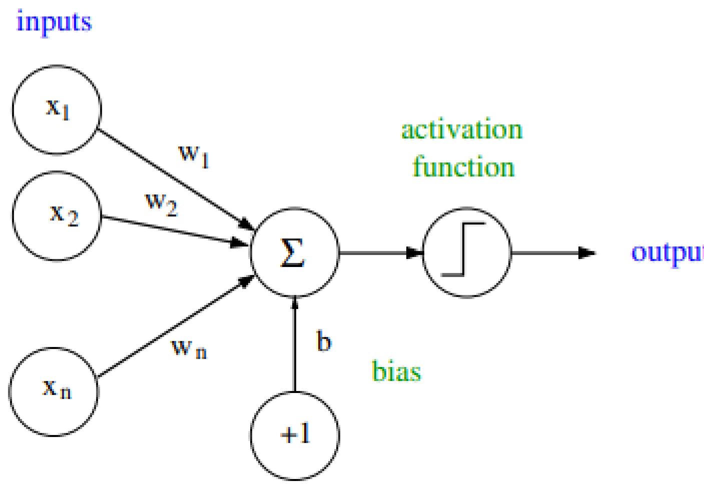
Expressiveness

This lesson is focused in what we can compute with a NN.



Suppose we have a single layer NN.

For the moment, let's suppose that we have a binary function as an activation

$$output = \begin{cases} 1 & if \sum_i w_i x_i + b \geq 0 \\ 0 & otherwise \end{cases} \qquad output = \begin{cases} 1 & if \sum_i w_i x_i \geq -b \\ 0 & otherwise \end{cases}$$

function:
The bias allow us to *fix* the *threshold* that we're *interested* in.

Hyperplane

The set of points:

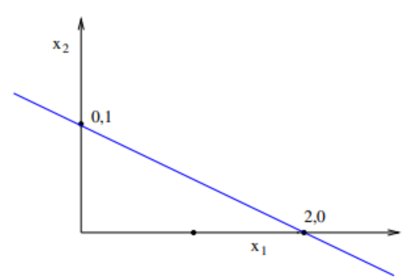
$$\sum_i w_i x_i + b = 0$$

defines a hyperplane in

Example:

$$-\frac{1}{2}x_1 + x_2 + 1 = 0$$

is a line in the bidimensional space



the space of the variables x_i.

The *hyperplane* divides the space in *two parts*: - to one of them (above the line) the perceptron gives value 1, - to the other (below the line) value 0.

NN logical connections

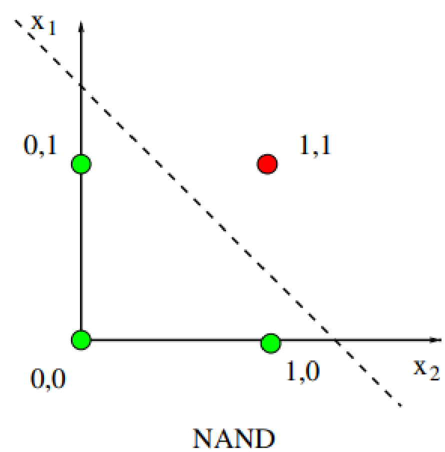
Can we implement this function (NAND) with a perceptron?

x_1	x_2	*output*
0	0	1
0	1	1
1	0	1
1	1	0

Can we find two weights w_1 and w_2 and a bias b such that

$$nand(x_1, x_2) = \begin{cases} 1 & if \sum_i w_i x_i + b' \geq 0 \\ 0 & otherwise \end{cases}$$

Yes!



line equation: $1.5 - x_1 - x_2 = 0$ or $3 - 2x_1 - 2x_2 = 0$

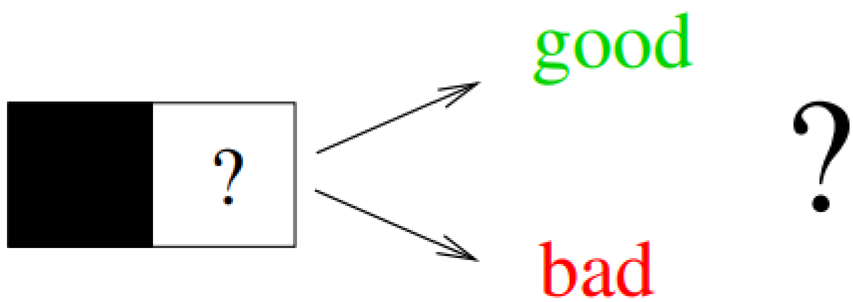
and the answer is…

But we *cant* represent *every* circuit with a linear perceptron (i.e. XOR).

Can we recognize these patterns with a perceptron (aka binary threshold)?

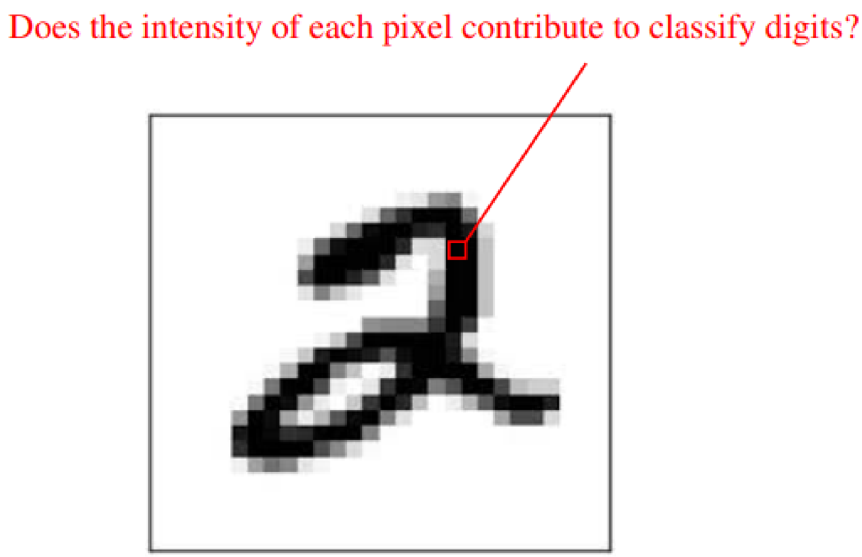
No, each pixel should individually contribute to the classification, that is not the case (more in the next slides). So considering more than one pixel at a time it's not a linear task.

Let us e.g. consider the first pixel, and suppose it is black (the white case is sym-



metric).
does this improve our knowledge for the purposes of classification? No, since we have still the same probability to have a good or a bad example.

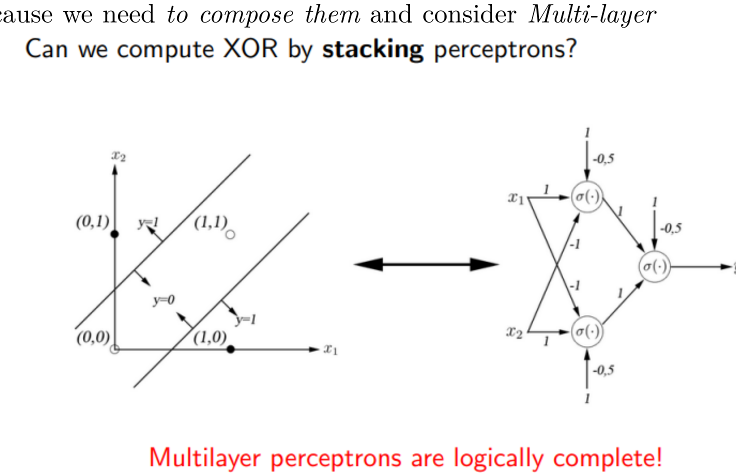
MNIST Example Can we address digit recognition with linear tools? (perceptrons, logistic regression, . . .) When we want to use a linear technique for learning, we have to ask ourself, is each one of the features informative by itself or should consider them in a particular



context?

Multi-layer perceptrons

- we know we can *compute nand* with a perceptron
- we know that nand is logically complete (i.e. we can compute any connective with nands)
- so: why perceptrons are not complete?
  - answer: because we need *to compose them* and consider *Multi-layer perceptrons.*



So... since shallow networks are already complete, why going for *deep networks*? With deep nets, the same function may be computed with *less neural units* (Cohen, et al.) - *Activation functions* play an essential role, since they are the only source of nonlinearity, and hence of the expressiveness of NNs.

Formal expressiveness in the continuous case

What can we say instead of continuos functions? - approximating functions with logistic neurons