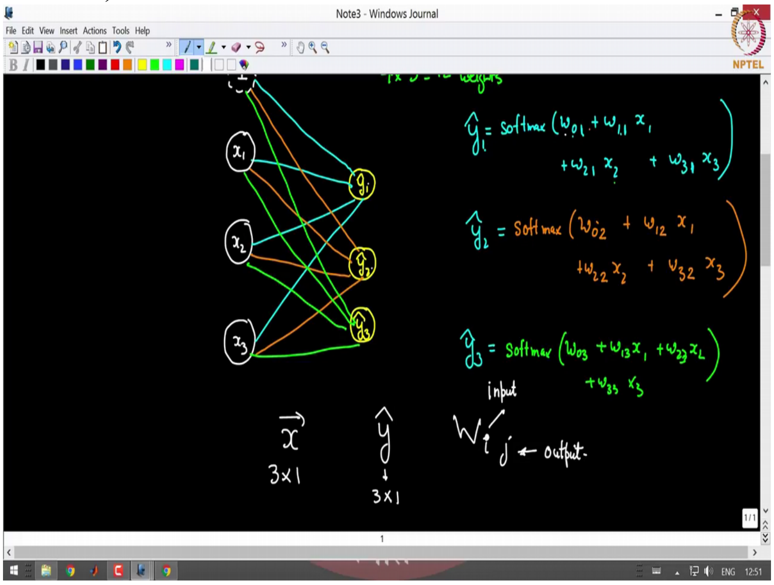


12 weights in order to account for the bias term also.

So how would we write this matrix wise? So we have x vector which was 3 cross 1, we have y hat which is also 3 cross 1 and we have w which is now a weight matrix.

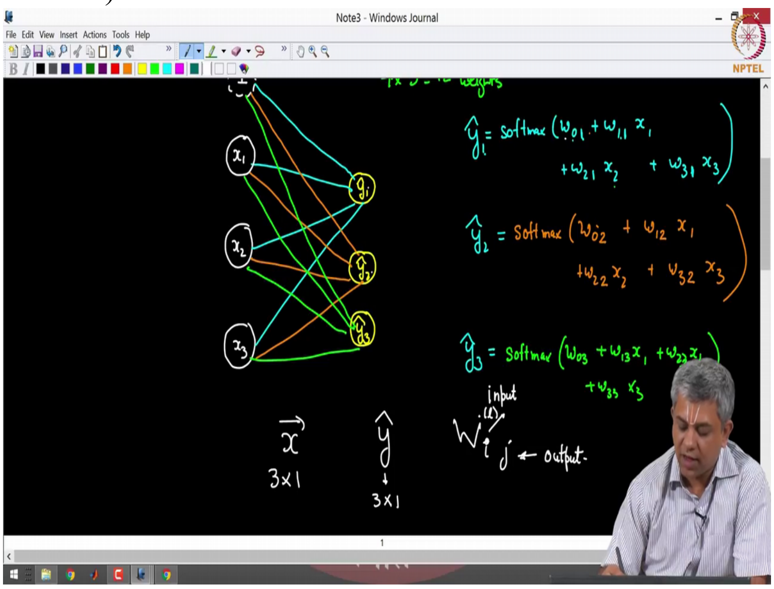
You can see w has 2 indices. w i j

(Refer Slide Time: 05:22)



has i as the input feature, j is the output feature. You had also seen in the earlier video with XOR that you could have more than 1 layer. In that case typically

(Refer Slide Time: 05:33)

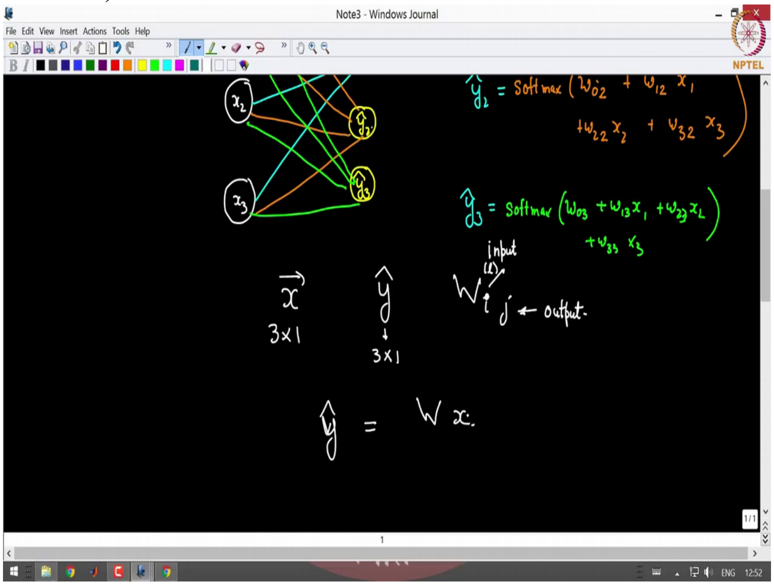


we add an L here which denotes the level. So you could have w i j 1, w i j 2, w i j 3 etc, etc.

So you will have multiple weights. So this is the large number of weights as you will see in next the, in the next, the videos in the next week when we come to convolutional neural networks, you have billions and billions of parameters in usual practical neural networks that sit in today which is why they are extremely powerful, Ok.

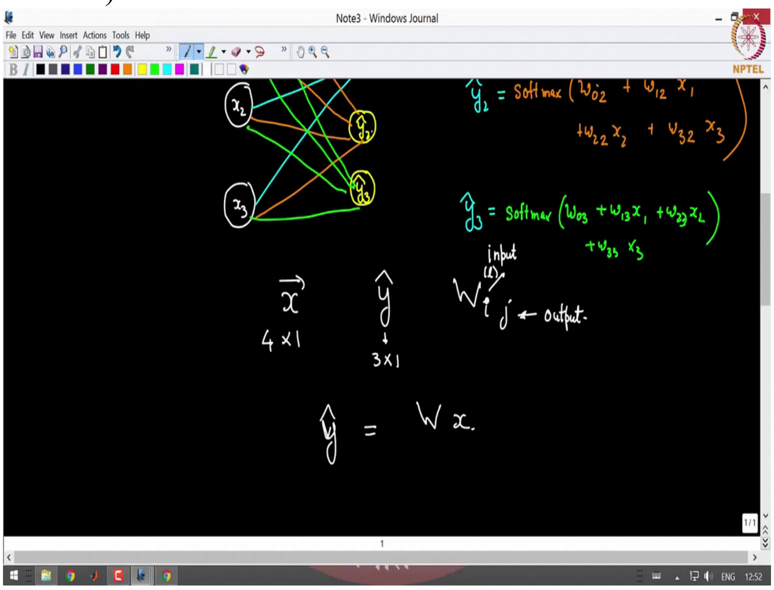
Coming back to this, if we want to write y hat as w times x,

(Refer Slide Time: 06:12)



so just for this case I will make x as 4 cross 1 so that

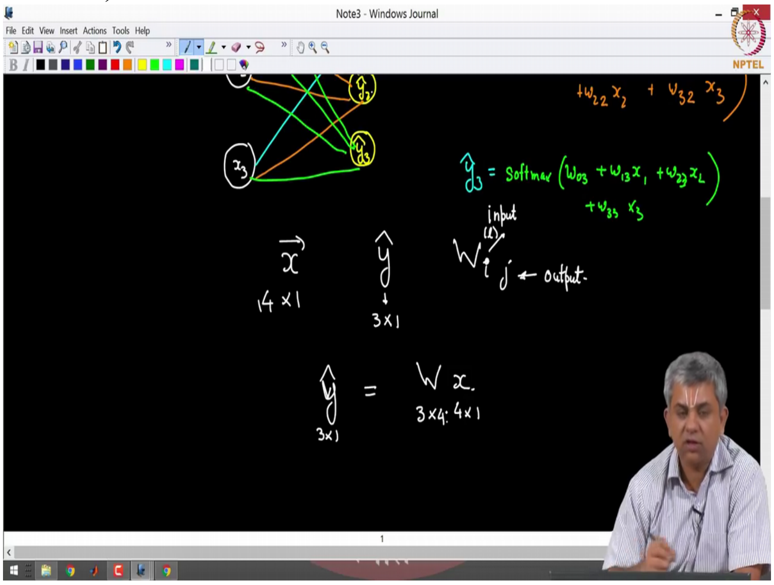
(Refer Slide Time: 06:21)



the 4 includes our bias unit also.

So you can write y hat as W x. x will be 4 cross 1, y will be 3 cross 1. So if you want an appropriate W, please imagine what W should be. This should be

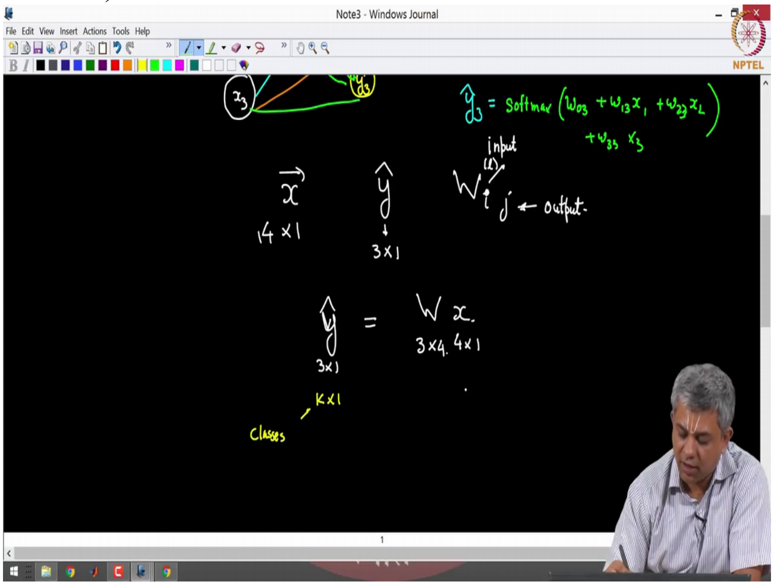
(Refer Slide Time: 06:43)



3 cross 4.

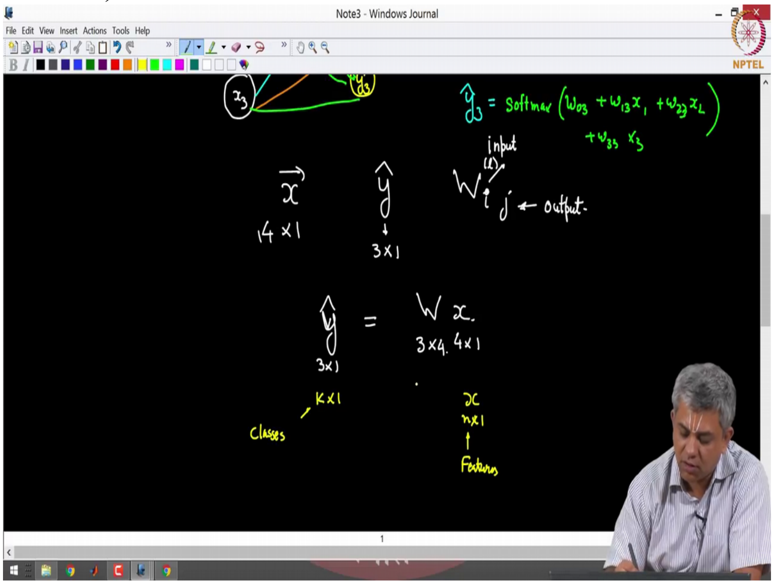
In the general case, if y hat is k cross 1 where k is the number of classes

(Refer Slide Time: 06:56)



and x is n cross 1 where n is the number of features.

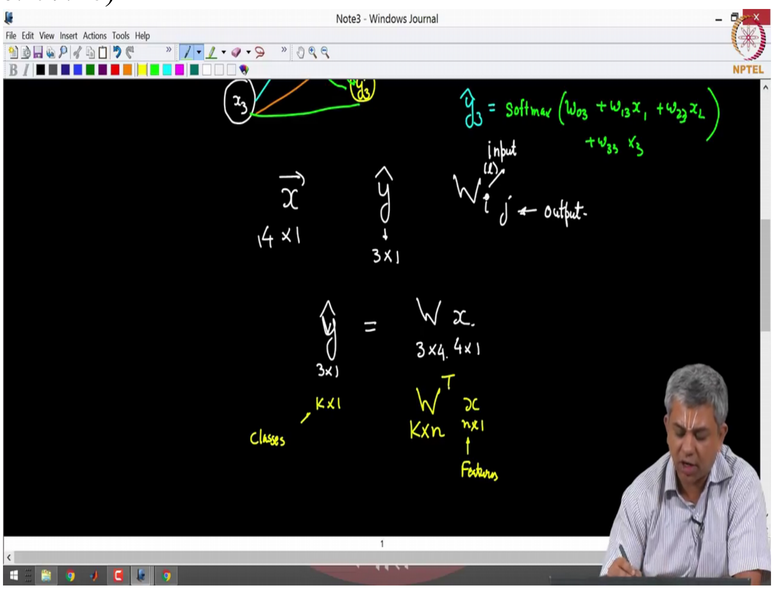
(Refer Slide Time: 07:05)



Then w should have the size k cross n, Ok.

Now there are some people who will denote this w as

(Refer Slide Time: 07:16)

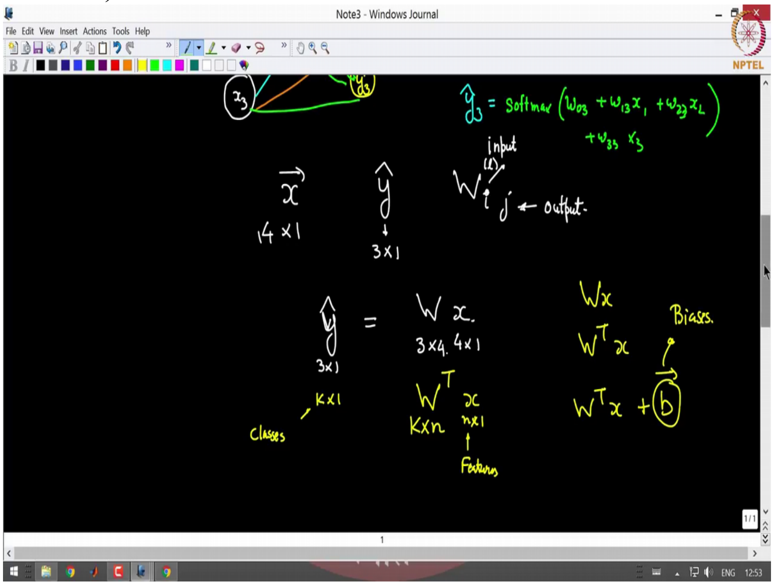


W transpose so as to be consistent with the notation I have used here.

So you might see this at multiple places sometimes, you will see w x, sometimes you will see w transpose x. Sometimes you will also see w transpose x plus B where B is a vector, the vector of biases.

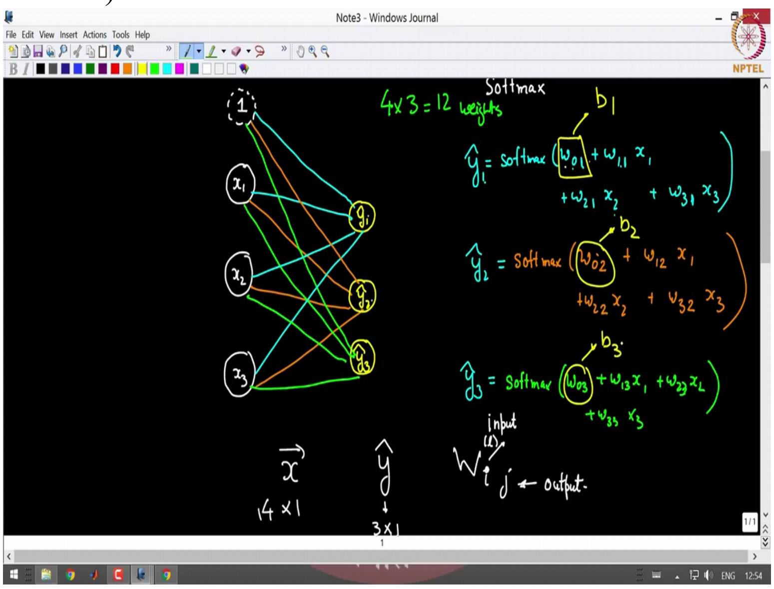
So just to

(Refer Slide Time: 07:38)



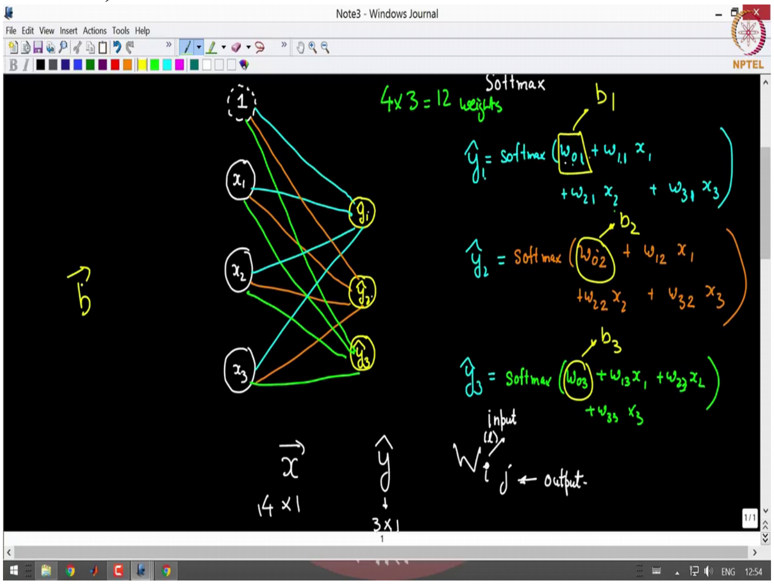
clarify this notation for you, please notice, if we remove the bias separately this will become b 1, this will become b 2 and this will become b 3.

(Refer Slide Time: 07:50)



b vector

(Refer Slide Time: 07:53)



is separated and w x is separated in such notations.

So we will be using this kind of notation. As I said before we will be using this interchangeably especially when it comes to future videos and future weeks, thank you.