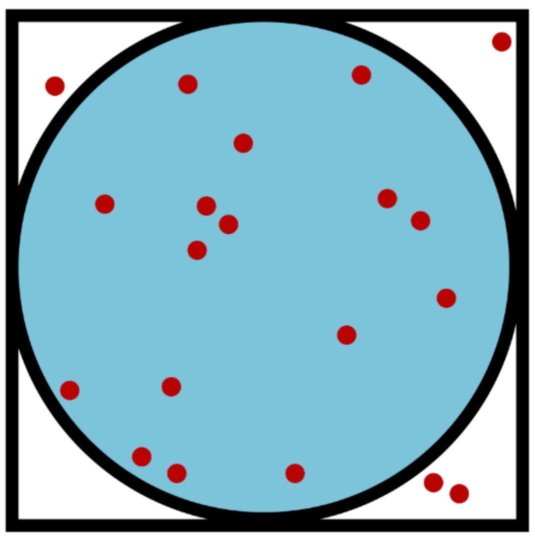
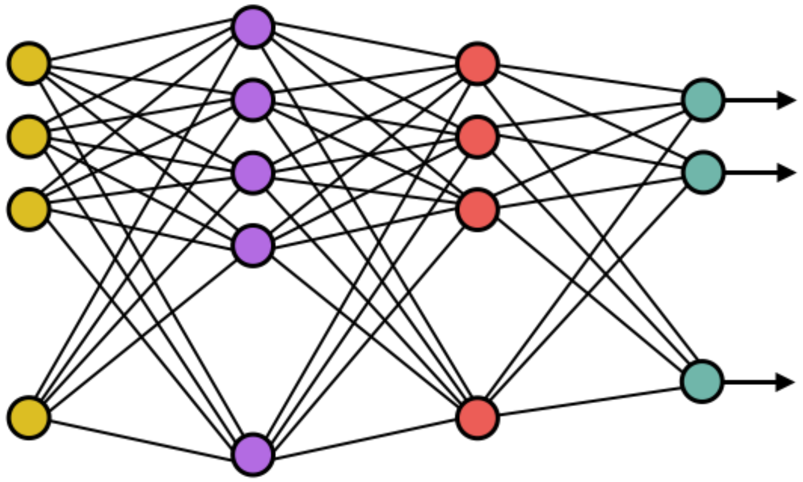
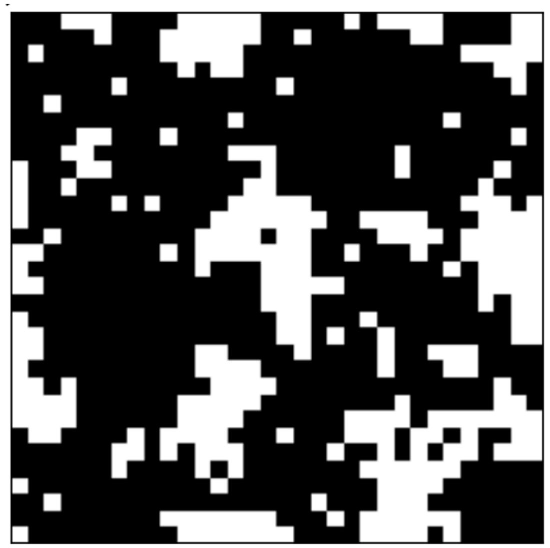
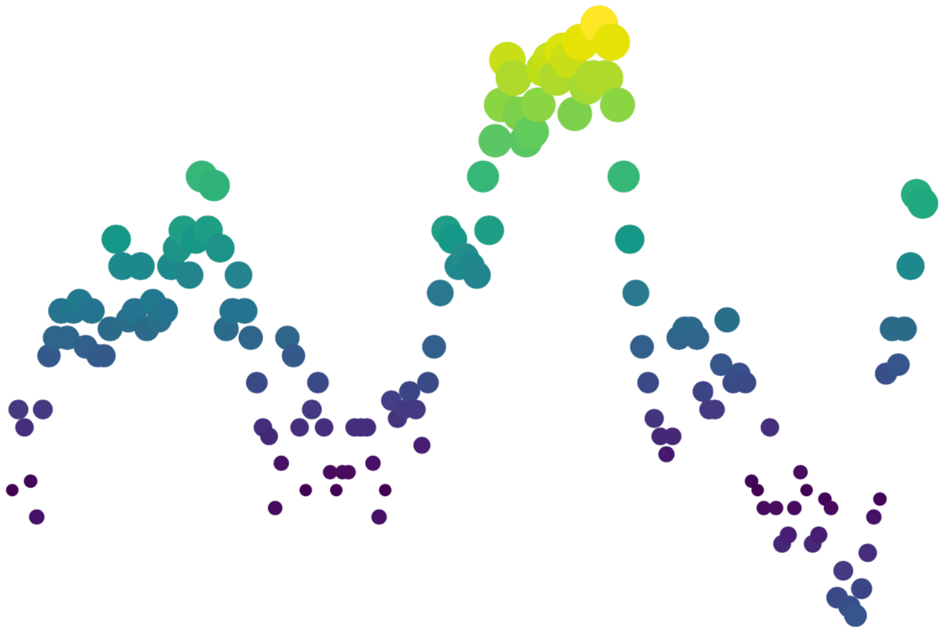
Numerical Methods



Lauren Hayward
PSI START Online School







PERIMETER INSTITUTE

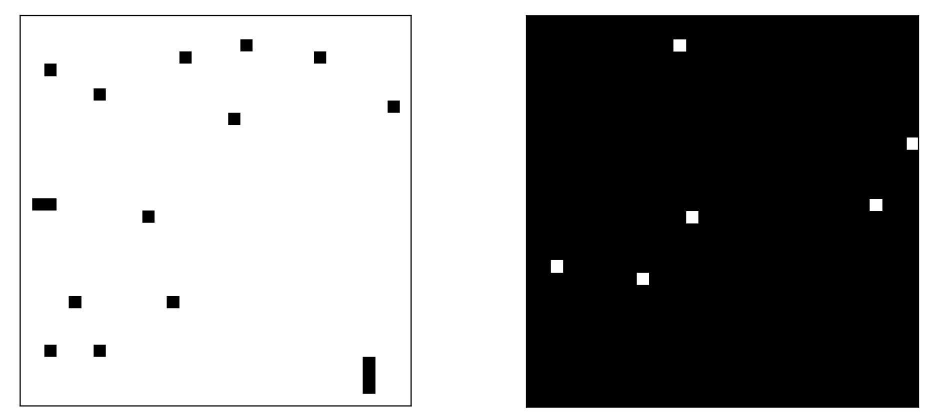
Last lecture: Monte Carlo methods in statistical physics

Ising model in 2D: $$E = -J \sum_{\langle ij \rangle} s_i \, s_j$$

At high temperatures:

Paramagnetic phase

At low temperatures:



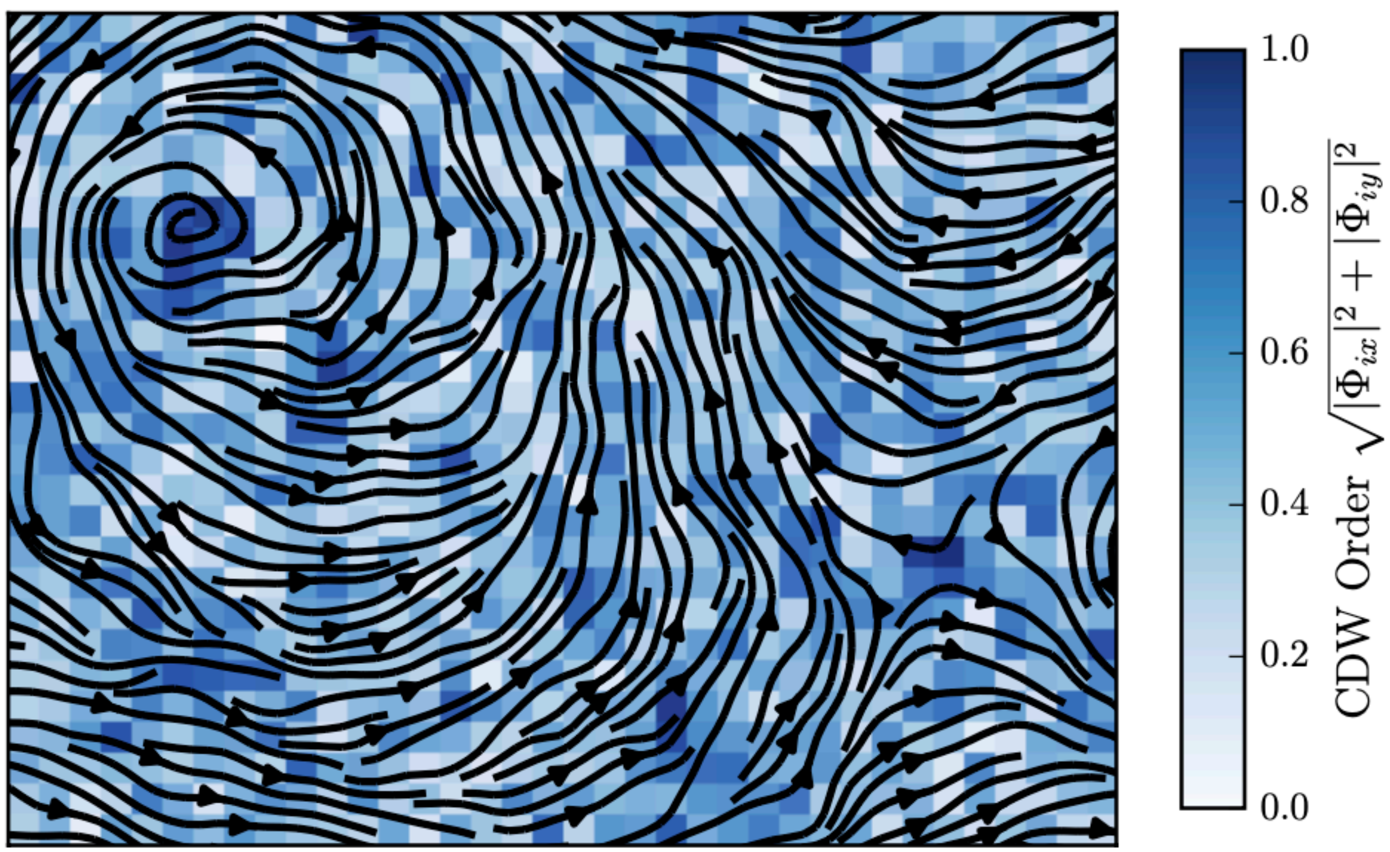
Ferromagnetic phase

PERIMETER INSTITUTE

Monte Carlo to study high-temperature superconductivity

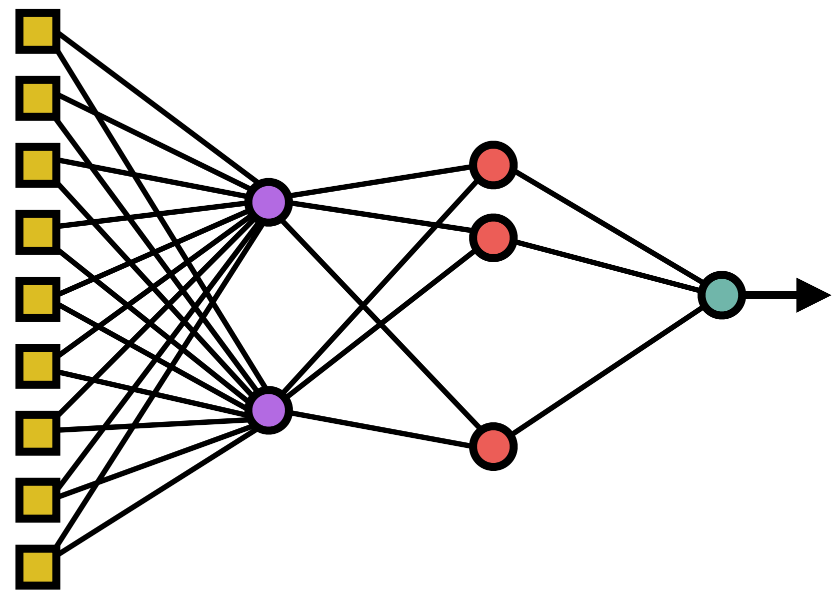
$$H = \frac{\rho_s}{2} \sum_{\langle ij \rangle} \left[\sum_{\alpha=1}^{2} (n_{i\alpha} - n_{j\alpha})^2 + \lambda \sum_{\alpha=3}^{6} (n_{i\alpha} - n_{j\alpha})^2 \right]$$

$$+ \frac{\rho_s a^2}{2} \sum_{i} \left[g \sum_{\alpha=3}^{6} n_{\alpha}^2 + g' \left(\sum_{\alpha=3}^{6} n_{\alpha}^2 \right)^2 + w \left[\left(n_{i3}^2 + n_{i4}^2 \right)^2 + \left(n_{i5}^2 + n_{i6}^2 \right)^2 \right] \right]$$



PERIMETER INSTITUTE

Outline for today



Machine learning and neural networks

- Feedforward neural network architecture: layers, weights, biases, activation functions

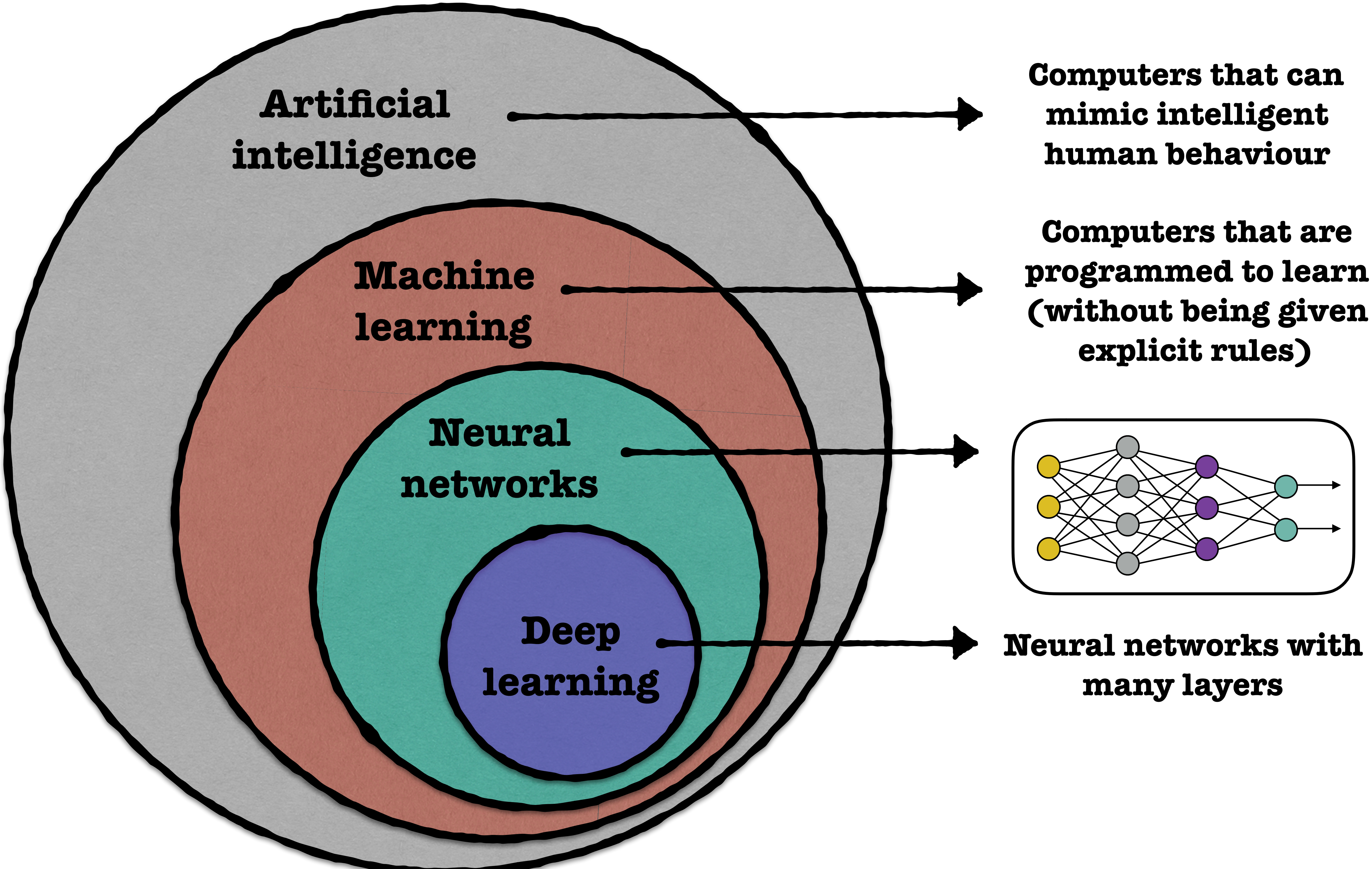
- Feedforward neural networks in PyTorch

What is machine learning?

Machine learning: training computers to detect and characterize features from data

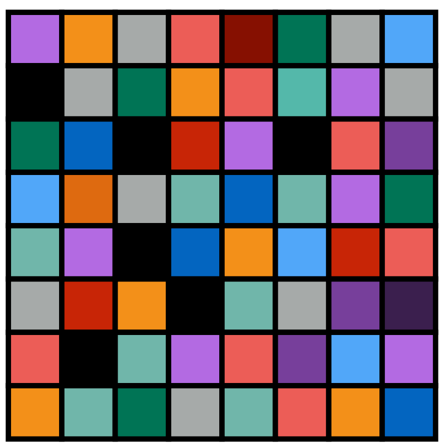
Compare with a goal of **statistical physics:** predicting and explaining macroscopic phenomena (features) from microscopic quantities (data)

PERIMETER INSTITUTE



PERIMETER INSTITUTE

Application of neural networks: classifying images



Input: an image

Computer

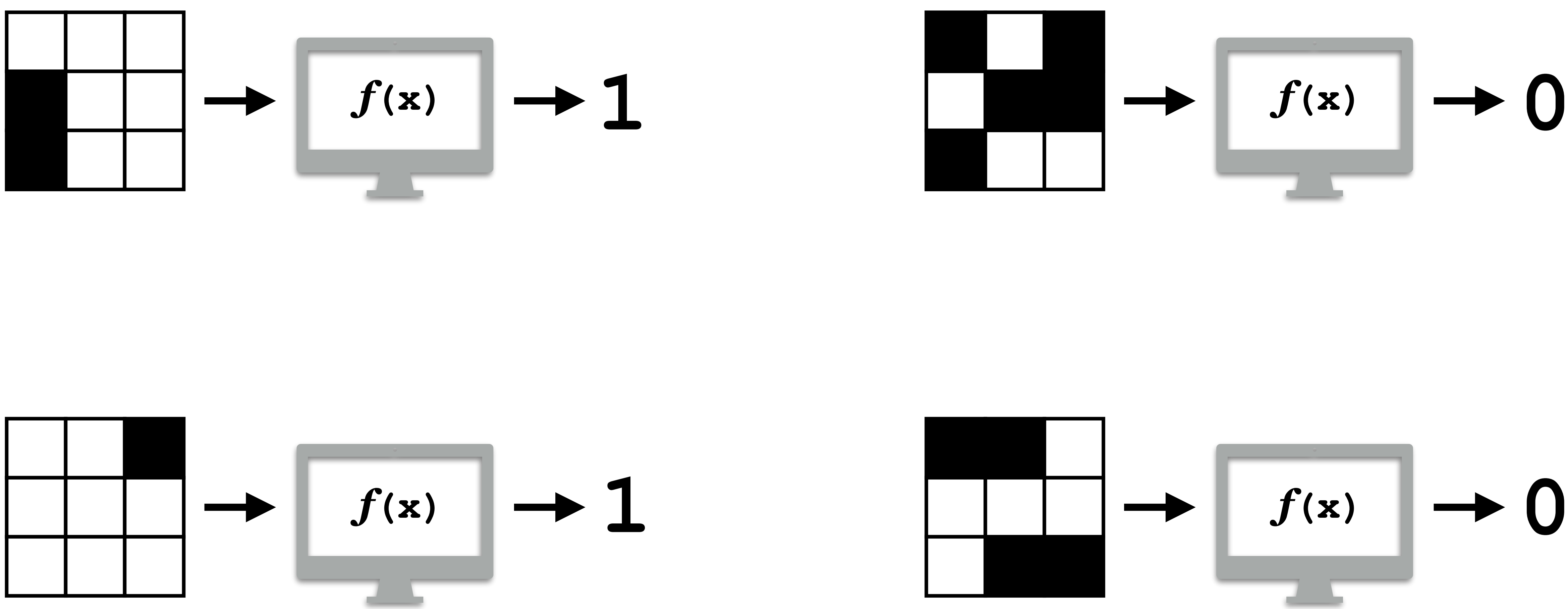
Output: some description of the image

Or, more mathematically:

Input: $\mathbf{x}$ → **Function** f → **Output:** $f(\mathbf{x})$

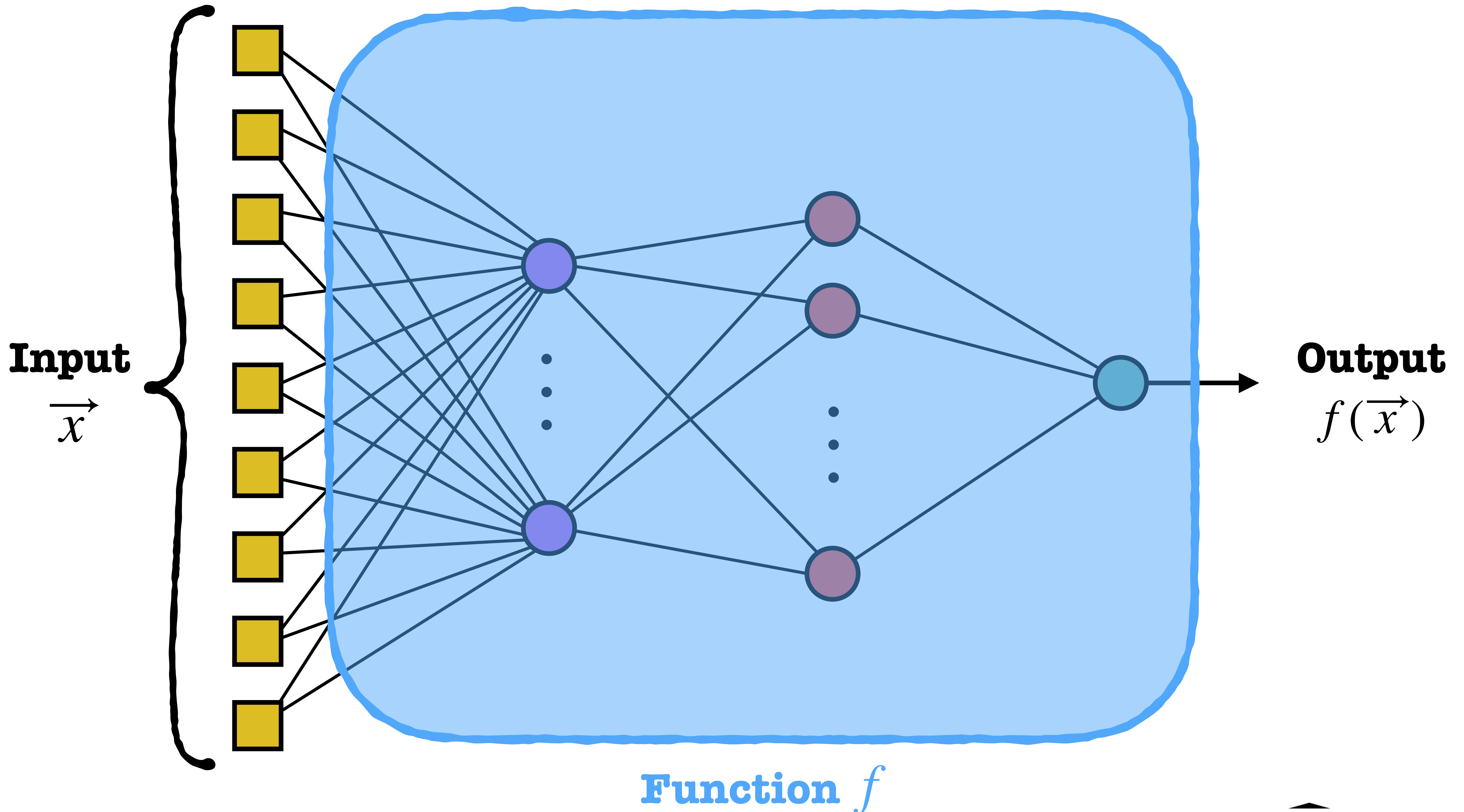
Application of neural networks: classifying images

Example: identifying whether an image contains one rectangle



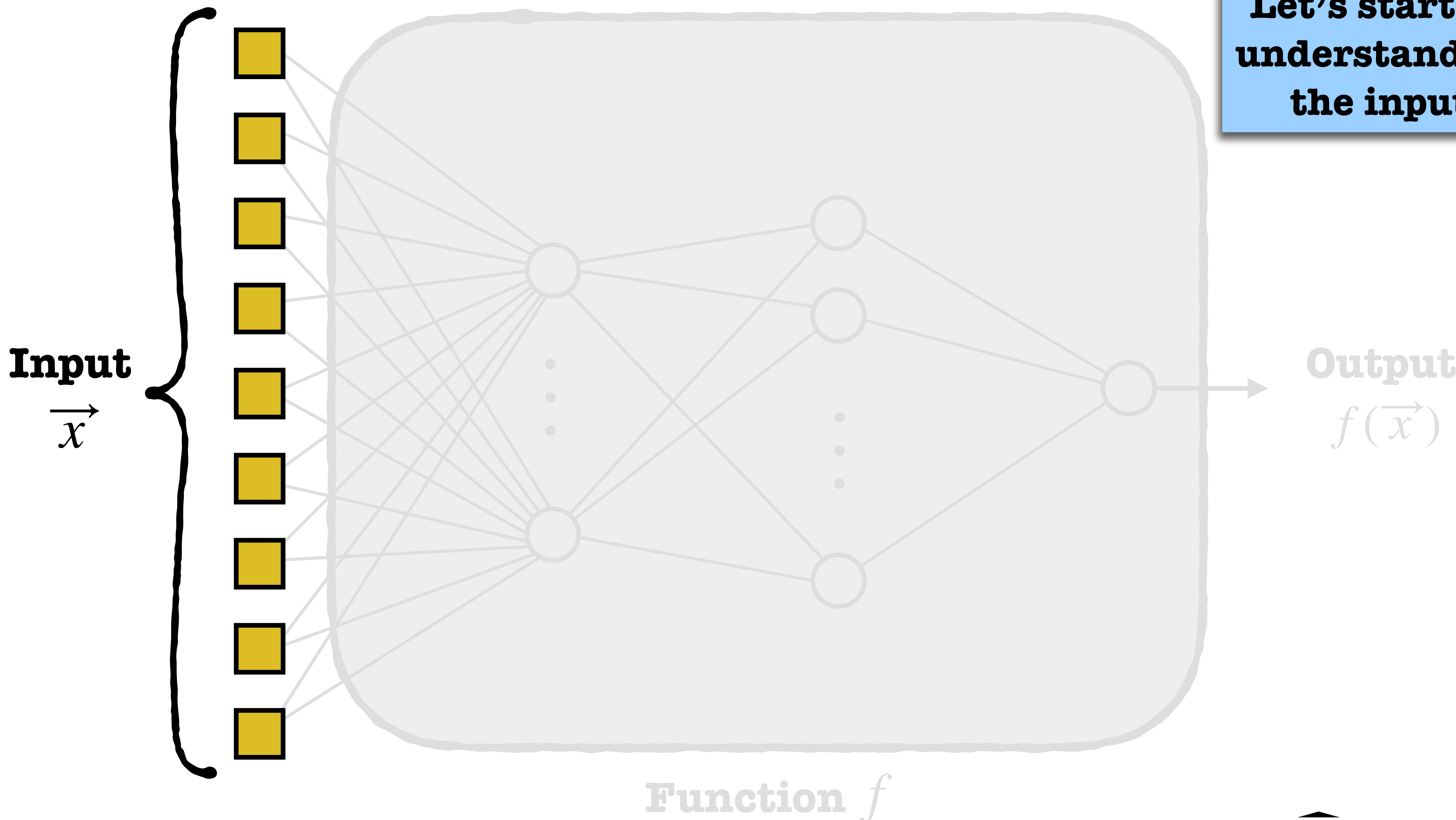
PERIMETER INSTITUTE

Feedforward neural network



PERIMETER INSTITUTE

Feedforward neural network



Let's start by understanding the input

PERIMETER INSTITUTE

Neural network input

Input $\vec{x}$

$\mathbf{x}_1 = 0$

$\mathbf{x}_2 = 1$

$\mathbf{x}_3 = 1$

$\mathbf{x}_4 = 0$

$\mathbf{x}_5 = 1$

$\mathbf{x}_6 = 0$

$\mathbf{x}_7 = 0$

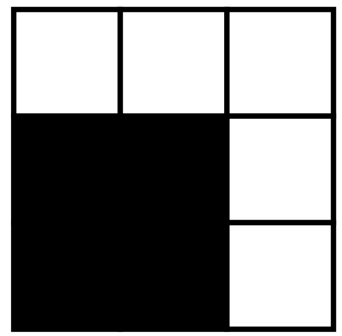
$\mathbf{x}_8 = 0$

$\mathbf{x}_9 = 1$

$$\vec{x} = [0,1,1,0,1,0,0,0,1]$$

PERIMETER INSTITUTE

Neural network input

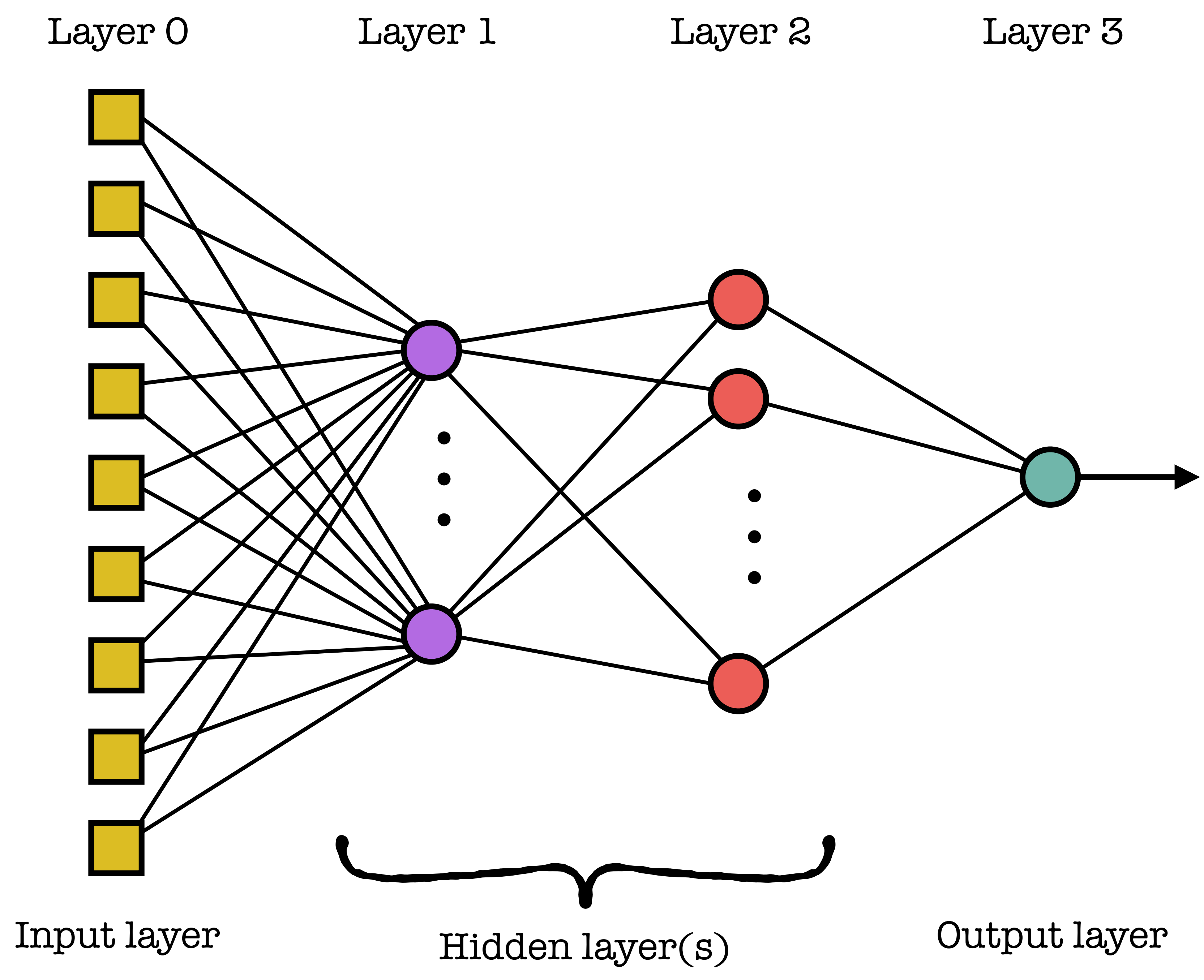
How do we translate  **into an input for the neural network?**

$x_1 = 0$	$x_2 = 0$	$x_3 = 0$
$x_4 = 1$	$x_5 = 1$	$x_6 = 0$
$x_7 = 1$	$x_8 = 1$	$x_9 = 0$

$$\overrightarrow{x} = [0,0,0,1,1,0,1,1,0]$$

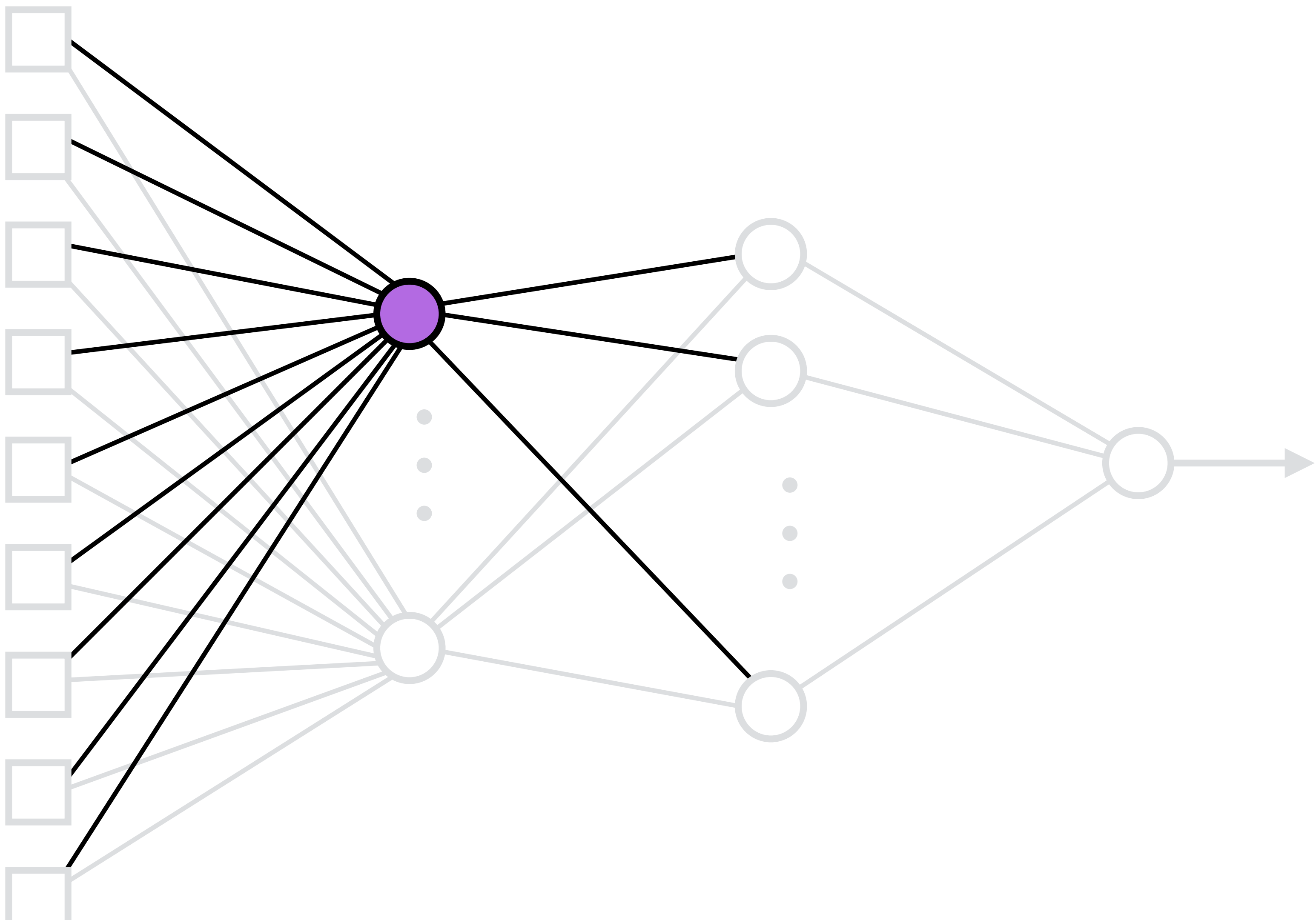
PERIMETER INSTITUTE

Neural networks layers



Neuron output

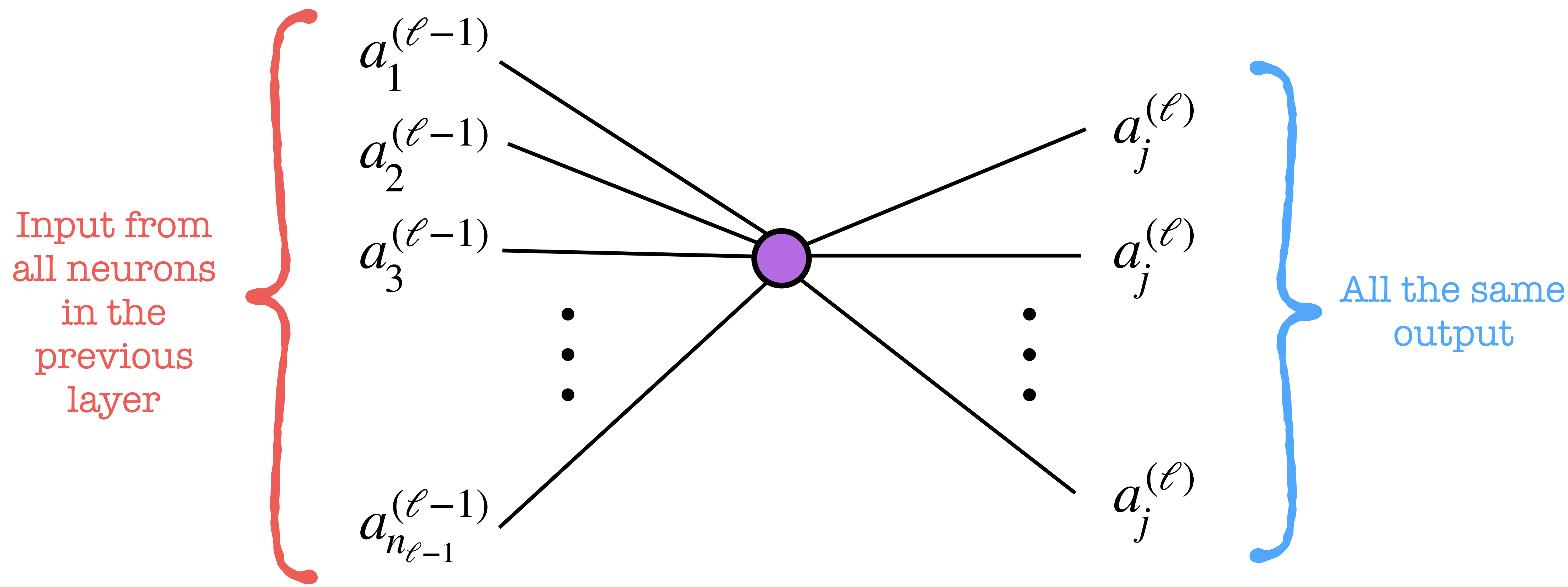
Let's zoom in on the j^{th} neuron in layer $\ell > 0$



PERIMETER INSTITUTE

Neuron output

Let's zoom in on the j^{th} neuron in layer $\ell > 0$



The output from this neuron is: $a_j^{(\ell)} = g_\ell\left(\sum_{i=1}^{n_{\ell-1}} a_i^{(\ell-1)} W_{ij}^{\ell} + b_j^{\ell}\right)$

PERIMETER INSTITUTE

Neuron output

$$a_j^{(\ell)} = \boxed{g_\ell}\left(\sum_{i=1}^{n_{\ell-1}} a_i^{(\ell-1)}\boxed{W_{ij}^{\ell}} + \boxed{b_j^{\ell}}\right)$$

non-linear **activation function**

weight for each link

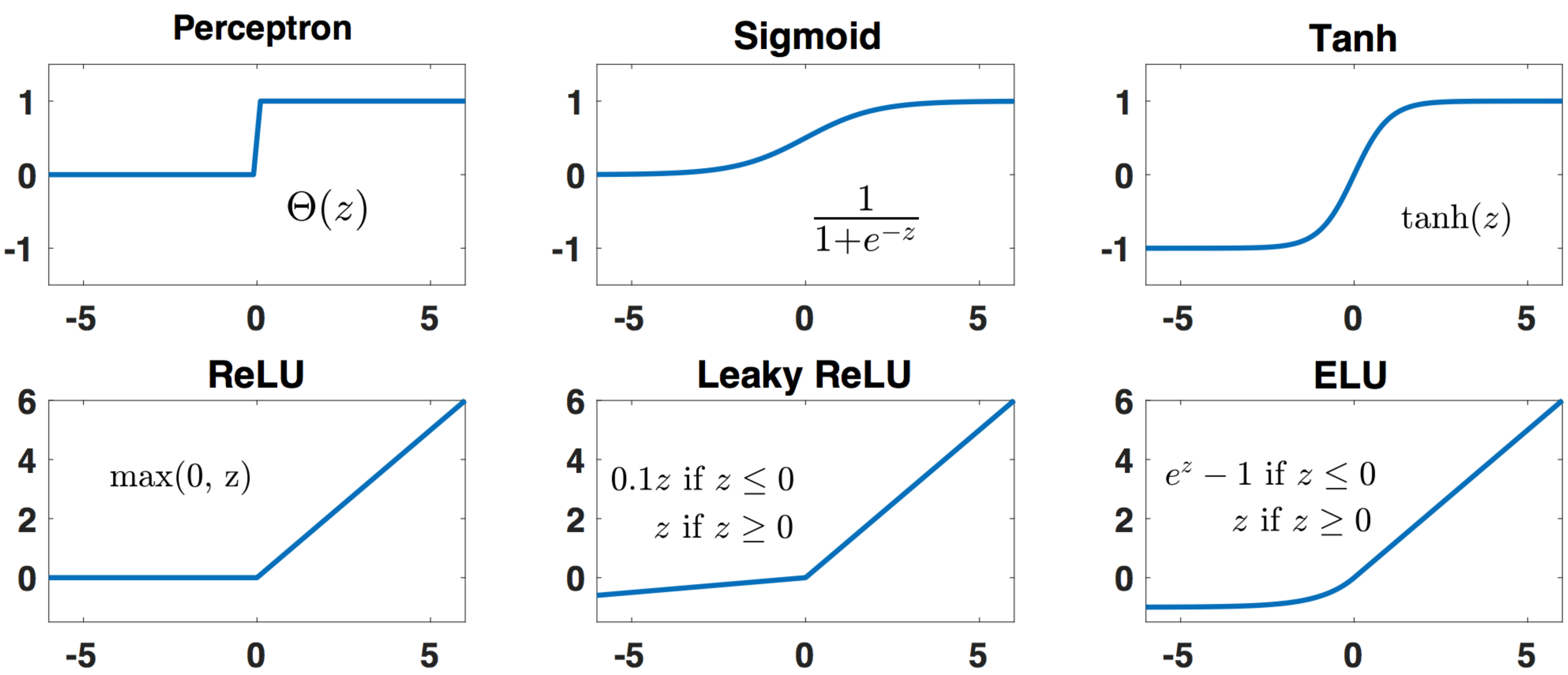
bias for each neuron

The weights and biases are adjusted as the network **learns**.

PERIMETER INSTITUTE

Activation functions

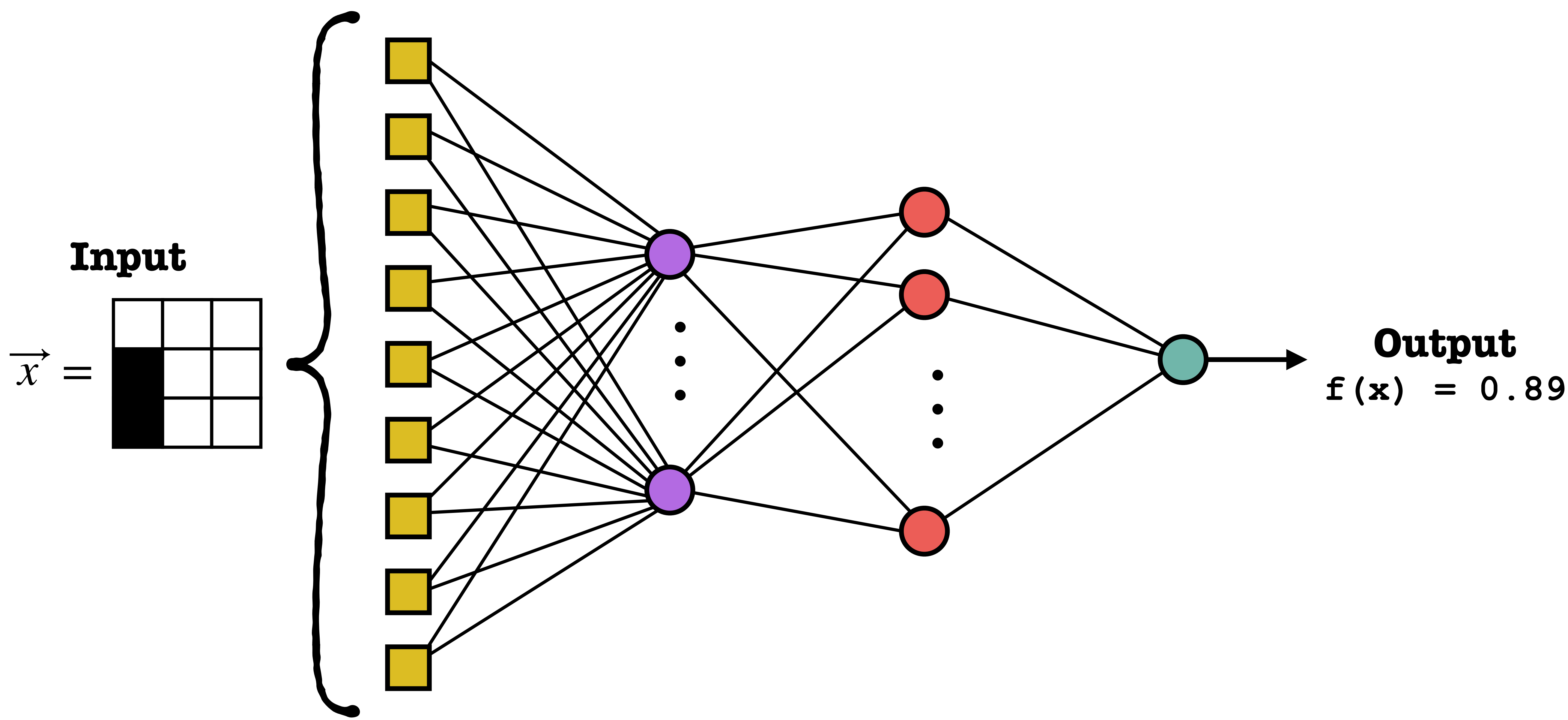
We can choose various non-linear activation functions g_ℓ, such as:



arXiv:1803.08823

PERIMETER INSTITUTE

Neural network output

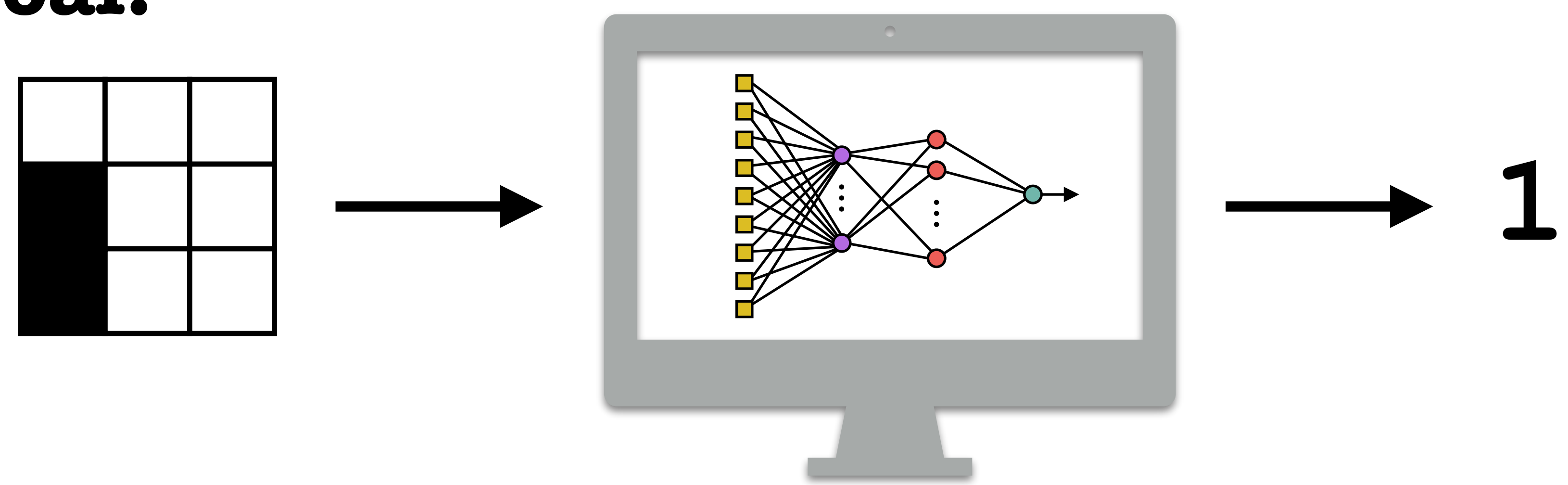


What does this output mean?

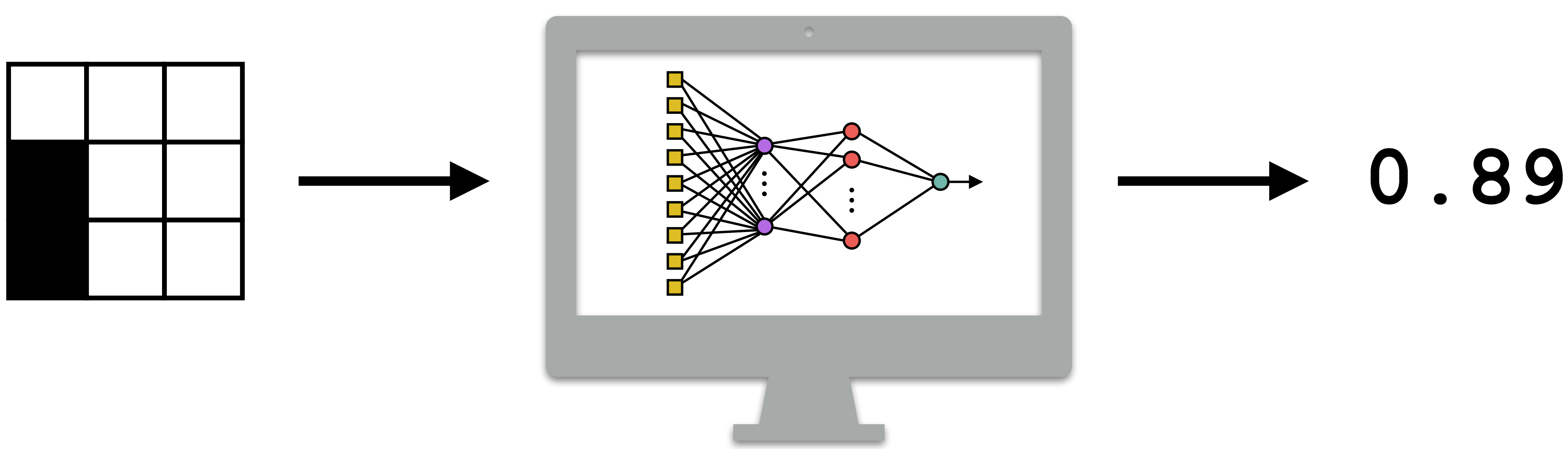
Neural network output

Recall that our goal is to identify whether an image contains one rectangle

Goal:



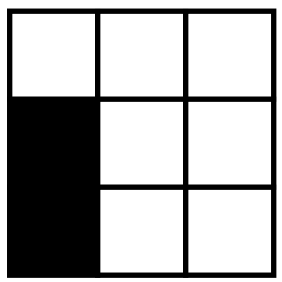
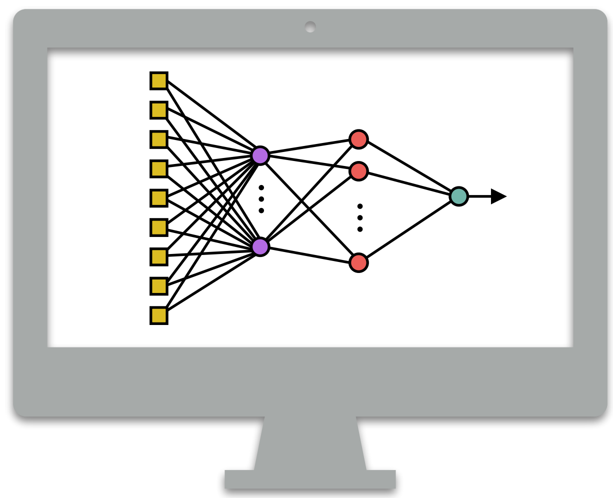
What we might find:



Our network is 89% sure that the input contains one rectangle.

Neural network output

In other words, our network might give the output:

$\vec{x} =$   $f(\vec{x}) = 0.89$

While the true answer should be: $y_{\text{true}}(x) = 1$

We would like to choose the weights W and biases b such that the difference between $f(x)$ and $y_{\text{true}}(x)$ is as small as possible.

Neural network output

We would like our neural network to classify all possible inputs using the same weights and biases.