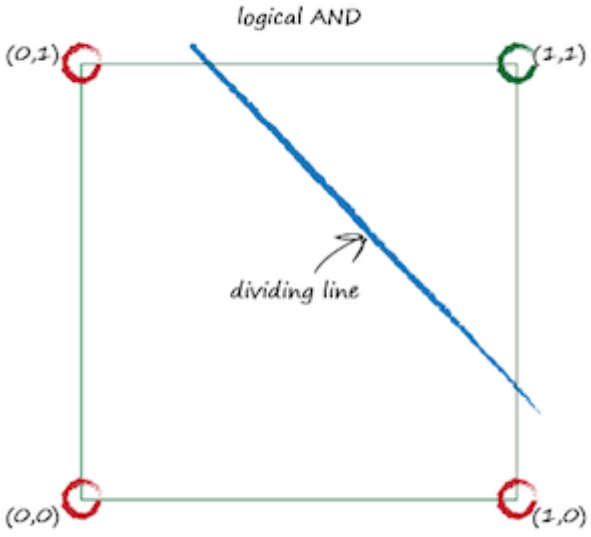
Representing Boolean Functions with Linear Classification

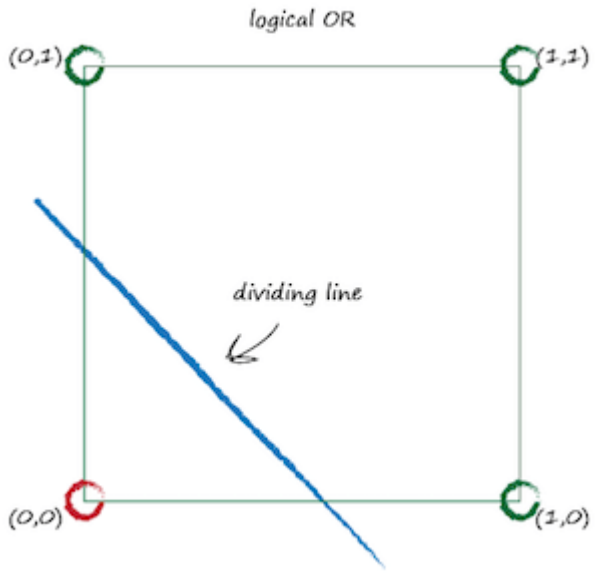
Is it possible to represent the data of any boolean function using a single linear classifier? Let's find out!

Imagine using a simple linear classifier to learn from training data whether the data was governed by a Boolean logic function. That's a natural and useful thing to do for scientists wanting to find causal links or correlations between some observations and others. For example, is there more malaria when it rains, AND it is hotter than 35 degrees? Is there more malaria when either (Boolean OR) of these conditions is true? Look at the following plot, showing the two inputs A and B to the logical function as coordinates on a graph. The plot shows that only when both are true, with value 1, is the output also true, shown as green. False outputs are shown red.



You can also see a straight line that divides the red from the green regions. That line is a linear function that a linear classifier could learn, just as we have done earlier. We won't go through the numerical workings out as we did before because they're not fundamentally different in this example. In fact, there are many variations on this dividing line that would work just as well, but the main point is that it is indeed possible for a simple linear classifier of the form $y = ax + b$ to learn the Boolean AND function. Now look at the

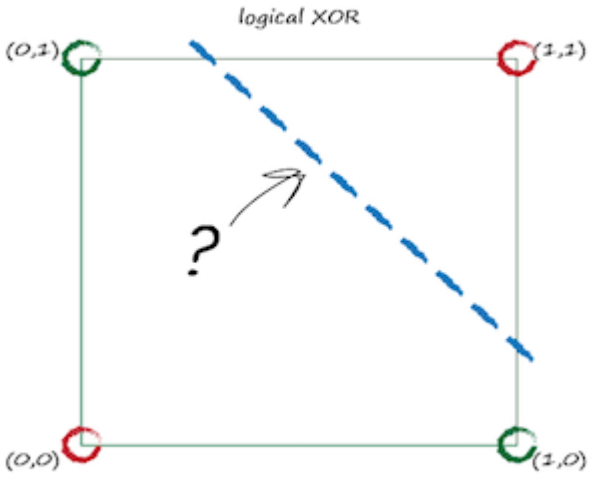
the form $y = ax + b$ to learn the Boolean AND function. Now look at the Boolean OR function plotted in a similar way:



This time only the $(0, 0)$ point is red because it corresponds to both inputs A and B are false. All other combinations have at least one A or B as true, and so the output is true. The beauty of the diagram is that it makes clear that it is possible for a linear classifier to learn the Boolean OR function, too. There is another Boolean function called XOR, short for exclusive OR, which only has a true output if either one of the inputs A or B is true, but not both. That is when the inputs are both false, or both true, the output is false. The following table summarises this:

Input A	Input B	Logical XOR
0	0	0
0	1	1
1	0	1
1	1	0

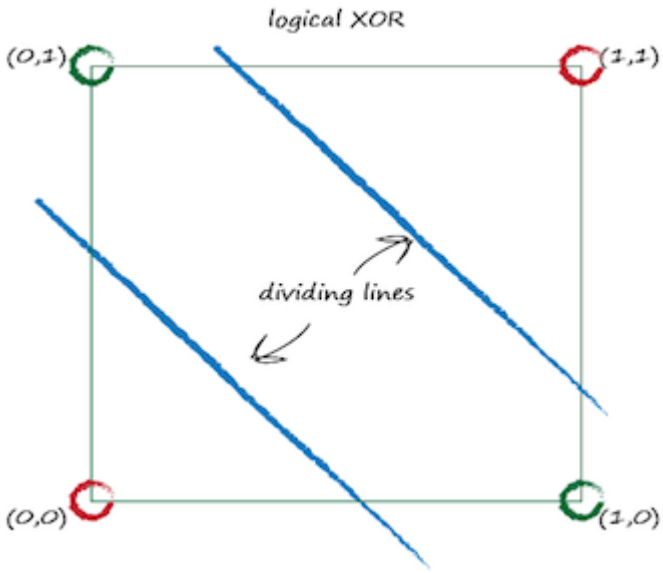
Now look at a plot of this function on a grid with the outputs colored:



This is a challenge! We can't seem to separate the red from the blue regions with only a single straight dividing line. It is, in fact, impossible to have a single straight line that successfully divides the red from the green regions for the Boolean XOR. That is, a simple linear classifier can't learn the Boolean XOR if presented with training data that was governed by the XOR function.

We've just illustrated a major limitation of the simple linear classifier. A simple linear classifier is not useful if the underlying problem is not separable by a straight line. We want neural networks to be useful for the many many tasks where the underlying problem is not linearly separable — where a single straight line doesn't help.

So we need a fix. Luckily the fix is easy. In fact, the diagram below which has two straight lines to separate out the different regions suggests the fix — we use multiple classifiers working together. That's an idea central to neural networks. You can imagine already that many linear lines can start to separate off even unusually shaped regions for classification.



We have now covered enough background knowledge and are ready to learn how Nueral Network works!