The week ahead

- **Quiz 10:** means is 89% and average completion time 5 min.
- **Assignment 4 due Wed, Nov 11th, 11:59 pm (midnight)**
  - **Exceptional late policy:** No penalty until Mon, Nov 23rd, 11:59 pm
- **Quiz 11, Friday, Oct 30th 6am until Nov 1st 11:59pm (midnight)**
  - Neural networks

Coming up soon

- **Touch-point 3**: deliverables due **Nov 22nd**, live-event Mon, Nov 23rd
  - Single-slide presentation outlining progress highlights and current challenges
  - Three-minute pre-recorded presentation with your progress and current challenges
- **Project final report due Dec 7th 11:59pm (midnight)**
  - GitHub page with all of the results you have achieved utilizing both unsupervised learning and supervised learning
  - Final seven-minute long pre-recorded presentation

Supervised learning

- Regression
  - Linear regression
  - Regularized linear regression
  - Neural Networks
  - Regression trees*
  - Convolutional neural networks
- Classification
  - Logistic regression
  - Bayes classifiers
  - Decision tree and random forest
  - Support vector machines
  - Kernel SVM
  - Neural networks
  - Convolutional neural networks

CS4641B Machine Learning

Lecture 21: Intro to neural networks

Rodrigo Borela ▸ rborelav@gatech.edu

These slides are from Vivek Srikumar, Chao Zhang and Mahdi Roozbahani

Outline

- Building blocks
- Neural Networks
- Expressivity of Neural Networks
- Predicting with Neural Networks

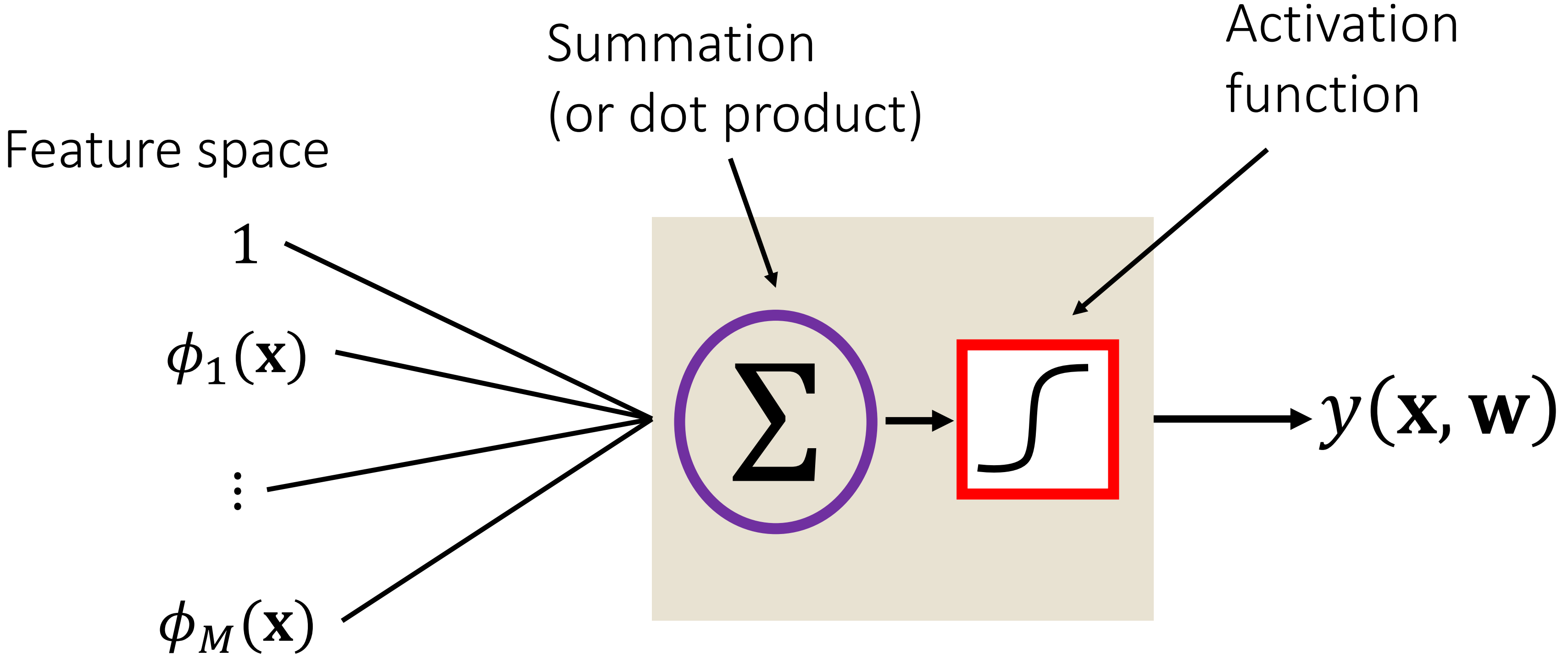
- *Complementary reading: Bishop PRML – Chapter 5, Section 5.1 to 5.3.3*

Outline

- Building blocks
- Neural Networks
- Expressivity of Neural Networks
- Predicting with Neural Networks

Linear models for regression and classification

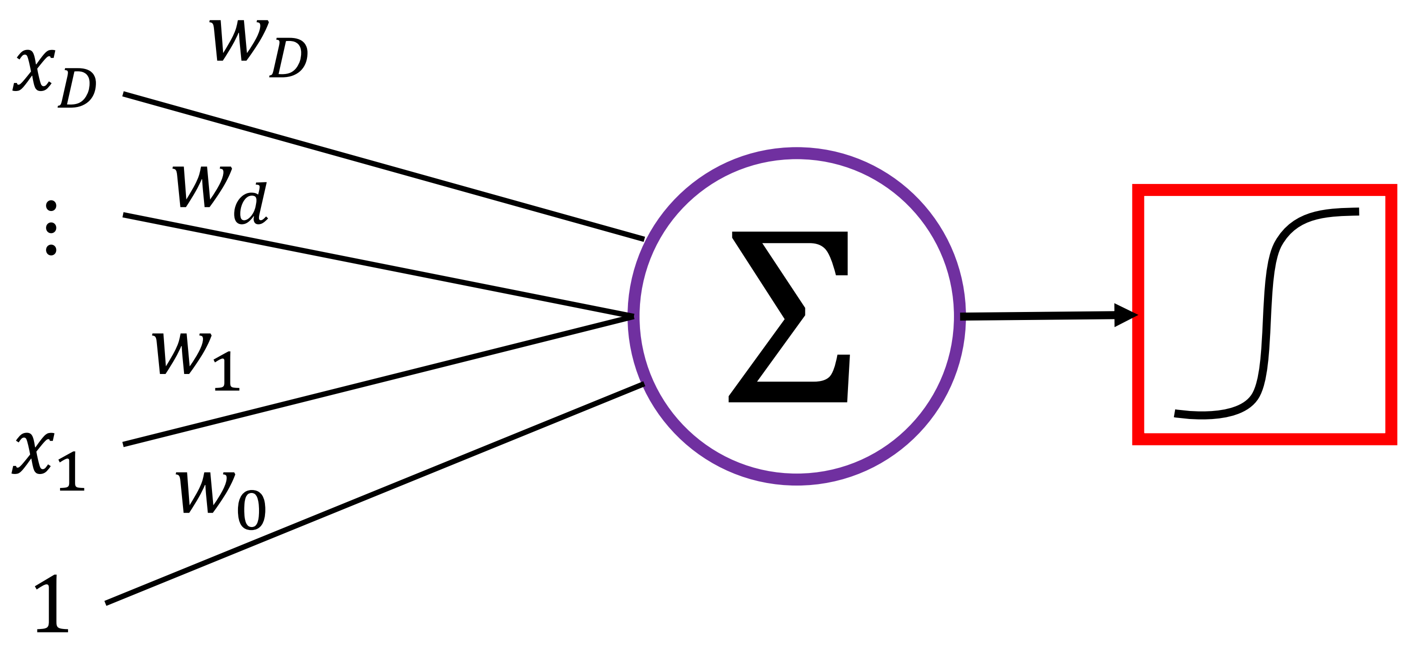
- Linear models



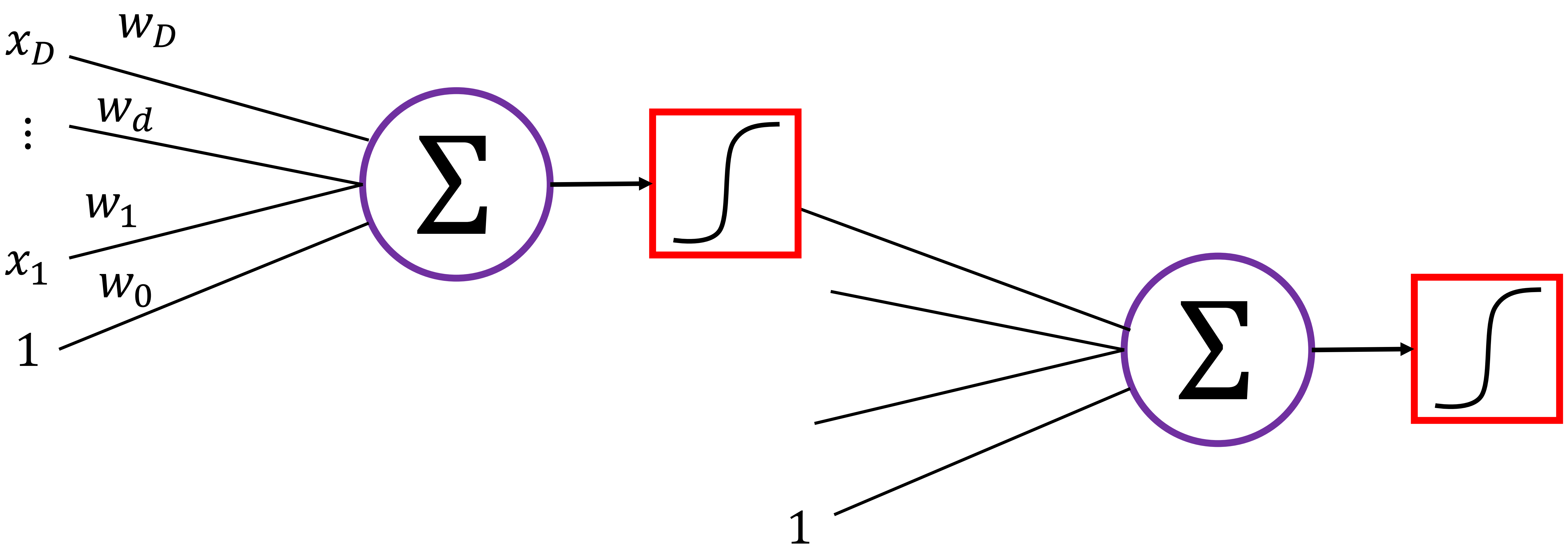
$$y(\mathbf{x},\mathbf{w}) = \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}) = h\left(\sum_{m=1}^{M} w_m\phi_m(\mathbf{x})\right)$$

- **Learning** (in general minimize loss)
  - Closed form solutions, gradient descent, stochastic gradient descent, etc.
- What if we could combine the these to achieve something more flexible?

Building blocks

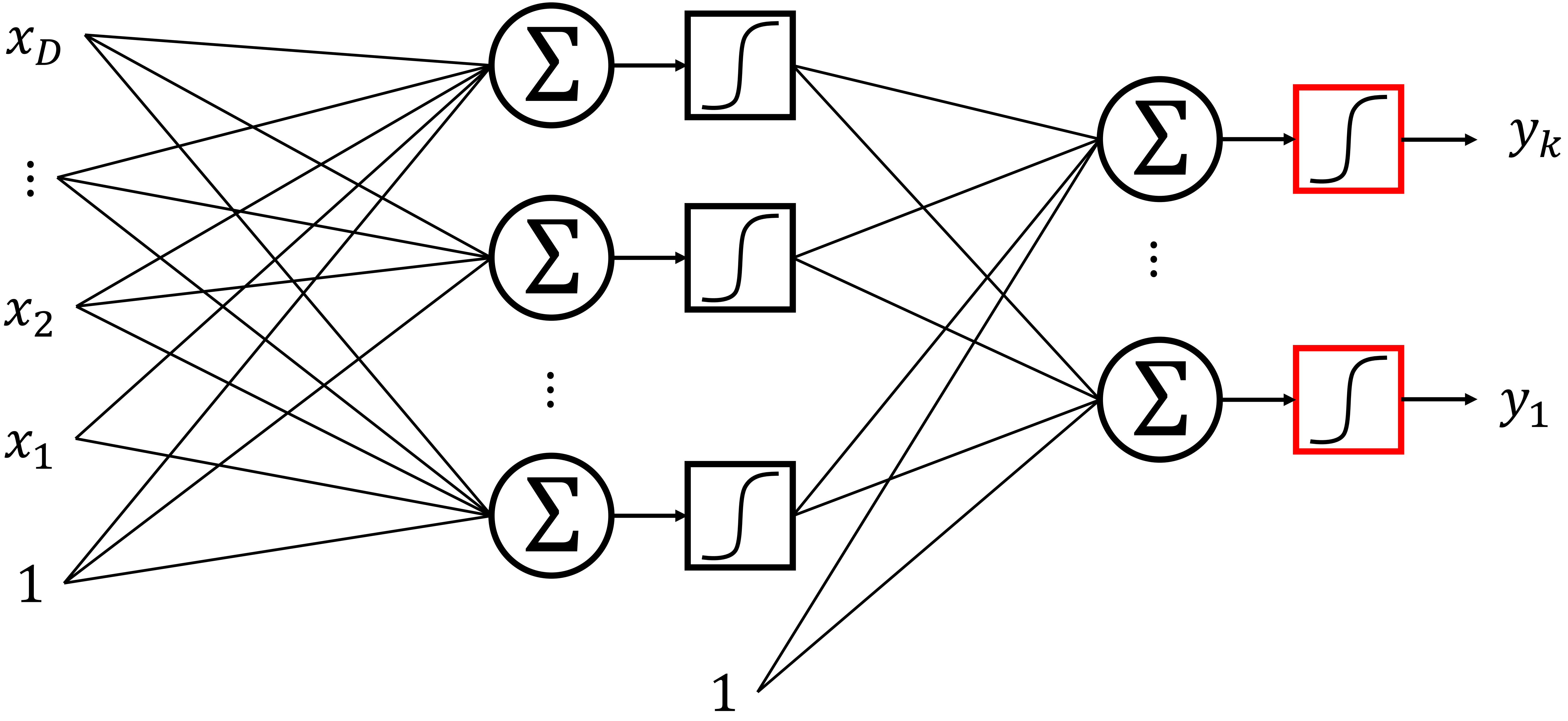


Building a network



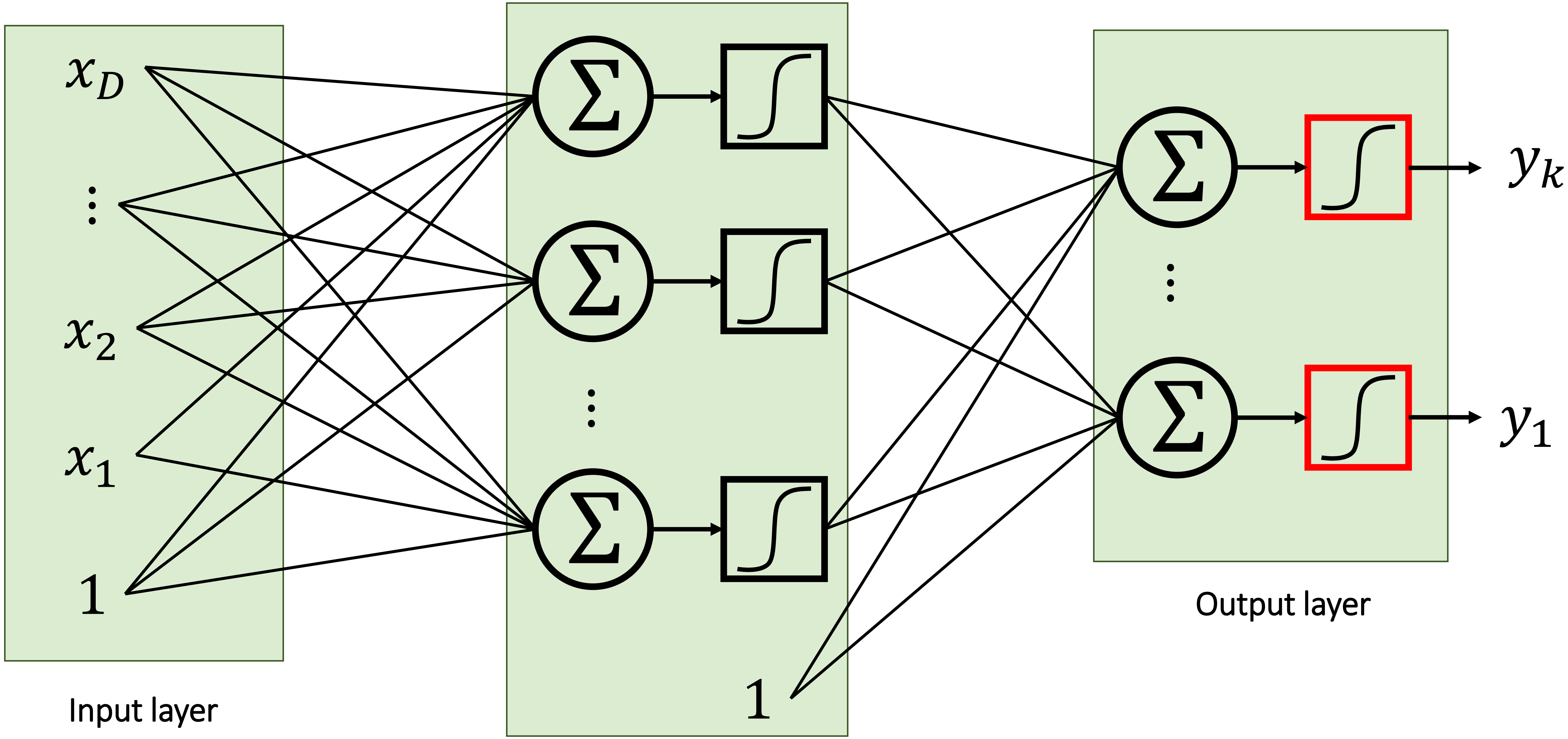
Building a network

- This is a **two layer** feed forward neural network



Building a network

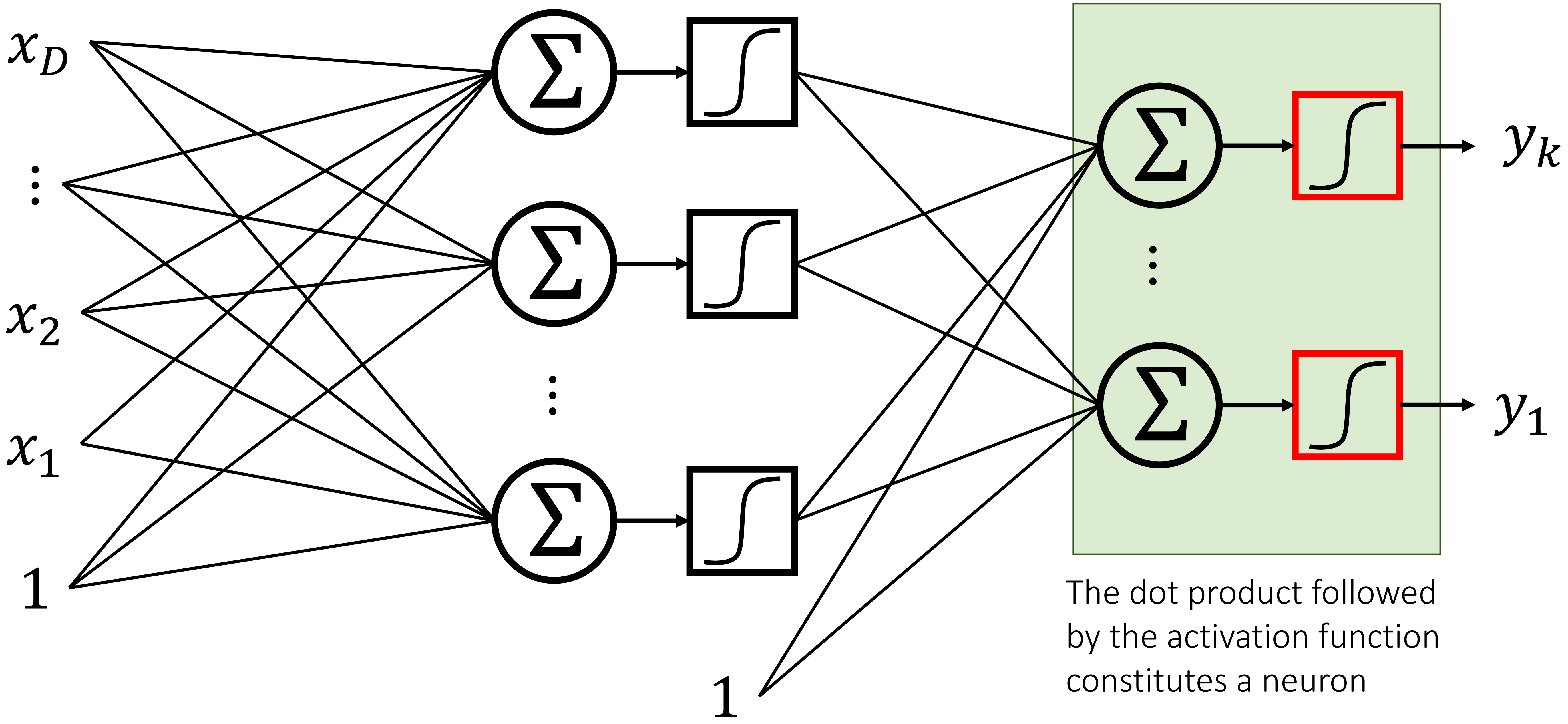
- This is a **two layer** feed forward neural network



Hidden layer: what is going on here?

Building a network

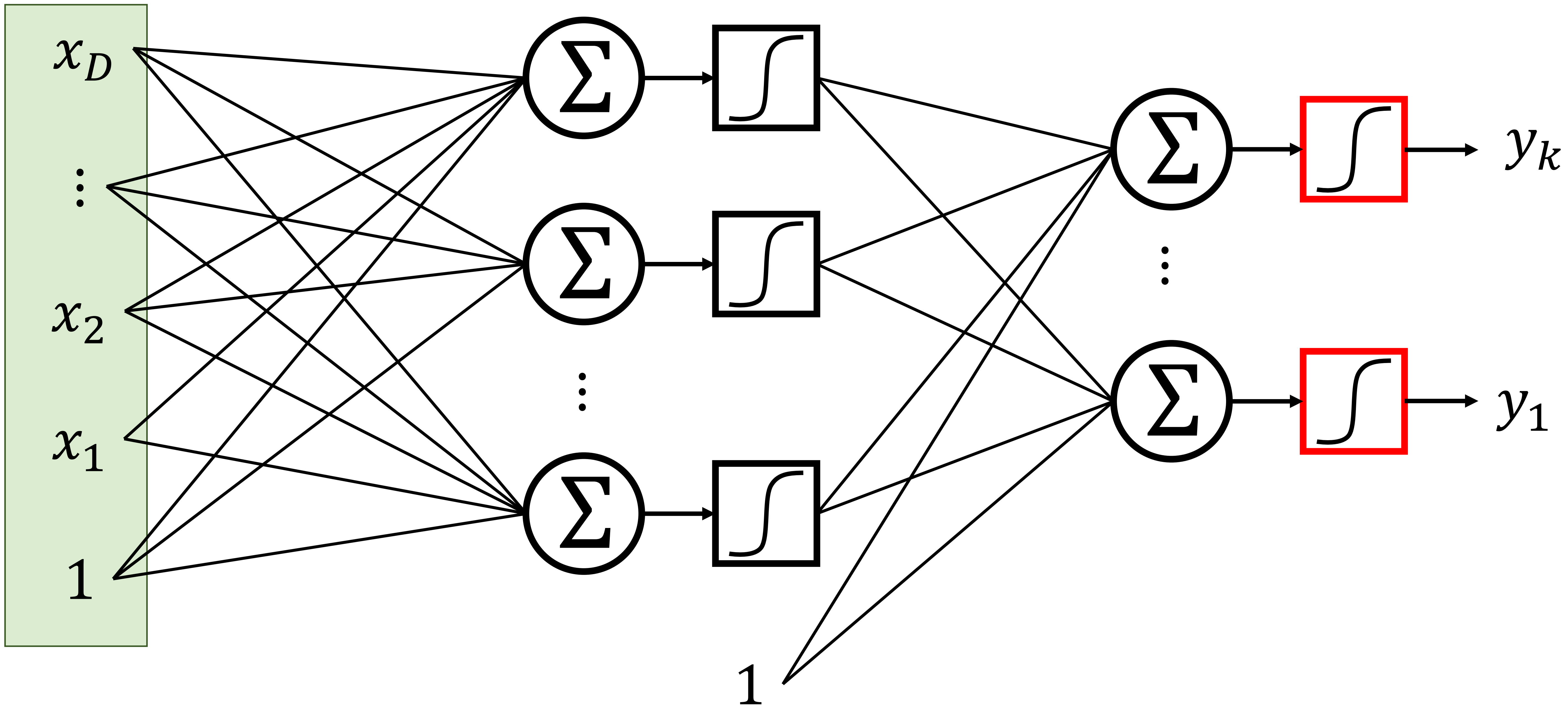
- This is a **two layer** feed forward neural network



The dot product followed by the activation function constitutes a neuron

Building a network

- This is a **two layer** feed forward neural network



What if the inputs were the outputs of another network? **We can make multiple layer networks**

Outline

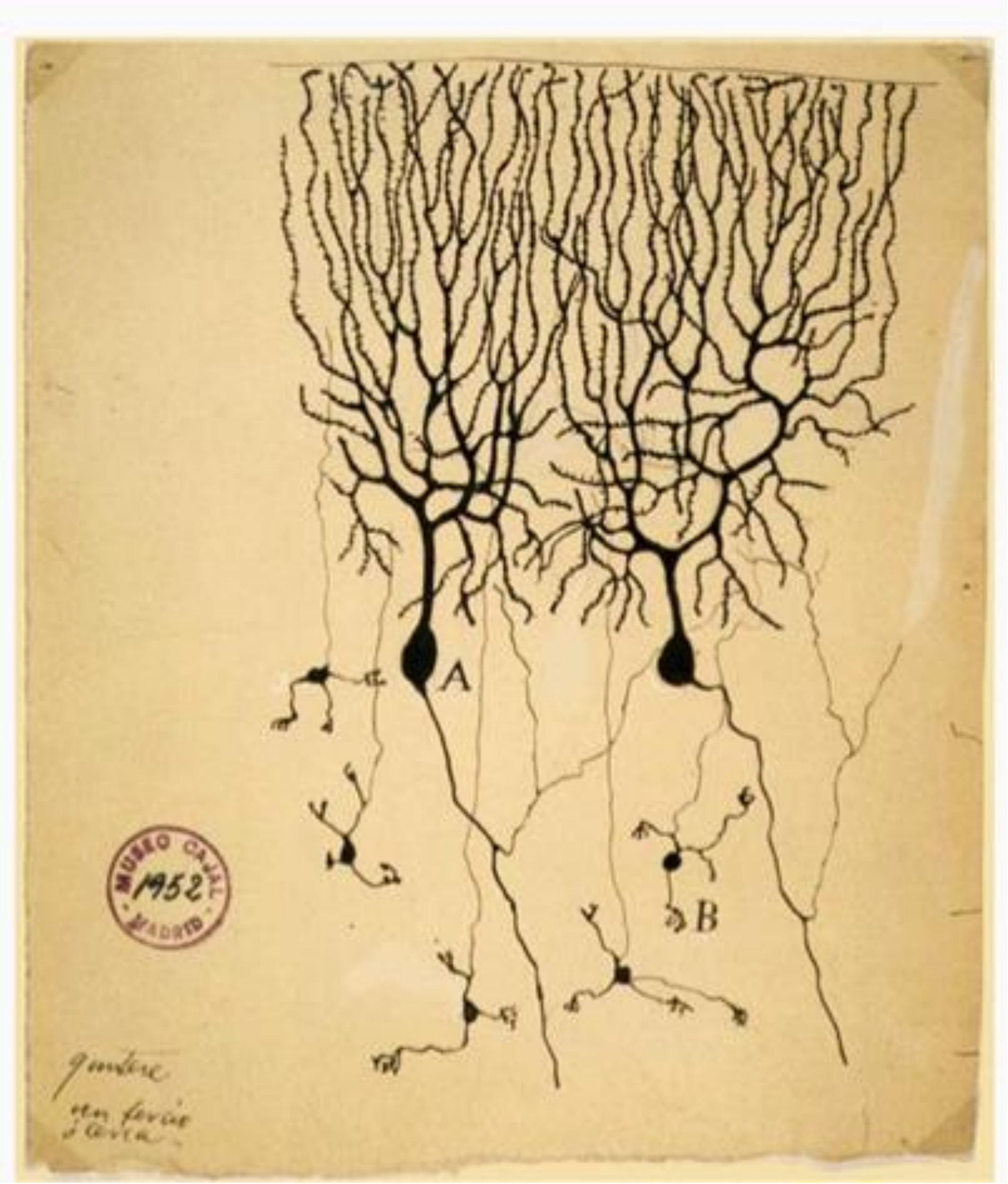
- Building blocks
- Neural Networks
- Expressivity of Neural Networks
- Predicting with Neural Networks

Neural networks

- A robust approach for approximating **real-valued, discrete-valued or vector valued** functions
- Among the most effective **general purpose** supervised learning methods currently known
  - Especially for **complex and hard to interpret data** such as real-world sensory data
- The **backpropagation algorithm** for neural networks has been shown successful in many practical problems
  - Handwritten character recognition, speech recognition, object recognition, some NLP problems

Inspiration from biological neurons

- Neurons: core components of brain and nervous system consisting of
  1. Dendrites that collect information from other neurons
  2. An axon that generates outgoing spikes



The first drawing of brain cells by Santiago Ramon y Cajal (1899)

Artificial neurons

- Functions that **very loosely** mimic a biological neuron
- A neuron accepts a collection of inputs (vector $\mathbf{x}$) and produce an output by:
  1. Applying a dot product with weights $\mathbf{w}_j$ and adding a bias w_{j0} (activation)

$$a_j = \sum_{i=1}^{D} w_{ij}^{(layer)} x_i + w_{0j}^{(layer)}$$

  2. Applying a (possibly non-linear) transformation called an **activation function,** $h(\cdot)$

$$z_j = h(a_j)$$

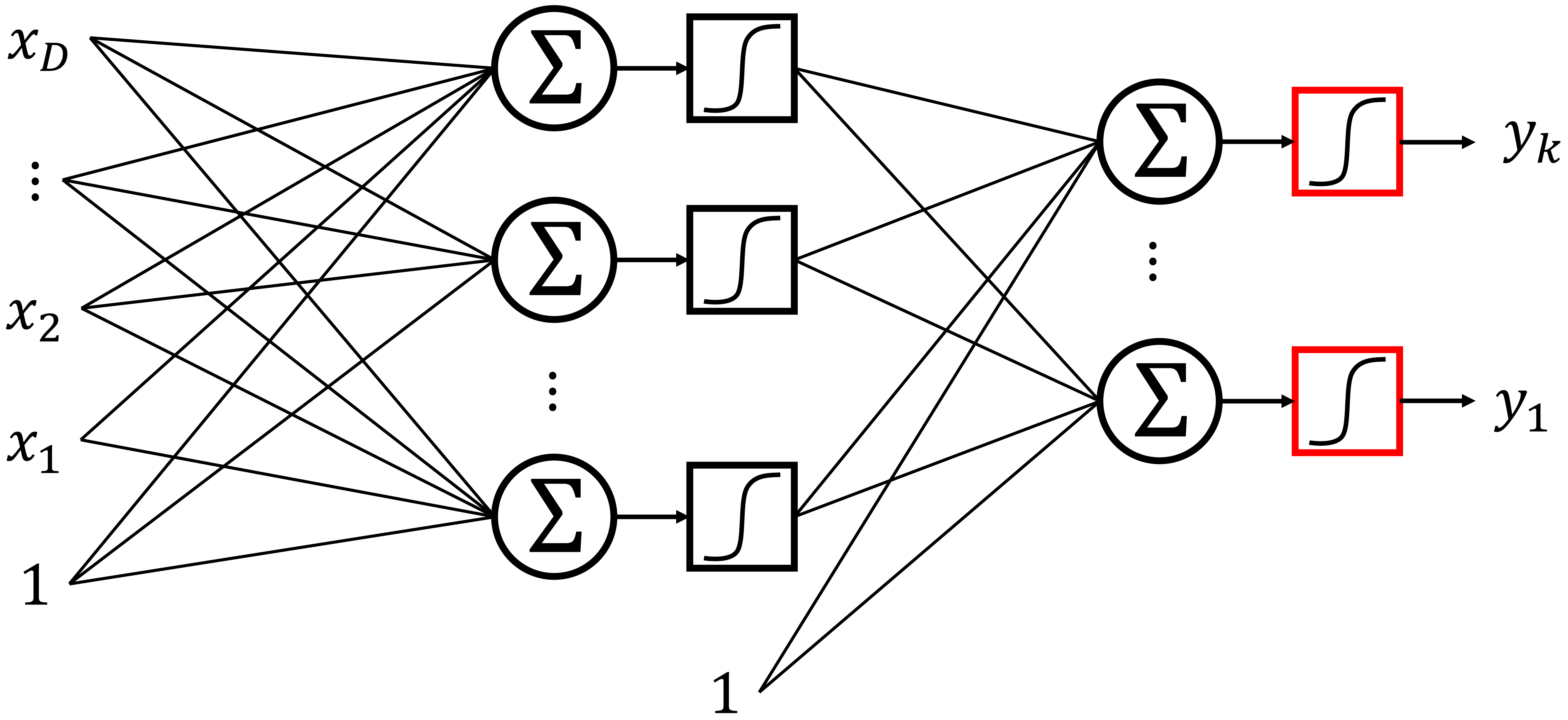
  3. Output unit activations

$$a_k = \sum_{i=1}^{D} w_{ij}^{(layer)} z_i + w_{0j}^{(layer)}$$

  4. Output activation function

$$y_k = \sigma(a_k)$$

Artificial neurons



Activation Functions

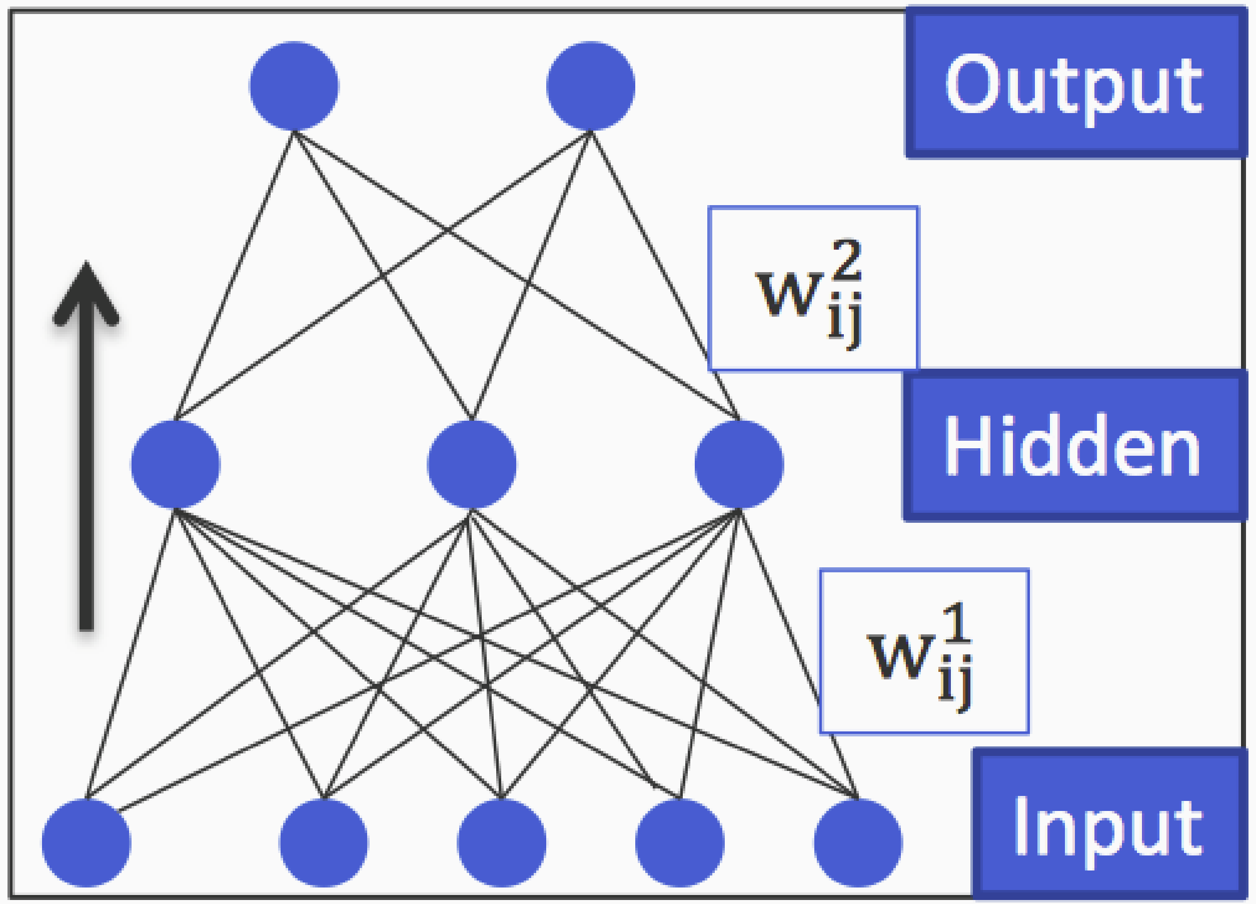
$$z = h(a)$$

Name of neuron	Activation function: $h(\cdot)$ or $\sigma(\cdot)$
Linear unit	a
Threshold/sign unit	$\mathrm{sgn}(a)$
Sigmoid unit	$\dfrac{1}{1 + \exp(-a)}$
Rectified linear unit (ReLU)	$\max(0, a)$
Tanh unit	$\tanh(a)$

- Many more activation functions exist (sinusoid, soft-max, Gaussian, polynomial, etc.)

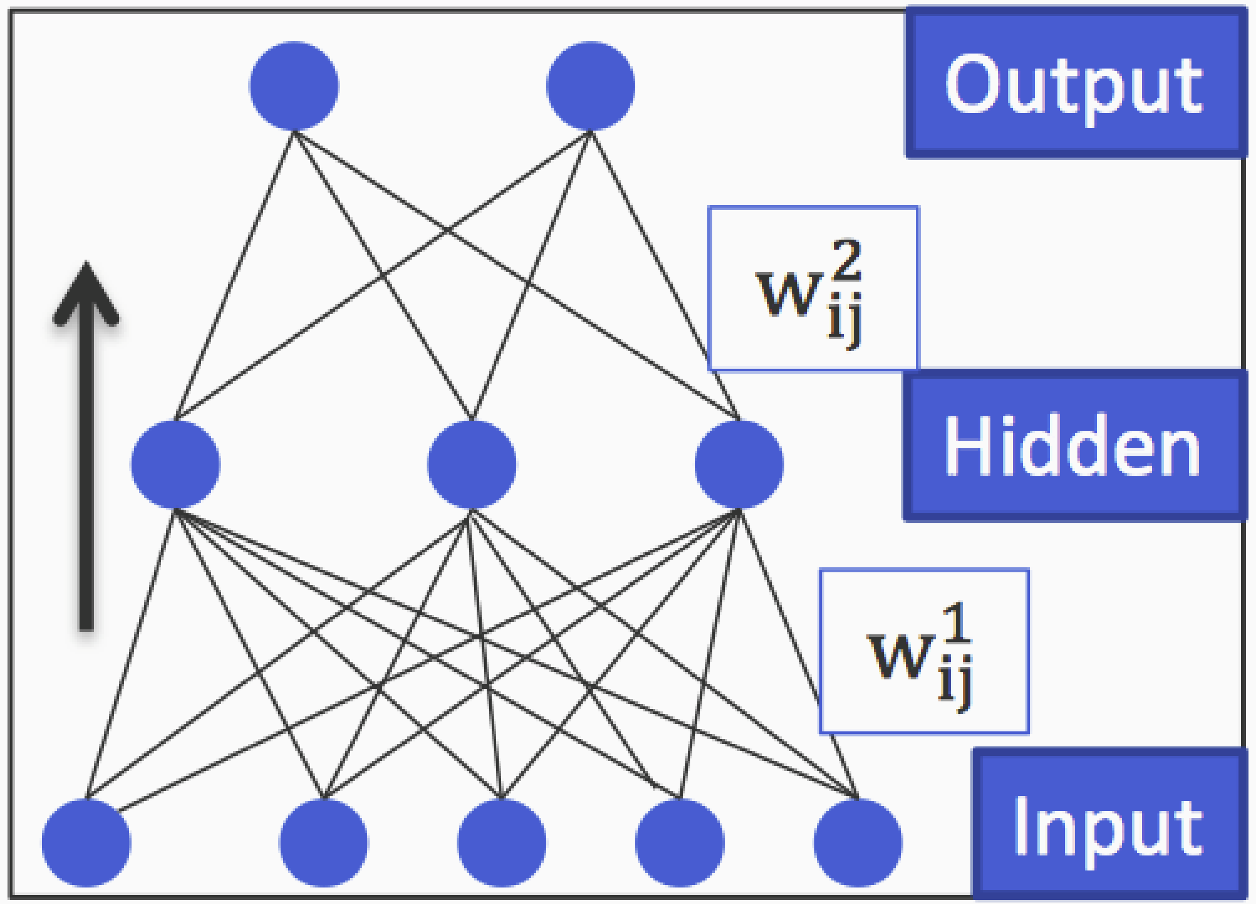
Neural networks

- A function that converts inputs to outputs defined by a **directed acyclic graph**
  - Nodes organized in layers correspond to neurons
  - Edges carry output of one neuron to another, associated with weights
- To define a neural network, we need to specify:
  - The structure of the graph
    - How many nodes, the connectivity
  - The activation function on each node
  - The edge weights



Neural networks

- The structure of the graph
  - Called the architecture of the network
  - Typically predefined, part of the design of the classifier
- The edge weights
  - Learned from data



A brief history of neural networks

- 1943: McCullough and Pitts showed how linear threshold units can compute logical functions
- 1949: Hebb suggested a learning rule that has some physiological plausibility
- 1950s: Rosenblatt, the perceptron algorithm for a single threshold neuron
- 1969: Minsky and Papert studied the neuron from a geometrical perspective
- 1980s: Convolutional neural networks (Fukushima, LeCun), the backpropagation algorithm (various)
- 2003: More compute, more memory, more data, deeper networks

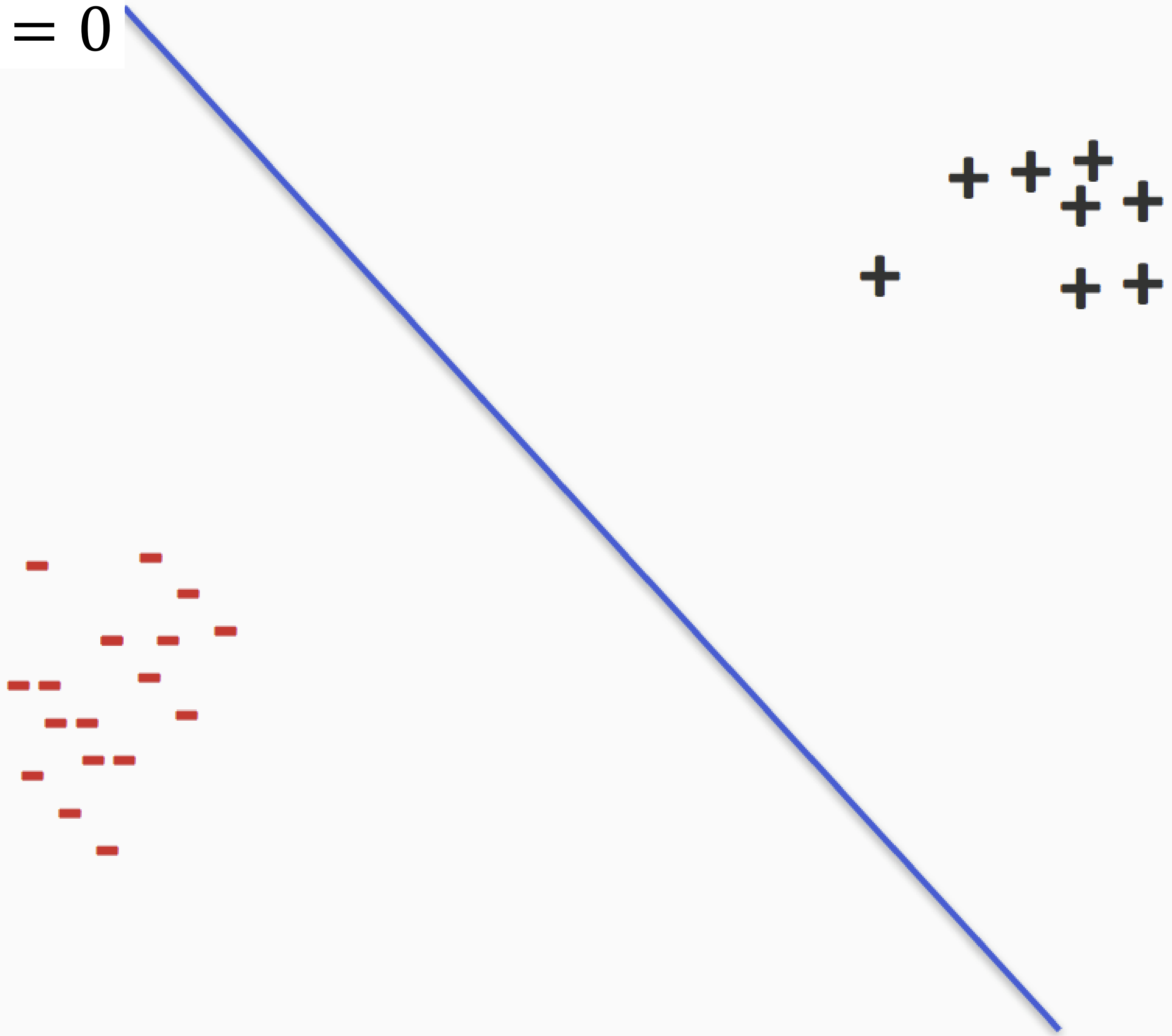
Outline

- Building blocks
- Neural Networks
- Expressivity of neural networks
- Predicting with neural networks

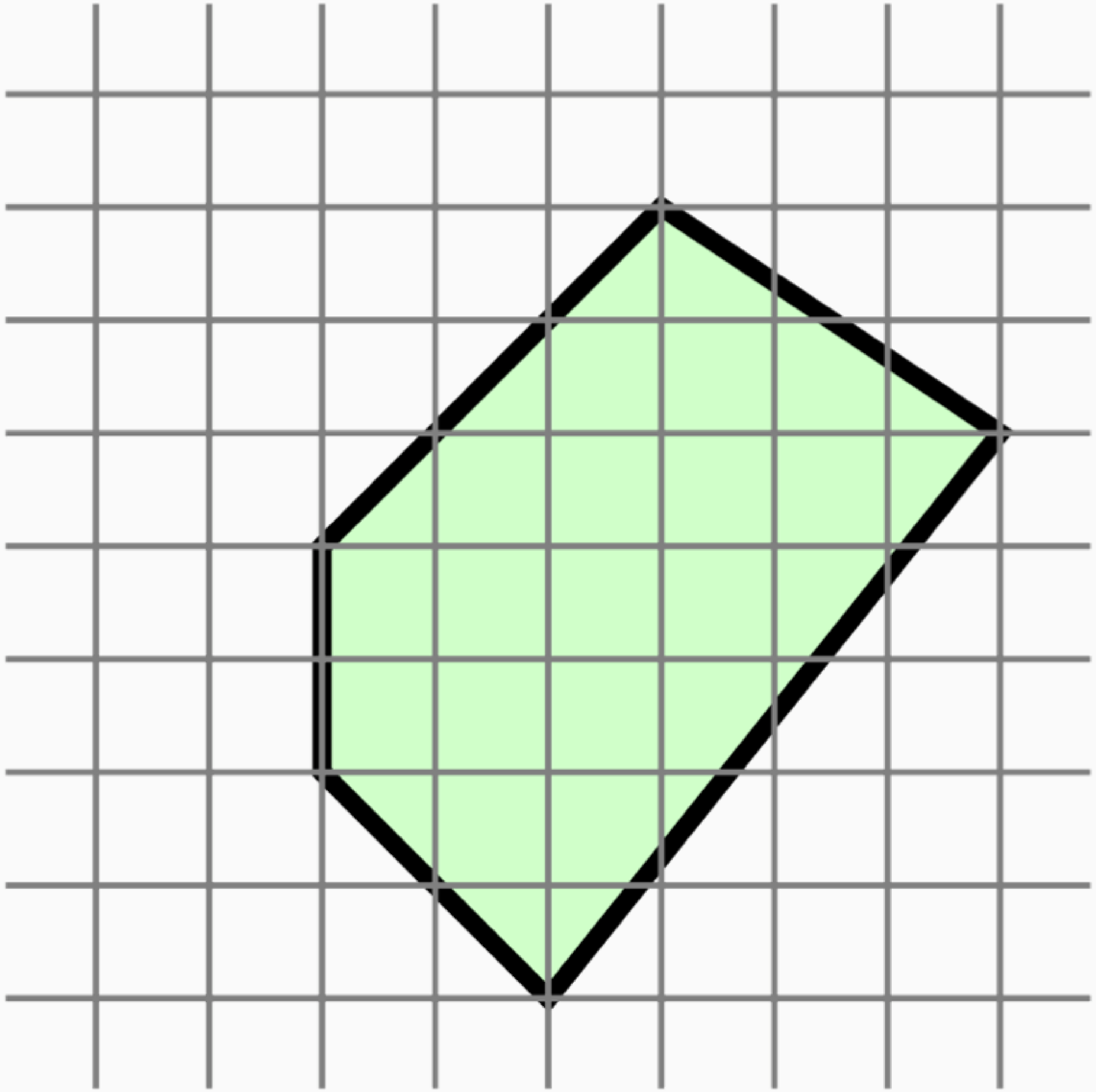
A single neuron with sign activation function

Prediction = $\mathrm{sgn}(w_1x_1 + w_2x_2 + w_0)$

$w_1x_1 + w_2x_2 + w_0 = 0$



Two layers with sign activation function



In general,
convex polygons

Figure from [Shai Shalev-Shwartz and Shai Ben-David, 2014]

Three layers with sign activation function

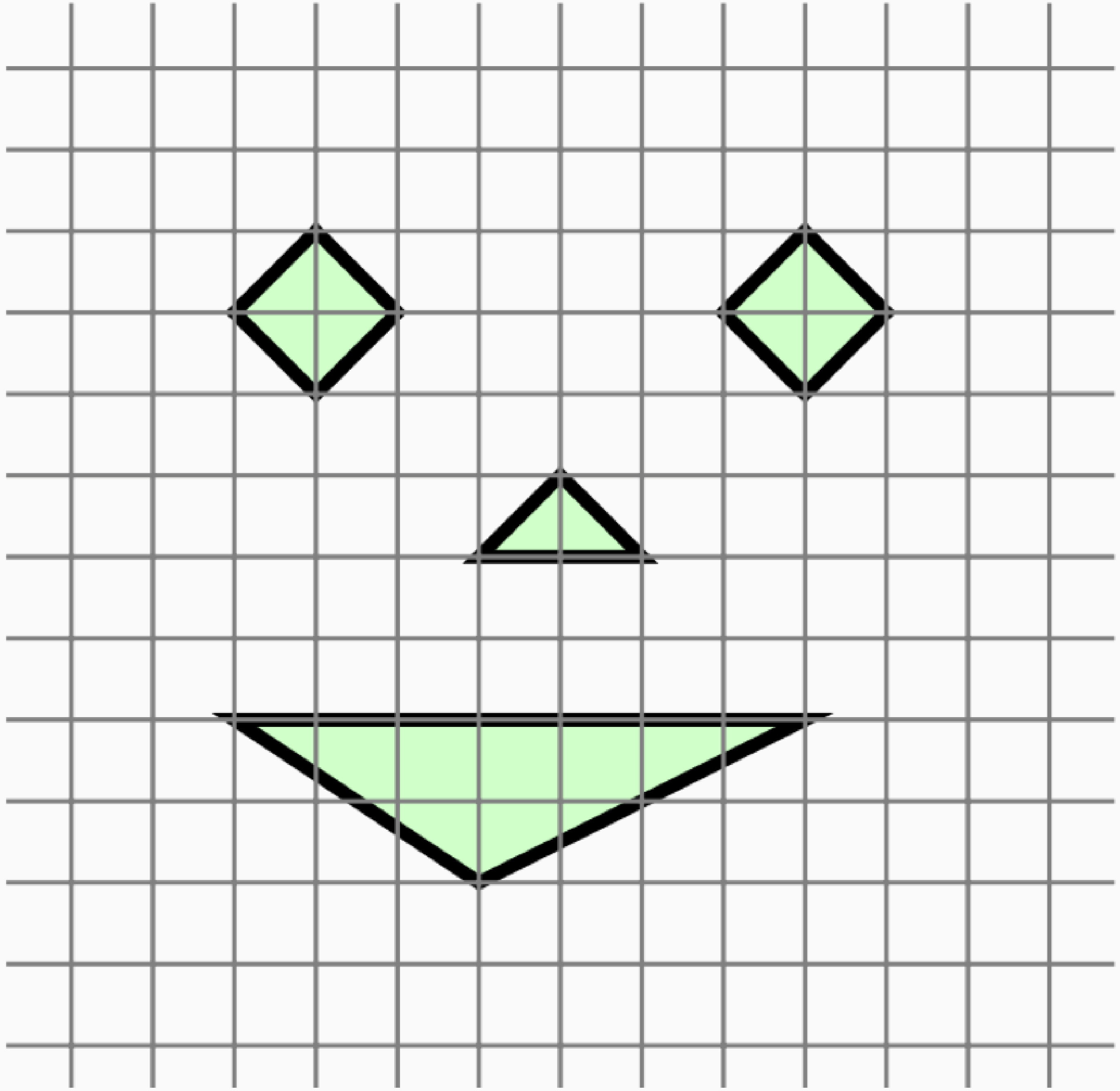


Figure from [Shai Shalev-Shwartz and Shai Ben-David, 2014]

In general, unions of convex polygons

What if I don't know?

https://playground.tensorflow.org/

NNs and universal function approximators

- Any continuous function can be approximated to arbitrary accuracy using one hidden layer of sigmoid units (Cybenko, 1989)

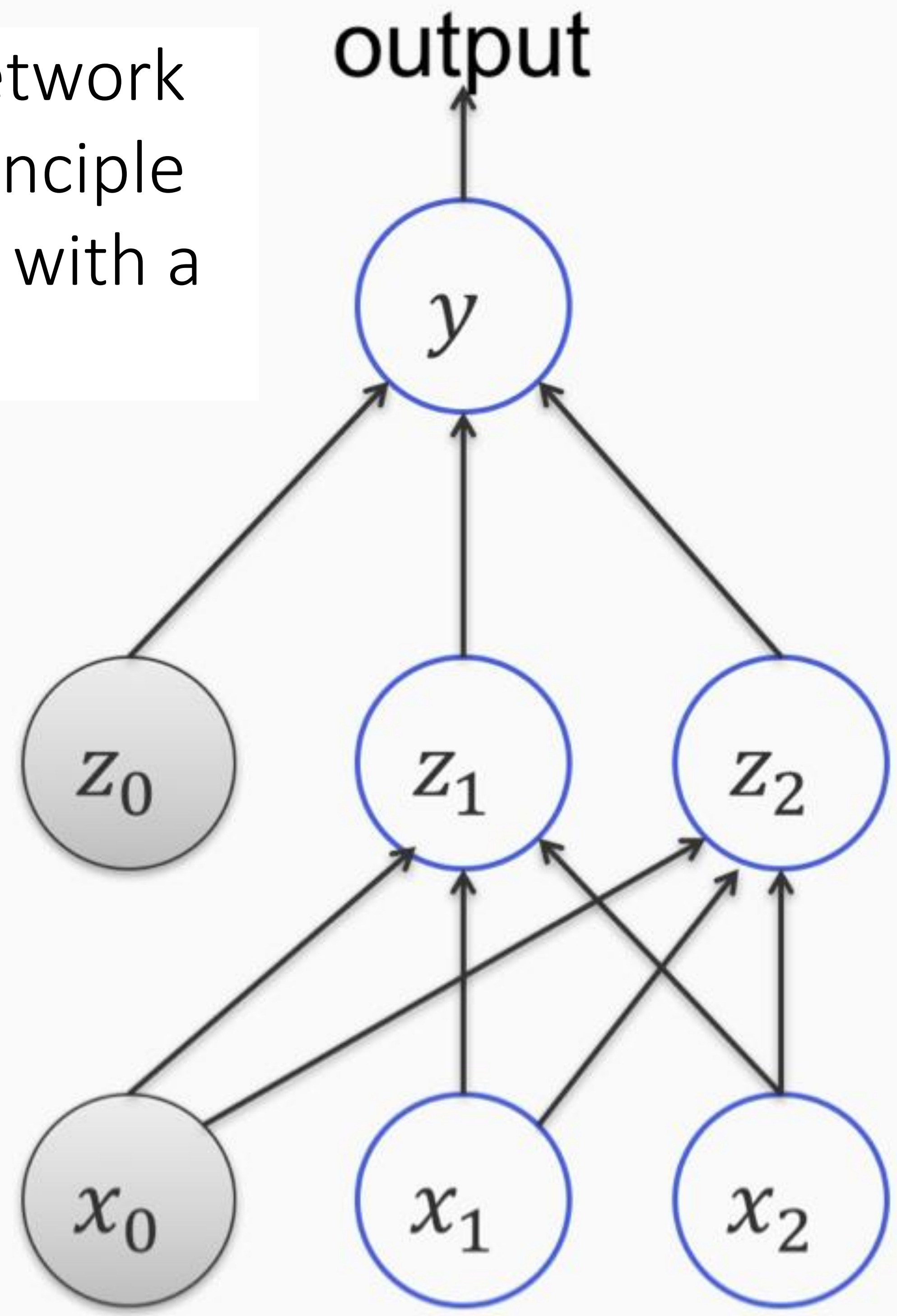
- Approximation error is insensitive to the choice of activation functions (DasGupta et al. 1993)

Outline

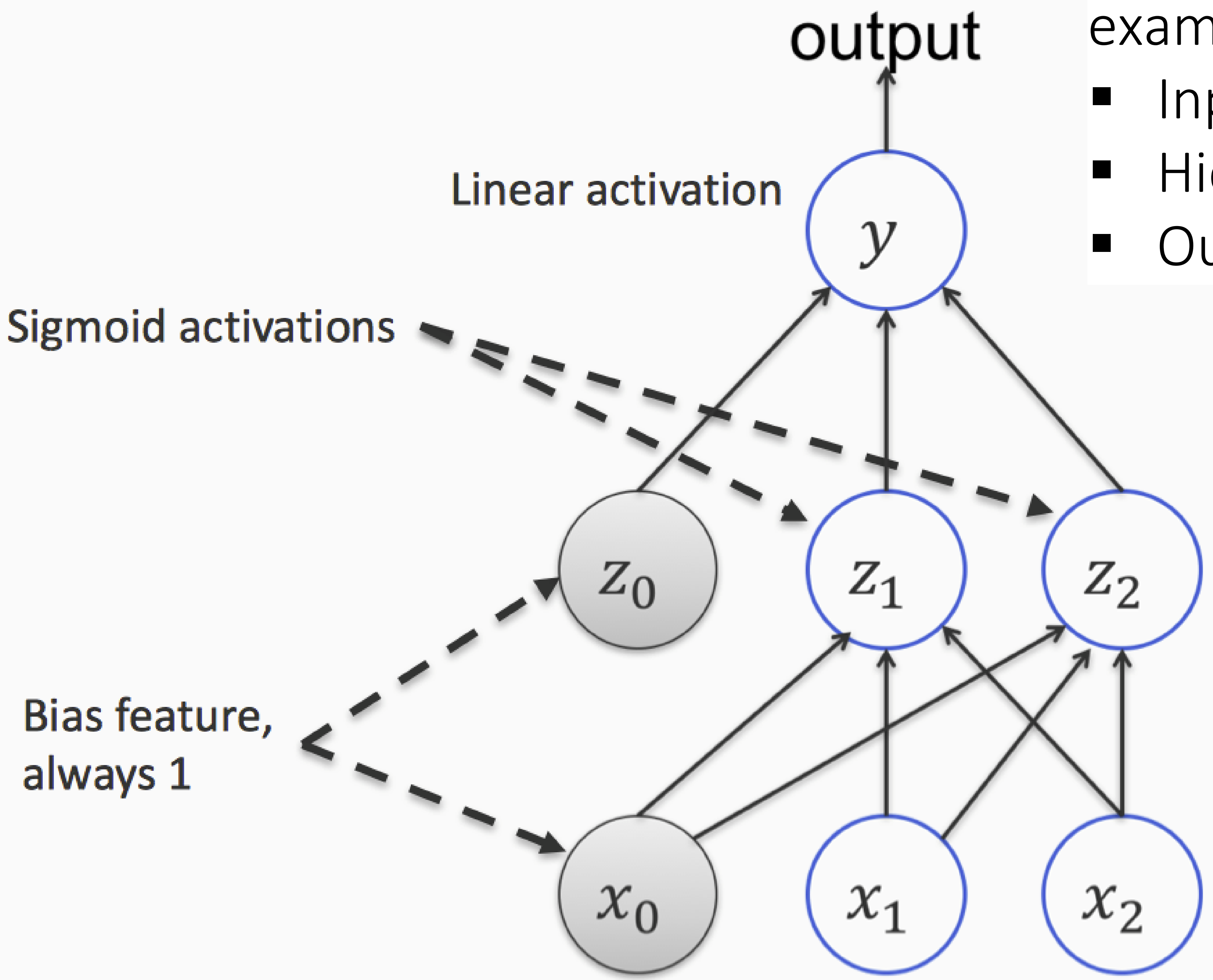
- Building blocks
- Neural networks
- Expressivity of neural networks
- Predicting with neural networks

Predicting with neural nets

We will use this example network to introduce the general principle of how to make predictions with a neural network



Predicting with neural nets



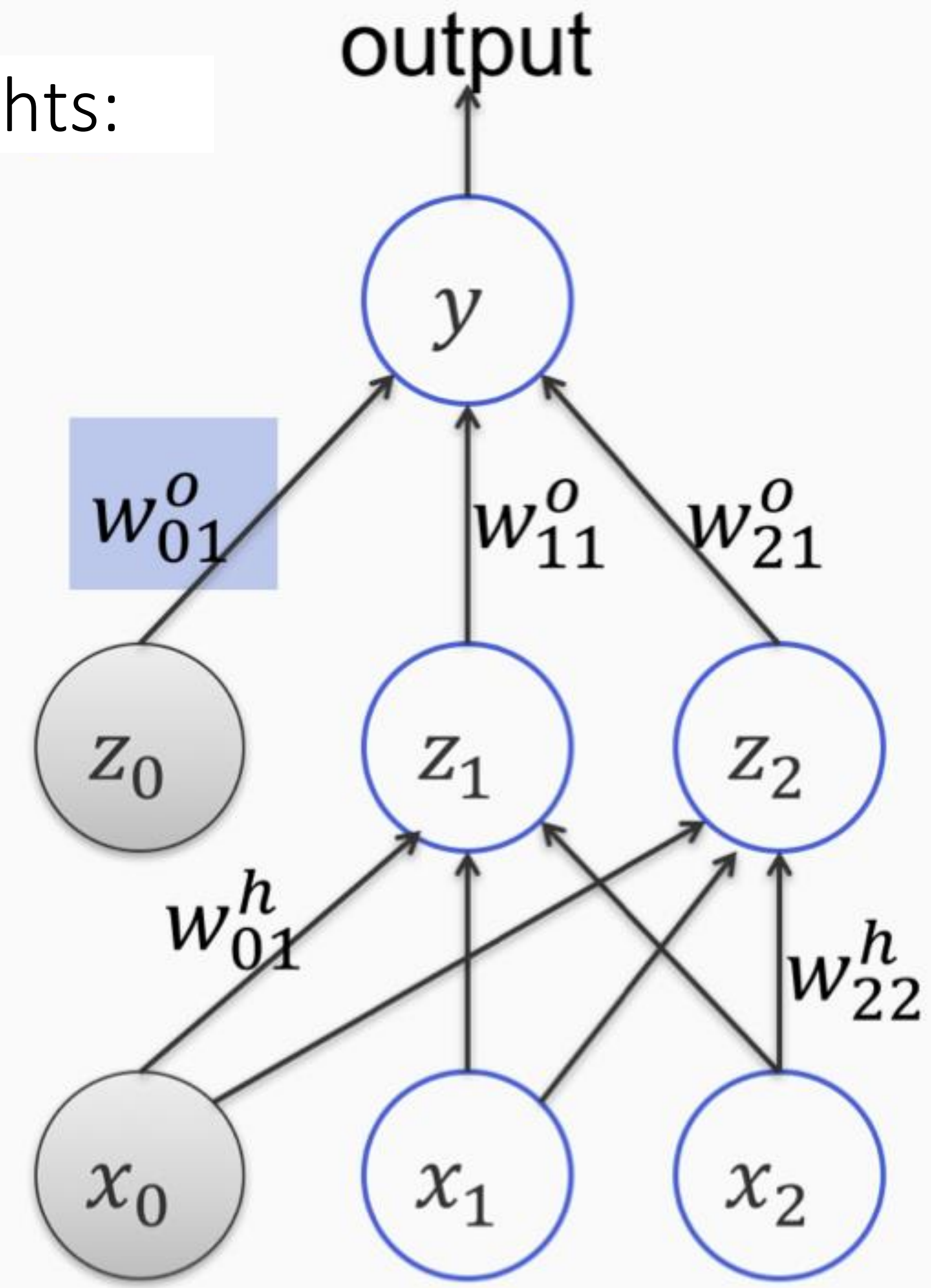
Naming conventions for this example

- Inputs: x
- Hidden: z
- Output: y

Predicting with neural nets

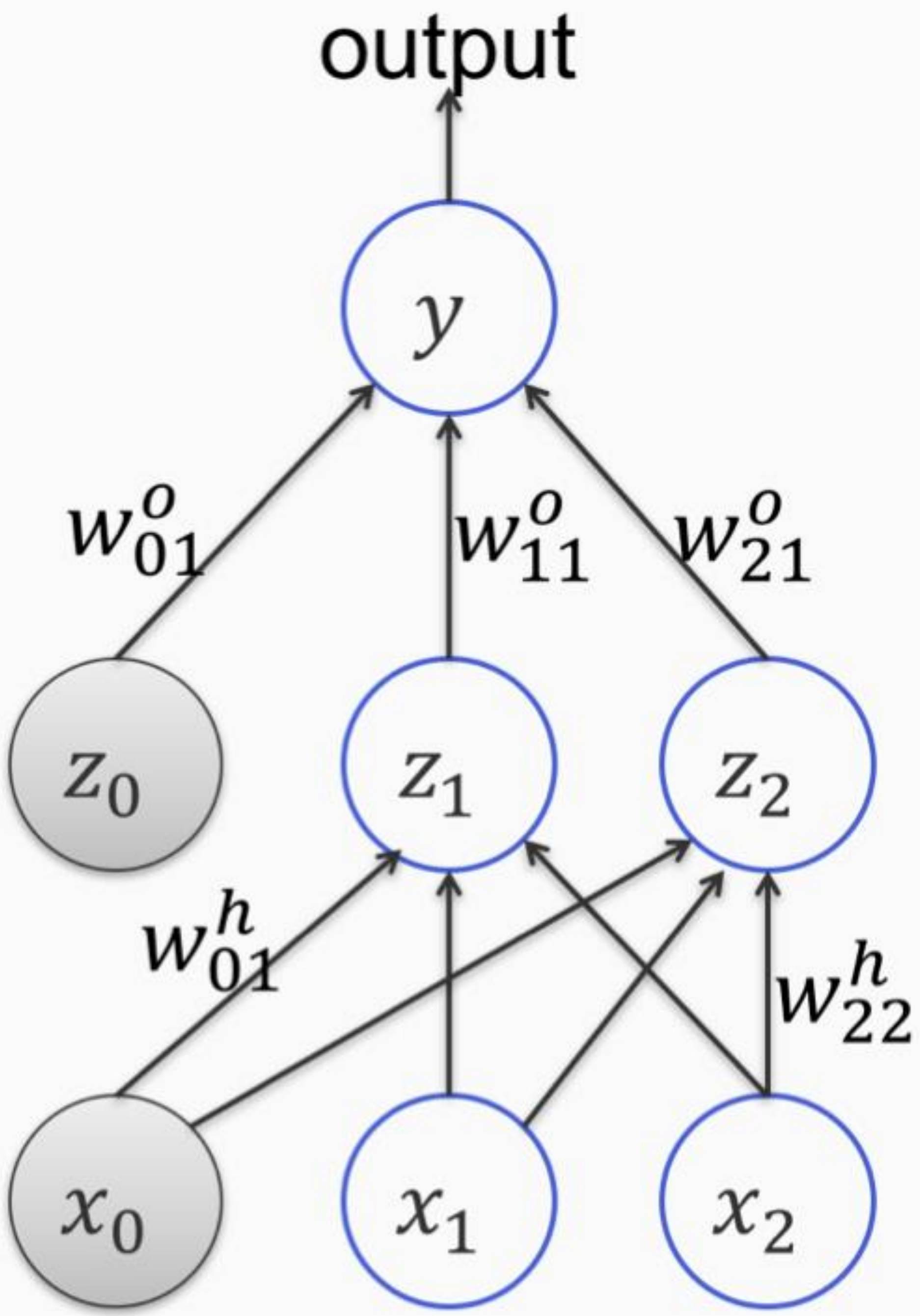
Naming conventions for weights:

$$w_{\text{from, to}}^{\text{layer}}$$



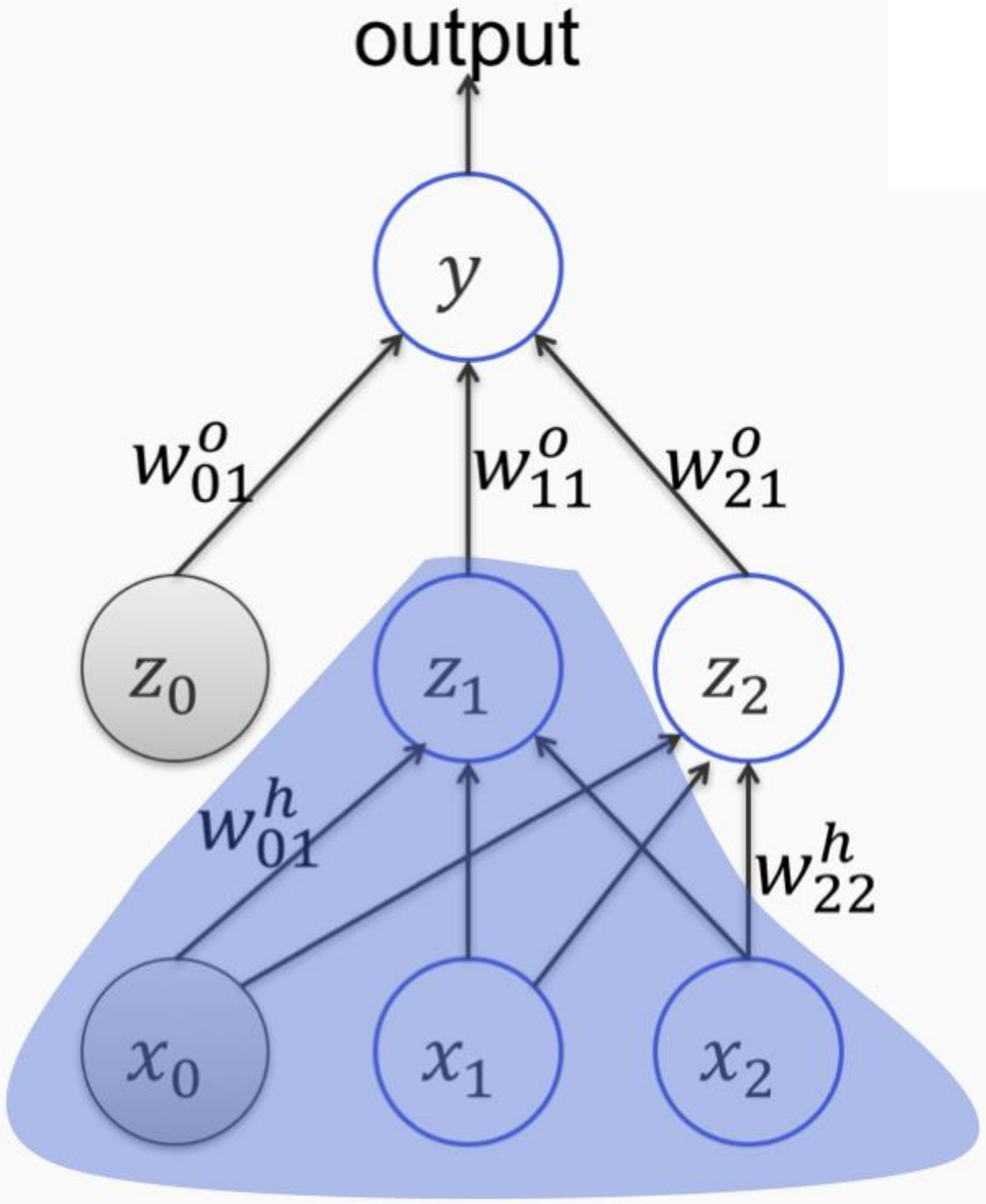
Example: $w_{0,1}^{o}$
From neuron 0
To neuron 1
In output layer

Predicting with neural nets: the forward pass



Given an input $\mathbf{x}$, how is the output predicted?

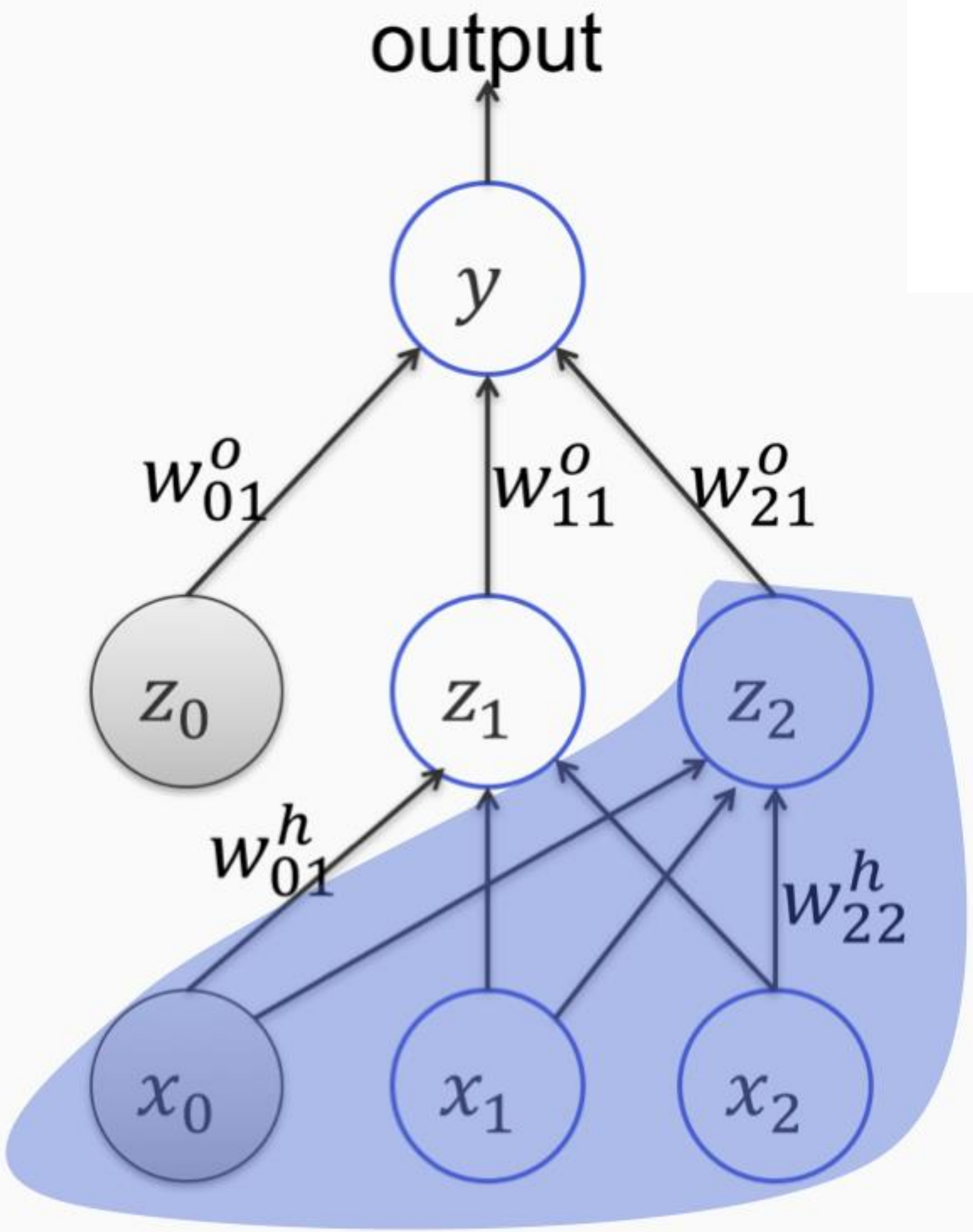
The forward pass



Given an input $\mathbf{x}$, how is the output predicted?

$$z_1 = h\left(w^h_{01} + w^h_{11}x_1 + w^h_{21}x_2\right)$$

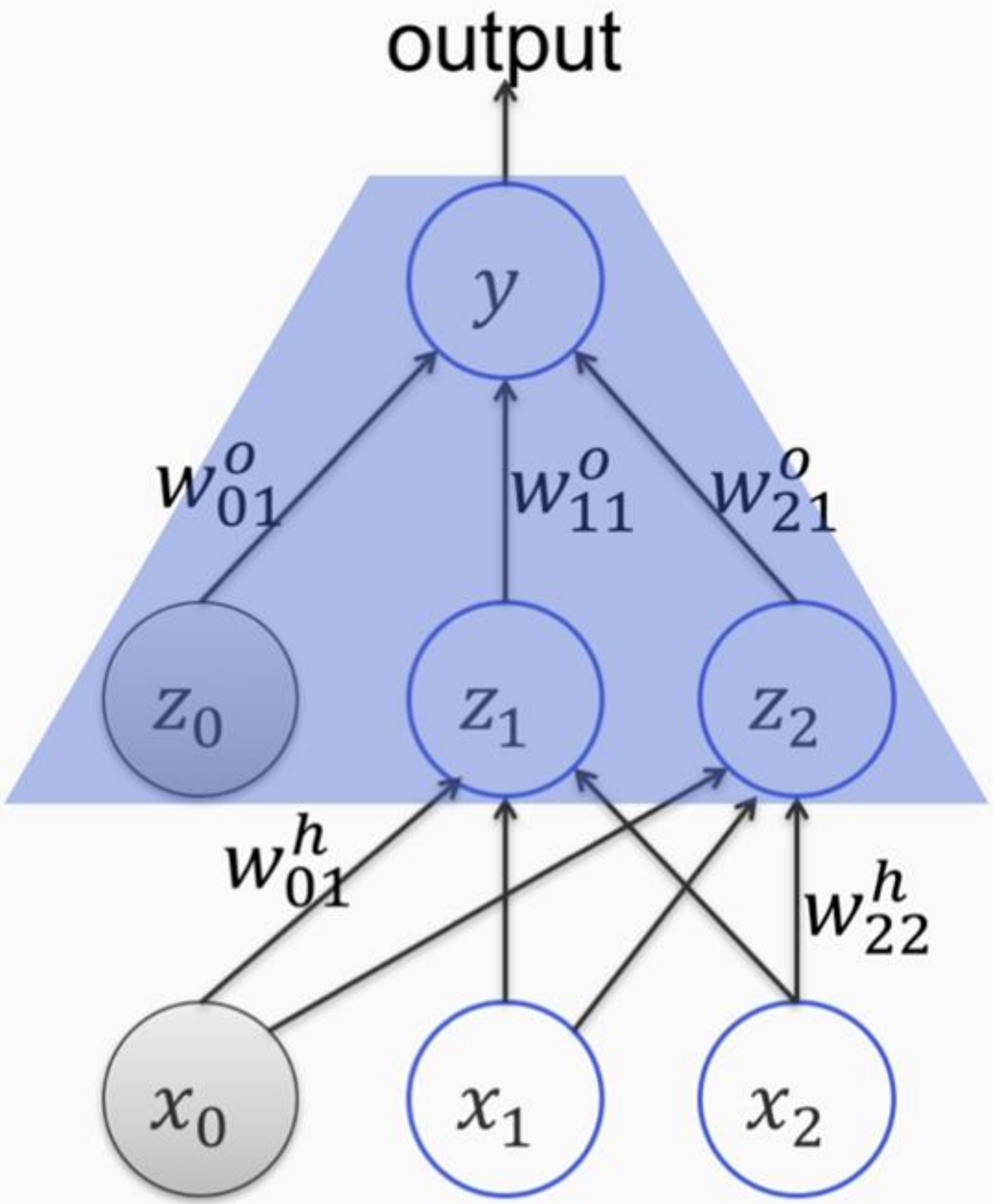
The forward pass



Given an input $\mathbf{x}$, how is the output predicted?

$$z_1 = h\left(w^h_{01} + w^h_{11}x_1 + w^h_{21}x_2\right)$$
$$z_2 = h\left(w^h_{02} + w^h_{12}x_1 + w^h_{22}x_2\right)$$

The forward pass



Given an input $\mathbf{x}$, how is the output predicted?

$$z_1 = h\left(w_{01}^{h} + w_{11}^{h}x_1 + w_{21}^{h}x_2\right)$$
$$z_2 = h\left(w_{02}^{h} + w_{12}^{h}x_1 + w_{22}^{h}x_2\right)$$

$$y = \sigma\left(w_{01}^{o} + w_{11}^{0}z_1 + w_{21}^{h}z_2\right)$$

Assuming a linear output activation function :

$$y = w_{01}^{o} + w_{11}^{0}z_1 + w_{21}^{h}z_2$$