Limitations of Linear Classifiers

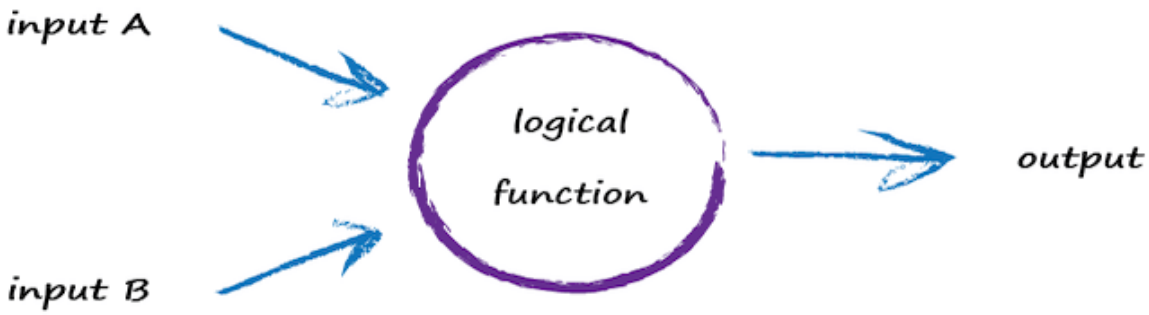
In this lesson, we will look at some of the limitations of linear classifiers.

The simple predictors and classifiers we have worked with so far — the ones that take some input, do some calculation and throw out an answer — although pretty effective as we have just seen, are not enough to solve some of the more interesting problems we hope to apply neural networks too. Here we will illustrate the limit of a linear classifier with a simple but stark example. Why do we want to do this, and not jump straight to discussing neural networks? The reason is that a key design feature of neural networks comes from understanding this limit — so worth spending a little time on.

We'll be moving away from garden bugs and looking at *Boolean* logic functions. If that sounds like mumbo-jumbo jargon — don't worry. George Boole was a mathematician and philosopher, and his name is associated with simple functions like **AND** and **OR**.

Boolean logic functions are like language or thought functions. If we say "you can have your pudding only if you've eaten your vegetables AND if you're still hungry" we're using the Boolean AND function. The Boolean AND is only true if both conditions are true. It's not true if only one of them is true. So if I'm hungry, but haven't eaten my vegetables, then I can't have my pudding.

Similarly, if we say "you can play in the park if it's the weekend OR you're on annual leave from work" we're using the Boolean OR function. The Boolean OR is true if any, or all, of the conditions, are true. They don't all have to be true like the Boolean AND function. So if it's not the weekend, but I have booked annual leave, I can indeed go and play in the park. If we think back to our first look at functions, we saw them as a machine that took some inputs, did some work, and output an answer. Boolean logical functions typically take two inputs and output one answer:



Computers often represent *true* as the number **1** and *false* as the number **0**. The following table shows the logical AND and OR functions using this more concise notation for all combinations of inputs A and B.

Input A	Input B	Logical AND	Logical OR
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

You can see quite clearly, that the AND function is only true if both A and B are true. Similarly, you can see that OR is true whenever any of the inputs A or B is true. Boolean logic functions are really important in computer science, and in fact, the earliest electronic computers were built from tiny electrical circuits that performed these logical functions. Even arithmetic was done using combinations of circuits which themselves were simple Boolean logic functions.