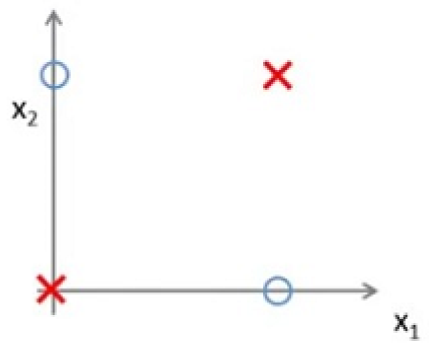
Examples and Intuitions I

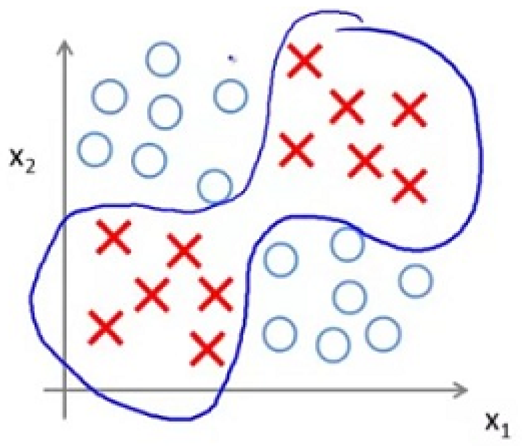
Non-linear classification example: XOR/XNOR ¶

We consider the following problem where we have two input features, x_1 and x_2 that are binary values.

x_1, x_2 are binary (0 or 1).

We can think the following example as a simplified version of a more complex learning problem where we may have a bunch of positive examples in the upper right and lower left and a bunch of negative examples denoted by the circles. And what we'd like to do is learn a **non-linear division of boundary** that may need to separate the positive and negative examples, So how can a neural network do this?





Concretely what a neural network is doing, is really computing the type of label y where:

$y = x_1 \text{ XOR } x_2$

$x_1 \text{ XNOR } x_2$

$\text{NOT } (x_1 \text{ XOR } x_2)$

$\text{NOT } (x_1 \text{ XOR } x_2)$ is the alternative notation for $x_1 \text{ XNOR } x_2$.

So, $x_1 \text{ XOR } x_2$ that's true only if exactly 1 of x_1 or x_2 is equal to 1.

Simple example: AND

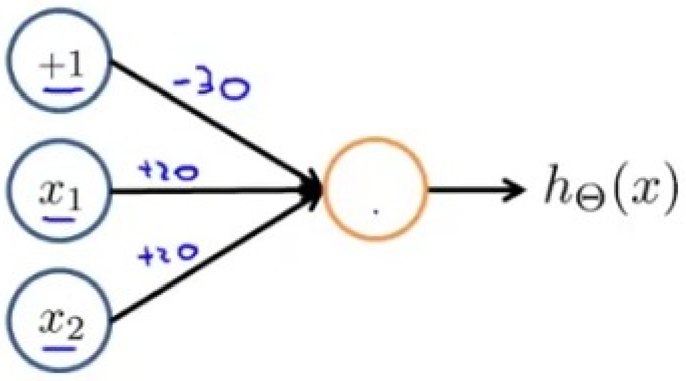
Concretely, let's say we have input x_1 and x_2 that are again binaries so, it's either 0 or 1 and let's say our target labels $y = x_1 \text{ AND } x_2$, is a logical AND.

$x_1, x_2 \in \{0, 1\}$

$y = x_1 \text{ AND } x_2$

So, can we get a one-unit network to compute this logical AND function?

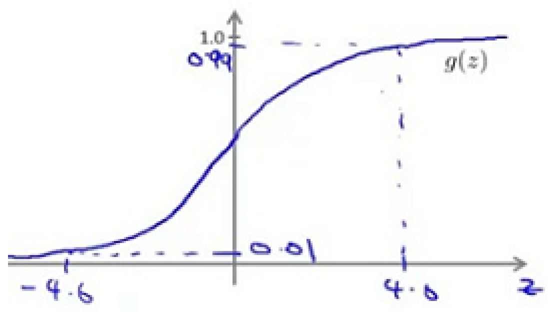
In order to do so, we're going to add the bias unit as well (the plus one unit), and assign some values to the weights or parameters of the network.

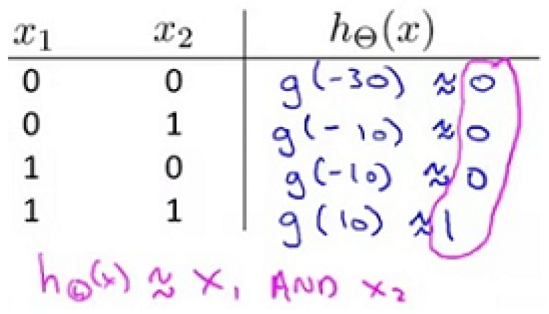


Concretely, this is saying that our hypothesis is: $h_\Theta(x) = g(-30 + 20x_1 + 20x_2)$

Sometimes it's convenient to add the weights into the diagram. These values are in fact just the Θ parameters so:

- $\Theta^{(1)}_{10} = -30$
- $\Theta^{(1)}_{11} = 20$
- $\Theta^{(1)}_{12} = 20$



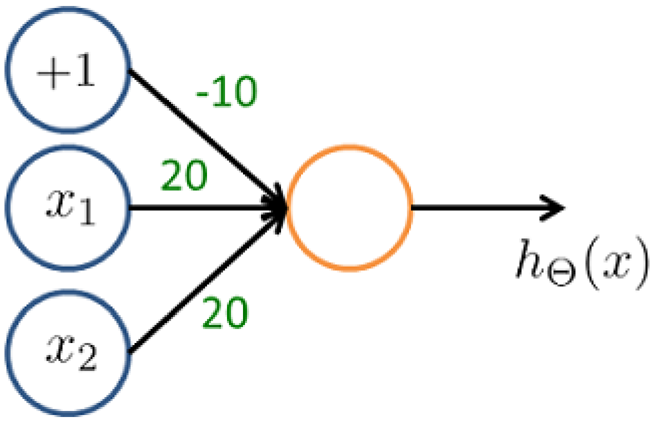


x_1	x_2	$h_\Theta(x)$
0	0	$g(-30) \approx 0$
0	1	$g(-10) \approx 0$
1	0	$g(-10) \approx 0$
1	1	$g(10) \approx 1$

$h_\Theta(x) \approx x_1$ AND x_2

So, as we can see, when we evaluate each of the four possible input, only $(1, 1)$ gives a positive output

Video Question: Suppose x_1 and x_2 are binary valued (0 or 1). What boolean function does the network shown below (approximately) compute? (Hint: One possible way to answer this is to draw out a truth table, similar to what we did in the video).



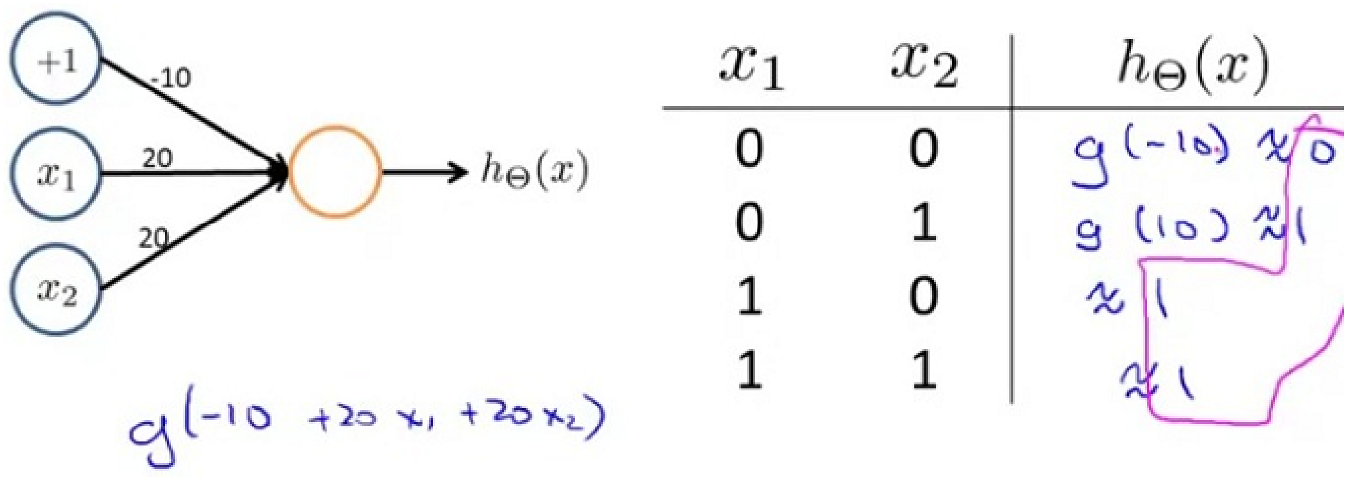
- x_1 AND x_2

- (NOT x_1) OR (NOT x_2)

x_1 OR x_2

- (NOT x_1) AND (NOT x_2)

Example: OR function



x_1	x_2	$h_\Theta(x)$
0	0	$g(-10) \approx 0$
0	1	$g(10) \approx 1$
1	0	≈ 1
1	1	≈ 1

Summary

A simple example of applying neural networks is by predicting x_1 AND x_2, which is the logical 'and' operator and is only true if both x_1 and x_2 are 1.

The graph of our functions will look like:

$$\begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix} \rightarrow \begin{bmatrix} g(z^{(2)}) \end{bmatrix} \rightarrow h_\Theta(x)$$

Remember that x_0 is our bias variable and is always 1.

Let's set our first theta matrix as:

$\Theta^{(1)} = \begin{bmatrix} -30 & 20 & 20 \end{bmatrix}$ This will cause the output of our hypothesis to only be positive if both x_1 and x_2 are 1. In other words:

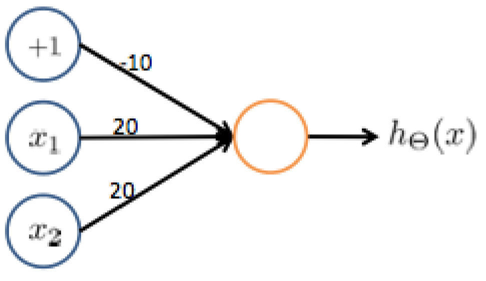
$$h_\Theta(x) = g(-30 + 20x_1 + 20x_2)$$

$$\begin{aligned} & x_1 = 0 \ \ and \ \ x_2 = 0 \ \ then \ \ g(-30) \approx 0 \\ & x_1 = 0 \ \ and \ \ x_2 = 1 \ \ then \ \ g(-10) \approx 0 \\ & x_1 = 1 \ \ and \ \ x_2 = 0 \ \ then \ \ g(-10) \approx 0 \\ & x_1 = 1 \ \ and \ \ x_2 = 1 \ \ then \ \ g(10) \approx 1 \end{aligned}$$

So we have constructed one of the fundamental operations in computers by using a small neural network rather than using an actual AND gate. Neural networks can also be used to simulate all the other logical gates.

The following is an example of the logical operator 'OR', meaning either x_1 is true or x_2 is true, or both:

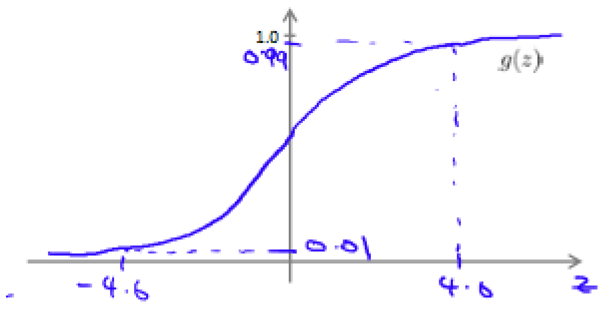
Example: OR function



$g(-10 + 20x_1 + 20x_2)$

x_1	x_2	$h_\Theta(x)$
0	0	$g(-10) \approx 0$
0	1	$g(10) \approx 1$
1	0	≈ 1
1	1	≈ 1

Where $g(z)$ is the following:



1.0
0.99
$g(z)$
0.01
-4.6
4.6
z