Logistic Regression

(SPEECH)
In

(DESCRIPTION)
Text, Basics of Neural Network Programming. Logistic Regression. Website, deep learning, dot, A.I.

(SPEECH)
this video, we'll go over logistic regression.

This is a learning algorithm that you use when the output labels Y in a supervised learning problem are all either zero or one, so for binary classification

(DESCRIPTION)
New slide, Logistic Regression.

(SPEECH)
problems.

Given an input feature vector X maybe corresponding to an image that you want to recognize as either a cat picture or not a cat picture, you want an algorithm that can output a prediction, which we'll call Y hat, which is your estimate of Y.

More formally, you want Y hat to be the probability of the chance that, Y is equal to one given the input features X.

So in other words, if X is a picture, as we saw in the last video, you want Y hat to tell you, what is the chance that this is a cat picture?

So X, as we said in the previous video, is an X dimensional vector, given that the parameters of logistic regression will be W which is also an X dimensional vector, together with b which is just a real number.

So given an input X and the parameters W and b, how do we generate the output Y hat?

Well, one thing you could try, that doesn't work, would be to have Y hat be w transpose X plus B, kind of a linear function of the input X.

And in fact, this is what you use if you were doing linear regression.

But this isn't a very good algorithm for binary classification because you want Y hat to be the chance that Y is equal to one.

So Y hat should really be between zero and one, and it's difficult to enforce that because W transpose X plus B can be much bigger than one or it can even be negative, which doesn't make sense for probability.

That you want it to be between zero and one.

So in logistic regression, our output is instead going to be Y hat equals the sigmoid function applied to this quantity.

This is what the sigmoid function looks like.

If on the horizontal axis I plot Z, then the function sigmoid of Z looks like this.

So it goes smoothly from zero up to one.

Let me label my axes here, this is zero and it crosses the vertical axis as 0.5.

So this is what sigmoid of Z looks like. And we're going to use Z to denote this quantity, W transpose X plus B.

Here's the formula for the sigmoid function.

Sigmoid of Z, where Z is a real number, is one over one plus E to the negative Z.

So notice a couple of things.

If Z is very large, then E to the negative Z will be close to zero.

So then sigmoid of Z will be approximately one over one plus something very close to zero, because E to the negative of very large number will be close to zero.

So this is close to 1.

And indeed, if you look in the plot on the left, if Z is very large the sigmoid of Z is very close to one.

Conversely, if Z is very small, or it is a very large negative number, then sigmoid of Z becomes one over one plus E to the negative Z, and this becomes, it's a huge number.

So this becomes, think of it as one over one plus a number that is very, very big, and so, that's close to zero.

And indeed, you see that as Z becomes a very large negative number, sigmoid of Z goes very close to zero.

So when you implement logistic regression, your job is to try to learn parameters W and B so that Y hat becomes a good estimate of the chance of Y being equal to one.

Before moving on, just another note on the notation.

When we programmed neural networks, we'll usually keep the parameter W and parameter B separate, where here, B corresponds to an inter-spectrum.

In some other courses, you might have seen a notation that handles this differently.

In some conventions you define an extra feature called X0 and that equals a one.

So that now X is in R of NX plus one.

And then you define Y hat to be equal to sigma of theta transpose X.

In this alternative notational convention, you have vector parameters theta, theta zero, theta one, theta two, down to theta NX And so, theta zero, place a row a B, that's just a real number, and theta one down to theta NX play the role of W. It turns out, when you implement your neural network, it will be easier to just keep B and W as separate parameters.

And so, in this class, we will not use any of this notational convention that I just wrote in red.

If you've not seen this notation before in other courses, don't worry about it.

It's just that for those of you that have seen this notation I wanted to mention explicitly that we're not using that notation in this course.

But if you've not seen this before, it's not important and you don't need to worry about it.

So you have now seen what the logistic regression model looks like.

Next to change the parameters W and B you need to define a cost function.

Let's do that in the next video.

Logistic Regression (SPEECH) In (DESCRIPTION) Text, Basics of Neural Network Programming. Logistic Regression. Website, deep learning, dot, A.I. (SPEECH) this video, we'll go over logistic regression. This is a learning algorithm that you use when the output labels Y in a supervised learning problem are all either zero or one, so for binary classification (DESCRIPTION) New slide, Logistic Regression. (SPEECH) problems. Given an input feature vector X maybe corresponding to an image that you want to recognize as either a cat picture or not a cat picture, you want an algorithm that can output a prediction, which we'll call Y hat, which is your estimate of Y. More formally, you want Y hat to be the probability of the chance that, Y is equal to one given the input features X. So in other words, if X is a picture, as we saw in the last video, you want Y hat to tell you, what is the chance that this is a cat picture? So X, as we said in the previous video, is an X dimensional vector, given that the parameters of logistic regression will be W which is also an X dimensional vector, together with b which is just a real number. So given an input X and the parameters W and b, how do we generate the output Y hat? Well, one thing you could try, that doesn't work, would be to have Y hat be w

transpose X plus B, kind of a linear function of the input X. And in fact, this is what you use if you were doing linear regression. But this isn't a very good algorithm for binary classification because you want Y hat to be the chance that Y is equal to one. So Y hat should really be between zero and one, and it's difficult to enforce that because W transpose X plus B can be much bigger than one or it can even be negative, which doesn't make sense for probability. That you want it to be between zero and one. So in logistic regression, our output is instead going to be Y hat equals the sigmoid function applied to this quantity. This is what the sigmoid function looks like. If on the horizontal axis I plot Z, then the function sigmoid of Z looks like this. So it goes smoothly from zero up to one. Let me label my axes here, this is zero and it crosses the vertical axis as 0.5. So this is what sigmoid of Z looks like. And we're going to use Z to denote this quantity, W transpose X plus B. Here's the formula for the sigmoid function. Sigmoid of Z, where Z is a real number, is one over one plus E to the negative Z. So notice a couple of things. If Z is very large, then E to the negative Z will be close to zero. So then sigmoid of Z will be approximately one over one plus something very close to zero, because E to the negative of very large number will be close to zero. So this is close to 1. And indeed, if you look in the plot on the left, if Z is very large the sigmoid of Z is very close to one. Conversely, if Z is very small, or it is a very large negative number, then sigmoid of Z becomes one over one plus E to the negative Z, and this becomes, it's a huge number. So this becomes, think of it as one over one plus a number that is very, very big, and so, that's close to zero. And indeed, you see that as Z becomes a very large negative number, sigmoid of Z goes very close to zero. So when you implement logistic regression, your job is to try to learn parameters W and B so that Y hat becomes a good estimate of the chance of Y being equal to one. Before moving on, just another note on the notation. When we programmed neural networks, we'll usually keep the parameter W and parameter B separate, where here, B corresponds to an inter-spectrum. In some other courses, you might have seen a notation that handles this differently. In some conventions you define an extra feature called X0 and that equals a one. So that now X is in R of NX plus one. And then you define Y hat to be equal to sigma of theta transpose X. In this alternative notational convention, you have vector parameters theta, theta zero, theta one, theta two, down to theta NX And so, theta zero, place a row a B, that's just a real number, and theta one down to theta NX play the role of W. It turns out, when you implement your neural network, it will be easier to just keep B and W as separate parameters. And so, in this class, we will not use any of this notational convention that I just wrote in red. If you've not seen this notation before in other courses, don't worry about it. It's just that for those of you that have seen this notation I wanted to mention explicitly that we're not using that notation in this course. But if you've not seen this before, it's not important and you don't need to worry about it. So you have now seen what the logistic regression model looks like. Next to change the parameters W and B you need to define a cost function. Let's do that in the next video.