

Inverse Methods in Heat Transfer
Prof. Balaji Srinivasan
Department of Mechanical Engineering
Indian Institute of Technology, Madras

Lecture - 49
Logistic Regression Binary Entropy Cost Function and Gradient

(Refer Slide Time: 00:19)

Logistic Regression

- Purpose - Binary Classification
- Data Representation :
Input : \vec{x} → As usual
 y → Scalar (0 or 1)
 $\hat{y} \in [0, 1]$. Represents $\text{prob}(y = 1 | x)$
- Forward Model
 $\hat{y} = w_n$

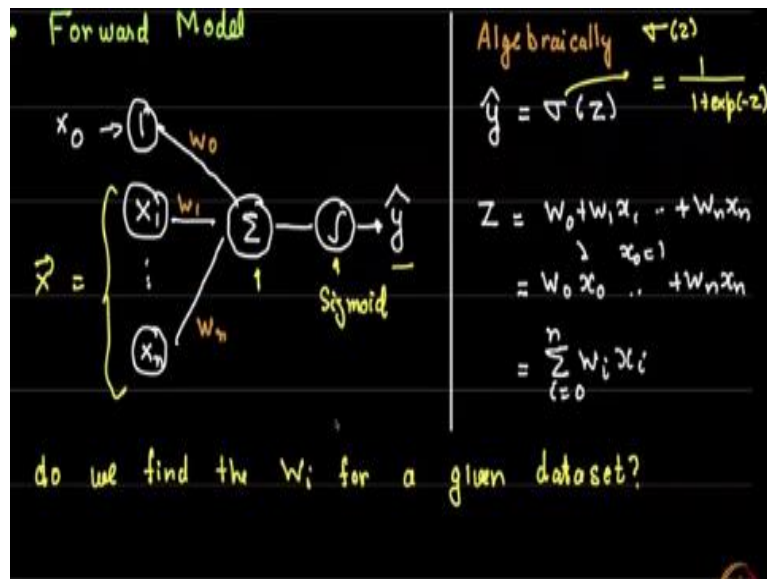
Algebraically $\hat{y} = \sigma(z) = \frac{1}{1 + \exp(-z)}$

Welcome back. We are on week 9 of inverse methods in heat transfer. In this video I want to continue where I left off in the last video which was for logistic regression, where we had discussed the forward model. So, remember what logistic regression the model is for. It is for binary classification. Binary classification is identifying which of two classes some things some object or some event or some image that you have given belongs to.

Now in this remember I had talked about four things that we require for machine learning, data representation, we require the forward model and we require the optimization algorithm and we require the loss function. Now for the forward model we need the data representation and as was discussed in the last video we give x as usual and y is described by a scalar which is either 0 or 1, \hat{y} on the other hand is a real number which lies between 0 and 1.

Notice this is binary and \hat{y} itself is a real number. So, that is because this represents a probability that y belongs to class one. For example, if one represents spam then \hat{y} would be the probability that email is spam.

(Refer Slide Time: 01:40)



We also saw the logistic regression forward model. The forward model is fairly simple if you have some x vector which has multiple features which are from x_1 through x_n , it has multiple components. For example, I gave you the example of body temperature and cough but you could think of temperature, pressure, velocity anything of that sort. So, any n features that we have all the forward model does is fairly simple does a linear combination and if it ended here, it would simply be linear regression.

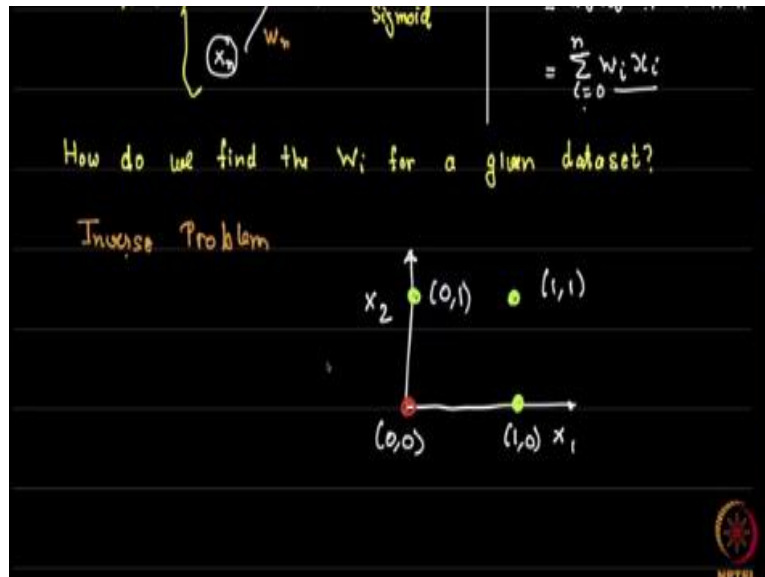
But then since we want the output between 0 and 1 and we do not want a real number in any range. We actually add this sigma function at the end. The sigma is basically turns serves two purposes turns it between 0 and 1 and you can also interpret now the output as a probability. Algebraically we can write the same thing as what is written on the right-hand side here. So, you can write this as,

$$z = w_0 + w_1x_1 + \dots + w_nx_n$$

And I have taken the liberty of putting w_0 as w_0x_0 just so that it looks symmetric where x_0 basically is the constant one. So, you can think of this as x_0 . so we can write this as the summation

between 0 to n of $w_i x_i$. The question now is this we already can do the forward model therefore given an x and given a w you can find out \hat{y} . The question is given a data set can you find out w_i . So, this of course is our standard inverse problem.

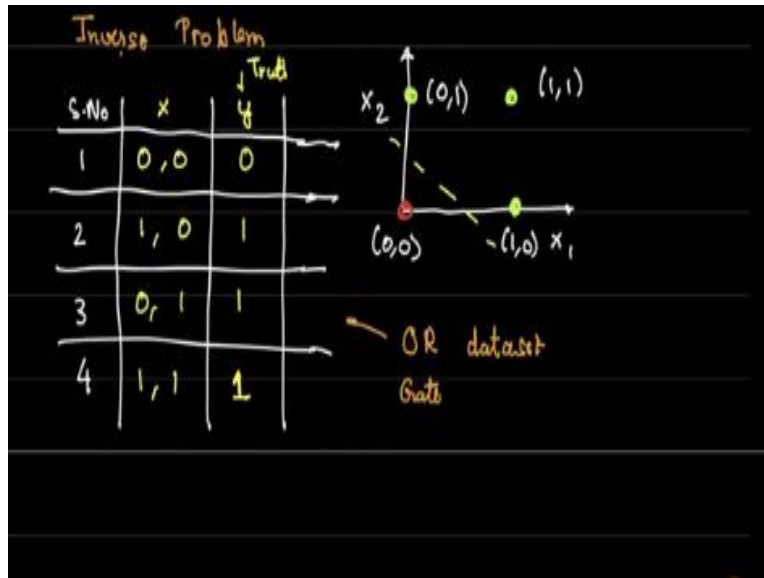
(Refer Slide Time: 03:26)



Now I am going to take a very simple example. This is an example like this I will first show the figure and I will give it in terms which are a little bit more familiar to you, let me redraw this. So, let us say this is instead of calling it x and y let us say there are two features x_1 and x_2 and we have our output. Our output is like this, you get a green at three locations and you get a red at one of the locations.

Now these three locations are like this, this is the only data set. So, this is 1, 1 this is 1, 0 this is 0,0 and this is 0,1.

(Refer Slide Time: 04:18)



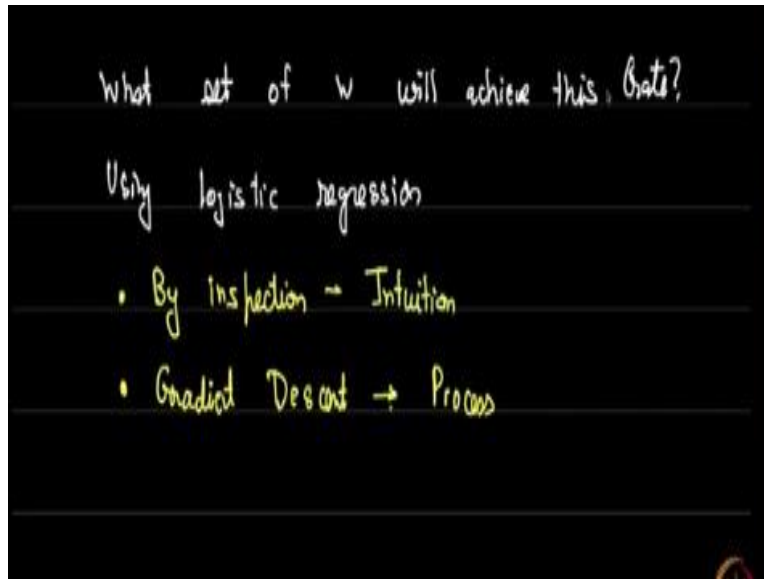
If we write it as a table, the table looks like this. You are simply just four data points 1, 2, 3, 4 not hundreds not thousands just four data points and we want to classify line for this. Of course, the classifying line you can imagine would be something of a sign. We will come back to this that is where we want to land up. So, here is x , x is if it is 0, 0 then the output is 0, if it is 1,0 the output is 1, if 0,1 the output is 1, I mean if this 1,1 the output is 1.

This is the ground truth and we want a classifying line which will classify this and show that if you give any new point. Let us say give a point here it can be located correctly as either 0 or 1. Now what does this remind you of this data set is of course a simple OR data set. So, you can imagine this just as the OR gate. Historically this was a very important example as far as logistic regression will not really logistic equation but neural networks were concerned.

The idea was this, the idea was that if you can simulate the basic gates basic logic gates OR gate, AND gate or NOR gate etcetera, then you can basically simulate any gate at all and therefore anything that a computer can do can be simply achieved using logistic regression or simple networks. As we will see just like linear regression, I showed you that can be interpreted as a neural network model.

Logistic regression is also an example of a neural network model. Now given this data set let us ask a question what set of w will achieve this gate. So, what set of w will achieve this gate using logistic regression.

(Refer Slide Time: 06:23)



Now I am going to do this in two ways one of this is purely by inspection. What does that mean we will just guess some weights and see whether we can make it work and the second is I will show you how to do it through gradient descent. Now I want to warn you when I do it through gradient descent, I will not give you the actual ways that I will give you here. In this case I will just give you the process.

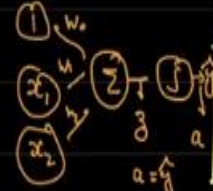
So, first we will try to do this purely by intuition and inspection. So, let us look at this data set again.

(Refer Slide Time: 07:11)

	x	y	\hat{y}
	0,0	0	<0.5
	1,0	1	>0.5
	0,1	1	>0.5
	1,1	1	>0.5

$\hat{y} = g(z)$
 $\sigma(z)$

$z = w_0 + w_1x_1 + w_2x_2$



The data set looks like this. I will draw it bigger in order for this to be convenient 0 0 I get a 0, 1, 0 I get a 1, 0, 1 I get a 1 and 1, 1 I again obtain of 1. So, this is my y. Now here it is I want my y-hat. Now by inspection what I know is this my model is $\hat{y} = g(z)$. Remember the logistic regression model, $z = w_0 + w_1x_1 + w_2x_2$. Again, for your convenience let me just draw this. Since you are starting this out, I will redraw the model just so that you remember it and here is like that.

The output that comes out at this end is called z, at this end is called a and a is basically \hat{y} . We have w_0, w_1, w_2 . Now if I want the output to be 0 you will remember from my previous videos that this means that really speaking this one will never be exactly zero because this is simply a sigma i. So, g was simply a sigma I had have to say. So, what we can expect is this is greater than 0.5 sorry less than 0.5 this is what we did for the success failure cases.

If we get the probability that it belongs to class one is very small, it is less than 0.5 then I would have classified it as a zero. I will come to a clear picture representation of this later. But these two should be greater than 0.5. So, we have identified in this case that \hat{y} should be less than 0.5.

(Refer Slide Time: 09:19)


$$\hat{y} = g(z) \quad z = w_0 + w_1 x_1 + w_2 x_2$$

$$\sigma(z)$$

But, $\hat{y} < 0.5$, $g(z) < 0.5$

$$z < 0 \quad \frac{1}{1+e^{-z}} < 0.5$$

$e^z > 1 \Rightarrow z < 0$



But \hat{y} less than 0.5 means that $g(z)$ is less than 0.5, $g(z)$ is $\frac{1}{1+e^{-z}}$ is less than 0.5. This will require for e^{-z} to be greater than 1, this requires z is negative. So, this we know, we know that the failure cases or on one side they are on z less than 0. So, for example in this figure one side of it, let us say this side of it is z less than 0 this side of it is z greater than 0.

(Refer Slide Time: 10:10)

$$z < 0 \quad \frac{1}{1+e^{-z}} < 0.5 \Rightarrow e^z > 1 \Rightarrow z < 0$$

$$z < 0 \Rightarrow w_0 + w_1 x_1 + w_2 x_2 < 0$$

$$\Rightarrow w_0 < 0 \quad (1) \quad w_0 = -3$$

Next $x_1 = 1, x_2 = 0 \Rightarrow \hat{y} > 0.5 \Rightarrow z > 0$

$$\Rightarrow w_0 + w_1 \cdot 1 + w_2 \cdot 0 > 0$$

$$\Rightarrow w_0 + w_1 > 0 \quad (2) \quad w_1 = 4$$

So, that is really what we want to achieve through our process. So, z less than 0, now z is less than 0 means $w_0 + w_1 x_1 + w_2 x_2$ is less than 0. Now we already know this data point this was x_1 was 0 and x_2 was 0. So, this tells us w_0 is less than 0 this also tells us the importance of having a what

is known as a bias unit. So, without this constant term I cannot make this model one. So, this constant term is called the bias unit in neural networks.


Coming back here I have seen that w_0 is less than 0. Now what about the next data point? We had the case that when $x_1 = 1$ and $x_2 = 0$ this gave us \hat{y} was greater than 0.5 because it is supposed to be classified as 1. So, here probability that it is one better than 0.5. So, this gives us $w_0 + w_1 * 1 + w_2 * 0$ is greater than so this will imply z is positive. If y is greater than 0.5 z has to be greater than 0 so this should be greater than 0.

It gives us $w_0 + w_1$ is greater than 0. So, we have these two equations or inequalities. So, let us make some choice, let us say w_0 is -3 and w_1 we can just choose it to be 1 or let us say 2 I will just choose it could be well sorry I can choose it to be 4. So, that $w_0 + w_1$ is greater than z so these two conditions are then satisfied.

(Refer Slide Time: 12:19)

Next $x_1 = 0, x_2 = 1 \Rightarrow \hat{y} > 0.5 \Rightarrow z > 0$
 $\Rightarrow w_0 + w_1 * 0 + w_2 * 1 > 0$
 $\Rightarrow w_0 + w_2 > 0 \Rightarrow w_2 = 4$

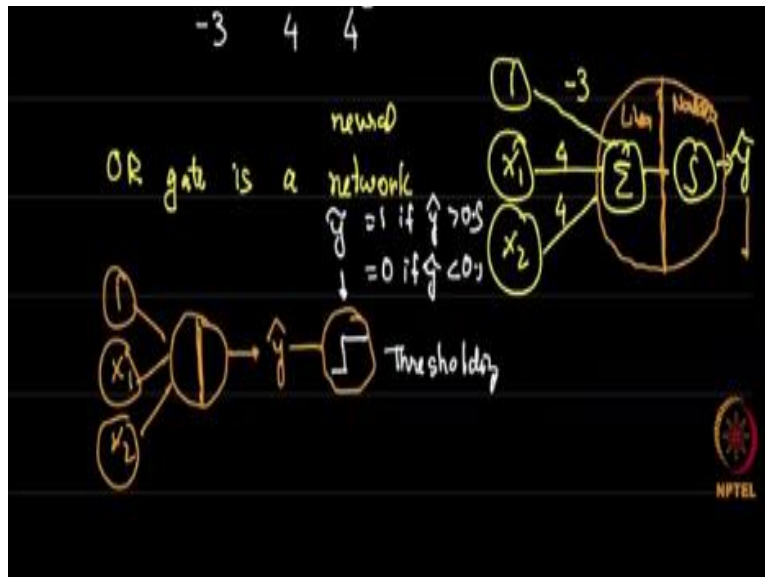
Finally $x_1 = 1, x_2 = 0 \Rightarrow \hat{y} > 0.5 \Rightarrow z > 0$
 $w_0 + w_1 + w_2 > 0$
 $-3 \quad 4 \quad 4$



Now next data point is $x_1 = 0, x_2 = 1$ and this gives us \hat{y} is again greater than 0.5, because this also was output equal to one case. So, again our probability has to be greater than 0.5. This in turn gives us that is greater than zero again. So, $w_0 + w_1 * 0 + w_2 * 1$ is greater than 0. So, this tells us $w_0 + w_2$ greater than 0. we can say let us say $w_2 = 4$. So, finally we have just four data points which we need to satisfy.

Obviously in reality you will have thousands but here it was $x_1 = 1$, $x_2 = 1$, \hat{y} had to be greater than 0.5, again z has to be greater than 0. So, you get $w_0 + w_1 + w_2$ greater than 0 which is true because w_0 we decided was -3 this is 4.

(Refer Slide Time: 13:39)



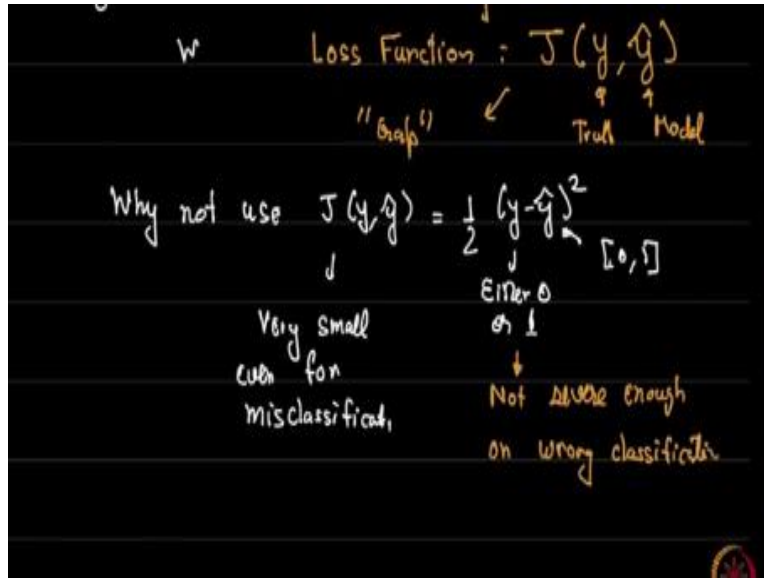
So, we actually achieve our end by choosing these four these three weights. So, we can say that OR gate is a network or a neural network I will call it a neural network with weights like this 1, x_1 , x_2 , -3, 4, 4, sigma and a sigmoid followed by \hat{y} . Typically, as I had said before we will put both these together into one big circle neuron with two functions a linear function and a non-linear function setting anything.

So, if you do this of course at the end of \hat{y} let me draw this clear field 1, x_1 , x_2 and let us say I am encompassing that in a single neuron outside of this is my y -hat but after that I need to undergo a step function. A step function which is $\hat{y} = 1$ if or let us call this instead of \hat{y} let us call this something else, $\tilde{y} = 1$ if \hat{y} greater than 0.5 and $\tilde{y} = 0$ if \hat{y} less than 0.5. Because at the end we still have to work with the gate and say whether it is one or zero.

In that case we go till here and then we will make a sharp threshold. so this is called a threshold. So, what was the point of all this? The point of all this was we can satisfy our conditions of this sort simply by inspection. Just by looking at it or by intuition or by doing some calculations we

were able to find out some weights w_0, w_1, w_2 . But for more complex cases obviously we cannot on.

(Refer Slide Time: 15:50)



In these cases, we will rely, in general we require a gradient descent. So, when we take gradient descent which way do we work? We work somewhat similar to the process that we had this was a rough trial and error process. but gradient descent as you all know is like this. You make a guess for a w , you put this value here put 0, 0 you will get \hat{y} equal to sigmoid of $w_0 + w_1x_1 + w_2x_2$. So, w means we of course give w_0, w_1, w_2 we give a guess that feeds in here.

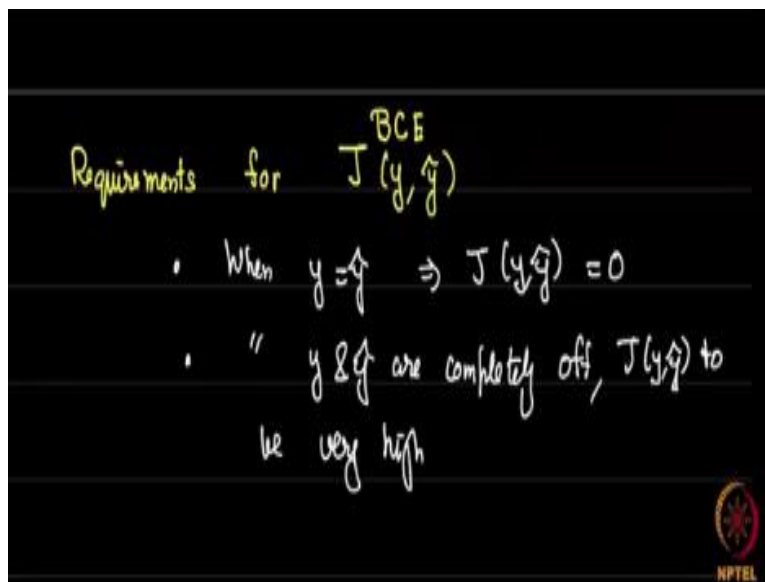
It makes a prediction for \hat{y} but that \hat{y} will not match Y . So, we require a loss function. So, in order for gradient descent to function we require two things. We require loss function which will take in J , J will take in y , our ground truth and \hat{y} my prediction from the model and basically it will give me a gap which I can sum. So, why not use our least square function $J(y, \hat{y})$ equal to half $y - \hat{y}$ square.

So, there are several problems with this I am going to mention only one. This is like I am going to wave my hands and give you a rough reason. Why this kind of thing does not work. The reason it does not work is noticing y is either 0 or 1 and \hat{y} always lies in the range 0 and 1. So, J is always going to be very small even for misclassification. For example, let us say a person does not have

cancer and I am guessing a model I have guessed the W and that model tells me that person has cancer.

Now these are serious consequences, you cannot just have a small correction for the fact that somebody who entirely had no cancer is being predicted as a person who is having cancer. Like I said this is a rough reason there are more technical reasons why this is true but I am going to skip that for now. Because of that this is not severe enough on wrong classification. So, we need a model which is severe on wrong classification.

(Refer Slide Time: 18:54)



So, the requirements for a general we are going to call it binary classification. I am going to finally call it BCE for another reason. But for binary classification we have some requirements. One of course is when $y = \hat{y}$ that is when you classify correctly this means $J(y, \hat{y})$ should of course be 0. But when y and \hat{y} are completely off, by completely off I mean \hat{y} says approximately 1 when y is exactly 0 then we want $J(y, \hat{y})$ has to be very high.

Now, notice when you do linear regression this problem is not there. If you actually your scope of y and \hat{y} is infinite a $w_0 + w_1 x_1$ without sigmoid for \hat{y} can grow as large as you want. Whereas this is not the case for logistic regression it is squeezed. The function is squeezed to be between 0 and 1 which is why this problem exists.

(Refer Slide Time: 20:14)

Binary Cross-Entropy Loss function

$$J(y, \hat{y}) = - \left[\overbrace{y \ln \hat{y}}^{J_1} + \overbrace{(1-y) \ln (1-\hat{y})}^{J_2} \right]$$

Possibilities

1. $y = 0 ; \hat{y} \approx 0 \quad J_1 = 0 + 1 \cdot \ln(1-\epsilon)$
 $\Rightarrow J \approx 0 \quad \approx \ln 1 \approx 0$

2. $y = 1 ; \hat{y} = 1 \quad J_1 = 1 \cdot \ln(1+\epsilon) + 0$
 $\Rightarrow J \approx 0 \quad \approx 0$

So, there are a few other requirements also, but I am not going to write that explicitly. I will give you a cost function which does at least the about two. This is called the binary cross entropy loss function. This is what is used for logistic impression and it functions in the following way. So, you say that,

$$J(y, \hat{y}) = -[y \ln \hat{y} + (1-y)\ln(1-\hat{y})]$$

So, let us look at this expression in slightly more detail.

So, what we do is think about what each of these terms can be. So, here are the possibilities let me write down the possibilities. Possibility one is that the ground truth y is 0 and \hat{y} is approximately zero. Remember \hat{y} is sigmoid of z it can never exactly be zero it can only be approximately 0 because you have $\frac{1}{1+e^{-z}}$. So, let us look at what happens in this case. What will happen in this case is that y is 0, so the first term is 0.

The second term so let us call this J_1 , let us call this J_2 . So, J_1 is 0 because, y is 0 and \ln of something approximately 0. So, we do not care about what it is, it is multiplying zero. let us look at J_2 , J_2 is 1 multiplying \ln of 1 minus a term which is epsilon it is approximately 0. So, this is approximately \ln of 1 so which is approximately 0. So, this tells us that J is approximately 0 if both y is 0 as well as \hat{y} is 0.

Second y is 1 and \hat{y} is also one now let us look at this J_1 the first term is this term is 1 multiplied by \ln of a term which is approximately one. Let us say it is $1 + \epsilon + 1 - y$ notice y is 1 so this is zero nothing else. So, this is approximately \ln of 1 so this is again approximately zero so this also gives us approximately zero. So, we satisfy the condition that when y and \hat{y} are approximately the same or when we get the same class in both cases the cost is 0.

(Refer Slide Time: 23:25)

3. $y = 0; \hat{y} \approx 1 \Rightarrow J_1 = -y \ln \hat{y} = 0$ $\epsilon \ll 1$
 $J_2 = -(1-y) \ln(1-\hat{y}) = -1 \ln(1-1+\epsilon) = -\ln \epsilon$
 $\epsilon \rightarrow 0 \Rightarrow J_2 \rightarrow \infty$
 $\Rightarrow J \rightarrow \infty$

4. $y = 1; \hat{y} \approx 0 \Rightarrow J \rightarrow \infty$
 Misclassification gets penalized heavily

So, I am not going to penalize a case where which something has been classified correctly. But let us look at the second case $y = 0$ but $\hat{y} = 1$. In this case J_1 the first term here is $-y \ln \hat{y}$. So, $-y \ln \hat{y}$ is 0 because y is 0. What about J_2 ? $J_2 = -(1-y) \ln(1-\hat{y})$. Now let us look at this now this is $-1 \ln(1-\hat{y})$ is approximately one. So, let us say it is $1 - \epsilon$ it will be a little bit less than epsilon.

Actually, here two I should have changed it to $1 - \epsilon$ because \hat{y} always changes between 0 and 1, it can never be $1 + \epsilon$. Coming back here what I am saying is that this term is 1 exactly and this term is approximately zero. This is equal to $-\ln \epsilon$ where ϵ is very small. Why is that? Because \hat{y} is approximately one it is almost close to one. Let us say it is 0.99 then you will get \ln of 0.001 which means this term as you can see epsilon tends to 0 means J_2 tends to infinity.

What this tells us is the loss tends to infinity for the binary cross entropy, in case we misclassified. If similarly, I will just leave this as an exercise in order to not board you. So, similarly if $y = 1$ and \hat{y} equal approximately equal to 0, J will again be quite large. So, misclassification x penalized

heavily so we basically put together these two terms. Now where did these two terms magically appear from?

Unfortunately, I do not have the time you can show a derivation from probability theory for this two just like we did for the least square term. We can show it with this with some binomial distribution or polynomial distribution but I will not get into that. So, currently suppose assuming this term comes, it satisfies these very two important properties that when y predicts correctly y is predicted correctly your error is zero.

But when y 's prediction is completely off that is it misclassifies, then the cost function becomes very high.

(Refer Slide Time: 26:29)

To update w , we need $\frac{\partial J}{\partial w}$

$$\frac{\partial J}{\partial w_1} = \frac{\partial J}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w_1}$$

$$\hat{y} = \sigma(z)$$

$$z = w_0 + w_1 x_1 + w_2 x_2$$

$$J = -[y \ln \hat{y} + (1-y) \ln (1-\hat{y})]$$

$$\frac{\partial J}{\partial \hat{y}} = - \left[\frac{y}{\hat{y}} + \frac{(1-y)}{(1-\hat{y})} \right] \quad (1)$$

$$\frac{\partial \hat{y}}{\partial w_1} = \frac{\partial \hat{y}}{\partial z} \cdot \frac{\partial z}{\partial w_1}$$

So, what we still need the gradient. Once we have J to update W , we need $\frac{\partial J}{\partial w}$. So, how do we find out $\frac{\partial J}{\partial w}$ with logistic regression. So, let us look at that shortly now. Just so, that we have a complete idea of how to establish gradient descent in this case. So, now we have $\frac{\partial J}{\partial w}$ let us say I want to do $\frac{\partial J}{\partial w_1}$ this will be the same as $\frac{\partial J}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w_1}$. What are the relationships we know?

We know that $\hat{y} = \sigma(z)$. What is Z ? $z = w_0 + w_1 x_1 + w_2 x_2$. What is J ?

$$J = -[y \ln \hat{y} + (1 - y) \ln (1 - \hat{y})]$$

So, let us differentiate these terms one by one. First thing we want $\frac{\partial J}{\partial \hat{y}}$ this term. So,

$$\frac{\partial J}{\partial \hat{y}} = - \left[\frac{y}{\hat{y}} + \frac{(1-y)}{(1-\hat{y})} \right]$$

This is the first term. So, let us note this down. Next, we want $\frac{\partial \hat{y}}{\partial w_1}$ but \hat{y} does not directly depend on w_1 .

So, you see a chain, first w_1 affects x and z affects \hat{y} . I will show this process very clearly in the next week's video but for now just remember this. So, instead of calculating this directly we say

$$\frac{\partial \hat{y}}{\partial w_1} = \frac{\partial \hat{y}}{\partial z} \frac{\partial z}{\partial w_1}$$

(Refer Slide Time: 29:03)

Handwritten derivation on a blackboard:

$\frac{\partial \hat{y}}{\partial z}$ where $\hat{y} = \frac{1}{1+e^{-z}}$ ✓ $1-\hat{y} = \frac{1}{1+e^z} = \frac{e^{-z}}{1+e^{-z}}$
 $\frac{d\hat{y}}{dz} = \frac{-1 * -e^{-z}}{(1+e^{-z})^2} = \frac{e^{-z}}{(1+e^{-z})^2}$
 $\frac{d\hat{y}}{dz} = \frac{e^{-z}}{(1+e^{-z})} \cdot \frac{1}{1+e^{-z}}$
 (Note: $\frac{e^{-z}}{(1+e^{-z})} = \frac{1}{1+e^z} = 1-\hat{y}$ and $\frac{1}{1+e^{-z}} = \hat{y}$)

Now $\frac{\partial z}{\partial w_1}$ is straight forward so $\frac{\partial z}{\partial w_1}$ is simply x_1 . So, this term is x_1 I will leave this as an equation 2, we have this term now. What is $\frac{\partial \hat{y}}{\partial z}$? This we do not know as you can see. We have to step by step unravel what is happening. So, $\frac{\partial \hat{y}}{\partial z}$ where \hat{y} is,

$$\hat{y} = \frac{1}{1 + e^{-z}}$$

So, now I differentiate this so $d\hat{y}$ it is just $\frac{\partial \hat{y}}{\partial z}$ in this case this is - 1 by the denominator square multiplied by the derivative of this with respect to z .

Which is $\frac{e^{-z}}{(1+e^{-z})^2}$. Now when you see this you can split it further. This is just a centric $\frac{e^{-z}}{(1+e^{-z})} \frac{1}{(1+e^{-z})}$

So, this is $\frac{\partial \hat{y}}{\partial z}$. This second term here is simply the original y hat that we have you can notice this.

Now the second term the first term here if you calculate,

$$\begin{aligned}(1 - \hat{y}) &= 1 - \frac{1}{1 + e^{-z}} \\ &= \frac{e^{-z}}{1 + e^{-z}}\end{aligned}$$

(Refer Slide Time: 30:55)

$$\begin{aligned}\frac{d\hat{y}}{dz} &= \hat{y}(1-\hat{y}) \quad \text{if } \hat{y} = \sigma(z) \\ \sigma'(z) &= \sigma(z)(1-\sigma(z))\end{aligned}$$

So, this is a convenient expression which is $\frac{\partial \hat{y}}{\partial z} = \hat{y}(1 - \hat{y})$ if $\hat{y} = \sigma(z)$. You might also see this at other places as,

$$\sigma'(z) = \sigma(z)(1-\sigma(z))$$

which has the same meaning as well right so far. So, let us call this equation 3. So, if we put together 1, 2 and 3 we get the following relationship. We get,

$$\frac{\partial J}{\partial w_1} = \left[\frac{\partial z}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z} \right] \frac{\partial z}{\partial w_1}$$

(Refer Slide Time: 32:06)

$$\begin{aligned}
 \frac{\partial J}{\partial w_1} &= \left[\frac{\partial J}{\partial \hat{y}} \quad \frac{\partial \hat{y}}{\partial z} \right] \frac{\partial z}{\partial w_1} \\
 &= - \left[\frac{y}{\hat{y}} + \frac{(1-y)}{(1-\hat{y})} \right] \hat{y}(1-\hat{y}) x_1 \\
 &= - [y(1-\hat{y}) + (1-y)\hat{y}] x_1 \\
 &= - [y - y\hat{y} - \hat{y} + y\hat{y}]
 \end{aligned}$$

Let us look at this term $\frac{\partial J}{\partial \hat{y}}$ was $-\left[\frac{y}{\hat{y}} + \frac{(1-y)}{(1-\hat{y})}\right] \hat{y}(1-\hat{y})x_1$. So, this can be simplified a little bit further. So, let us multiply this, this is, $-[y(1-\hat{y}) + (1-y)\hat{y}]x_1$, $-[y - y\hat{y} - \hat{y} + y\hat{y}]x_1$ sorry there should be a minus here.

(Refer Slide Time: 32:53)

$$\begin{aligned}
 &= - [y(1-\hat{y}) + (1-y)\hat{y}] x_1 \\
 &= - [y - y\hat{y} - \hat{y} + y\hat{y}] \\
 \Rightarrow \left. \frac{\partial J}{\partial w_1} \right\} &= \underbrace{[\hat{y} - y]}_{\text{Err}} x_1 \quad \text{- Same expression as linear regression's gradient!}
 \end{aligned}$$

Algorithm :

- ① Guess for w
- ② Dataset, \hat{y} using w
- ③ $w = w + \alpha \frac{\partial J}{\partial w} \rightarrow \frac{\partial J}{\partial w_j} = 2\hat{y} - y$

So, if you put these together you get $(\hat{y} - y)x_1$. So, this gives us,

$$\frac{\partial J}{\partial w_1} = (\hat{y} - y)x_1$$

This happens to be the same expression as linear regression. So, linear regression gradient if you go and look back has exactly the same expression. I will explain why actually at the beginning of next week. Why is it that we are hitting the same expression here as well as the other place.

So, as it turns out you can write one single program to do the gradient of both linear regression as well as logistic regression. So, it is a very simple idea as I had told you earlier weeks during gradient descent sorry during linear regression is the gradient with respect to w_1 is simply the error multiplied by the activation. So, what do we have here our algorithm been very simple in order to use logistic regression, guess for W is the first guess then for data set let us say we are doing a batch gradient descent then just calculate y at using W then find gradient.

$w = w - \alpha \frac{\partial J}{\partial w}$, where gradient $\frac{\partial J}{\partial w}$ any of the W 's J is simply $\sum(\hat{y} - y)x_j$. So, this is very similar in fact it is identical to what we did with linear regression. So, where was the linear regression different? The difference was finding this \hat{y} , \hat{y} does not have a sigmoid in linear regression whereas it has a sigmoid as far as logistic regression is concerned.

So, what we did in this video was looked at a couple of things we defined this binary cross entropy loss function that is why I called it BCE. So, binary cross entropy loss function you find out the binary cross entropy loss function and then you take this gradient. So, the gradient looked a little bit complex in fact what we will do next week is essentially automate this process of taking the gradient. I have made a mistake here too you can correct your notes.

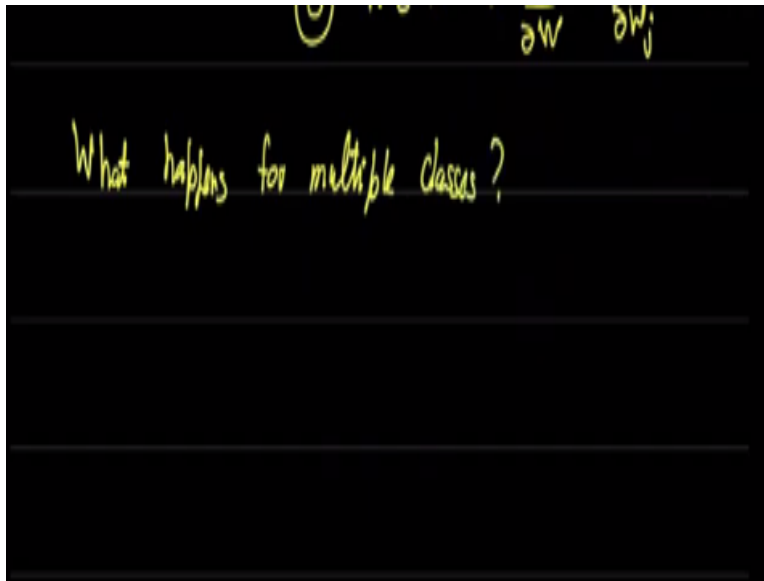
There should be a minus here because when I differentiate $1 - y$ there should be a minus coming up front. So, that is what reflects here in the final negative term here. So, you put that together you get a simple expression for the gradient and you can just update it. So, this data set or any other data set indeed which we will show in several demos. The next week I am not keeping any demos this week is primarily just a theoretical class.

So, if we look at a data set like this all you would do is make prediction make prediction make prediction make prediction add the four divides by 4, so that is your gradient. Now calculate the gradient as sigma of y hat - y multiplied by x . Just keep on repeating and you will find a way which

will magically land up here. So, it is more or less like magic when you actually see it in demonstration. So, I will show you this OR gate demonstration in the next screen.

Now in the next video I want to address a few more things. So, we looked at the case where we have only two classes.

(Refer Slide Time: 36:50)



But what happens when we have multiple classes? So, this case we will see in the next video. So, I will see you in the next video, thank you.