Lecture 13: Neural Networks Intro

COMP 411, Fall 2021

Victoria Manfredi

WESLEYAN
UNIVERSITY

Acknowledgements: These slides are based primarily on content from the book “Machine Learning” by Tom Mitchell and slides created by Vivek Srikumar (Utah) and Dan Roth (Penn)

Today's Topics

No homework over the break!

Neural networks

- Structure, expressiveness
- Prediction using a neural network
- Training neural networks
- Practical concerns

vumanfredi@wesleyan.edu

Neural Networks

INFORMALLY

Where are we?

Learning algorithms

- Decision trees
- Linear regression
- Perceptron

Produce *linear* classifiers, regressors

General learning principles

- Overfitting
- Mistake-bound learning
- Training and generalization errors
- Regularized Empirical Loss Minimization

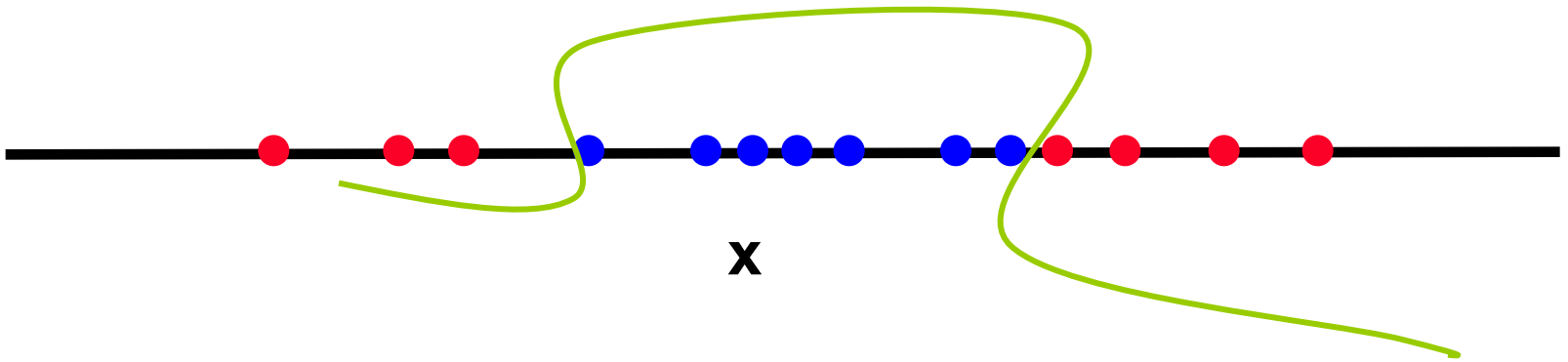
Not really resolved

- What if we want to train non-linear classifiers?
- Where do the features come from?

Functions Can be Made Linear

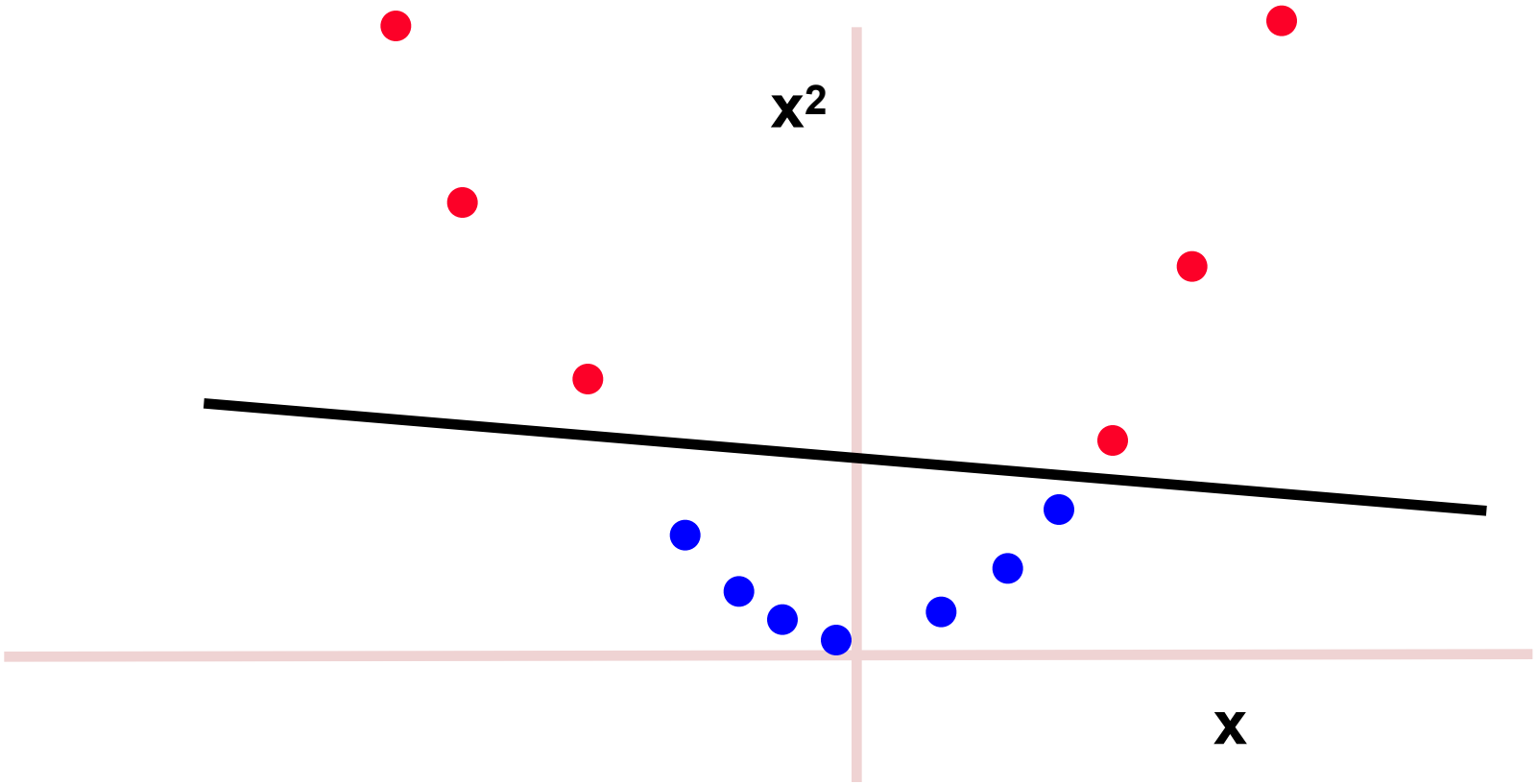
Data are not linearly separable in one dimension

- Not separable if you insist on using a specific class of functions



Blown Up Feature Space

Data are separable in <x, x^2> space



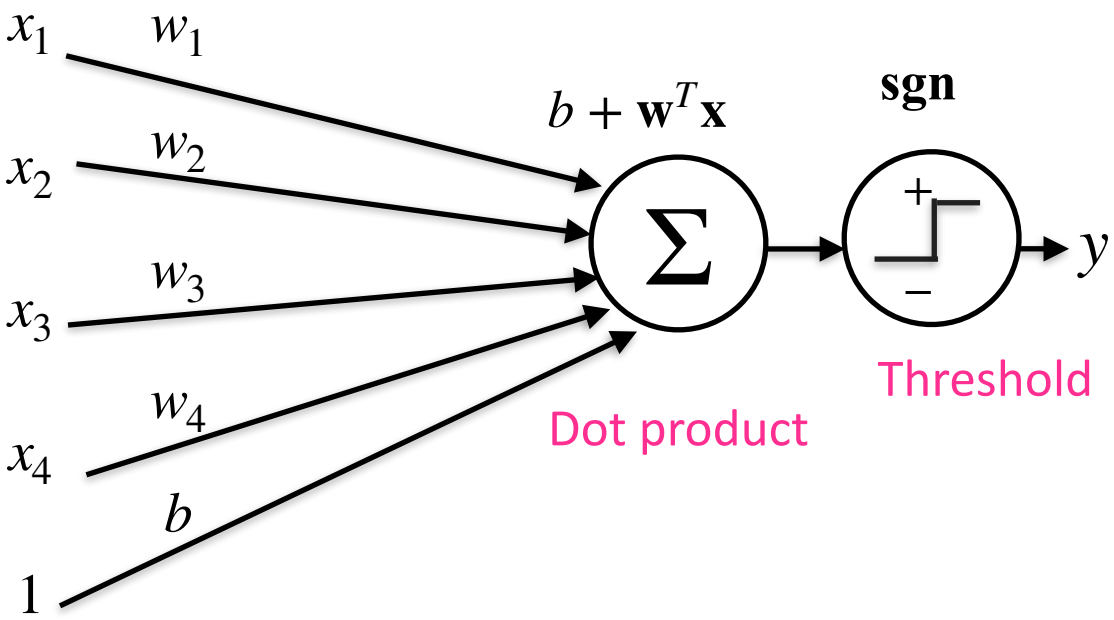
Non-linear ⇒ Neural networks

Linear separability depends on FEATURES!!

- A function can be not linearly separable with one set of feature but linearly separable in another

Have system to produce features, that make function linearly separable ⇒ Neural networks

We have seen linear threshold units



Features

Prediction

$$\text{sgn}(\mathbf{w}^T x + b) = \text{sgn}(\sum_i w_i x_i + b)$$

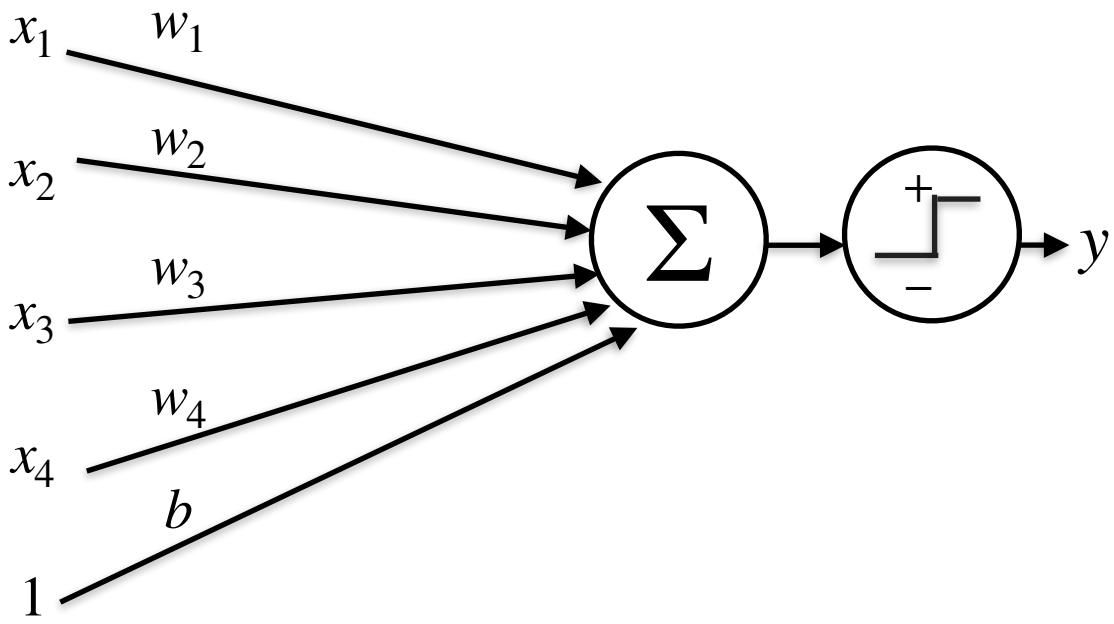
Learning

Various algorithms, in general, minimize loss

But where do these input features come from?

What if the features were outputs of another classifier?

Features from classifiers



Features from classifiers

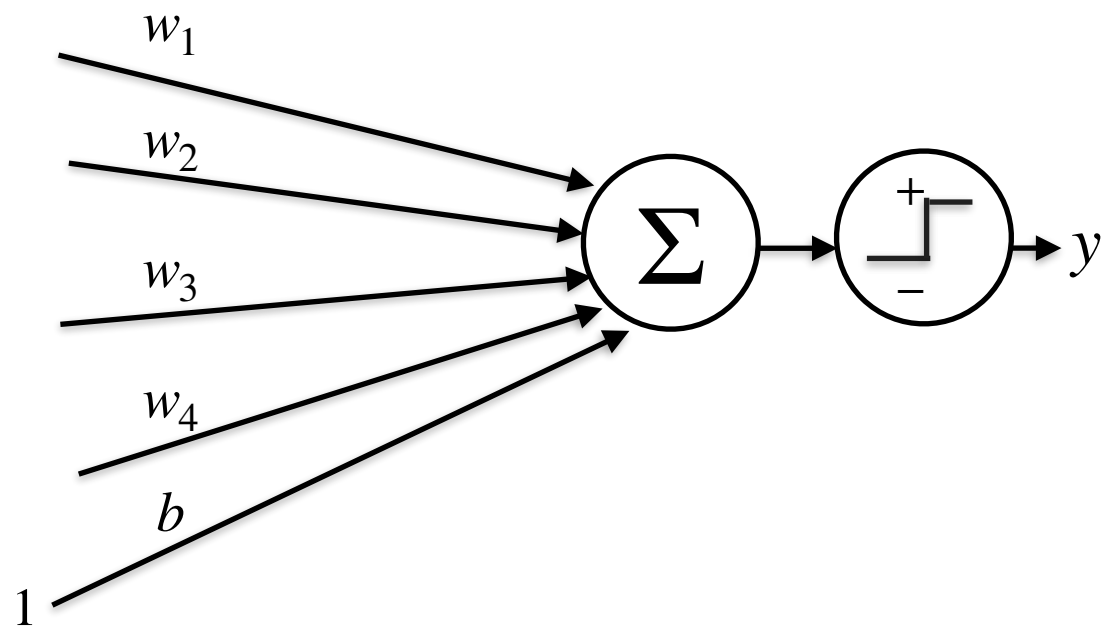
x_1

x_2

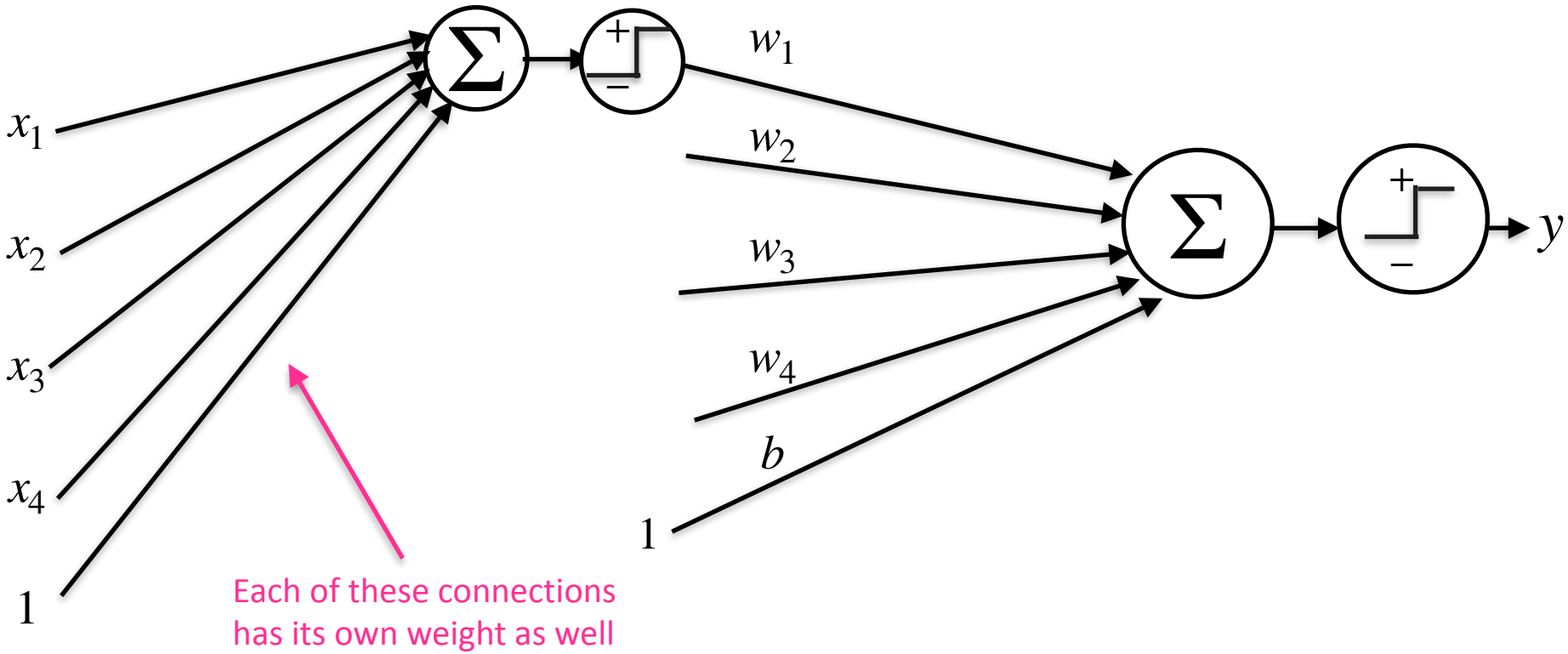
x_3

x_4

1

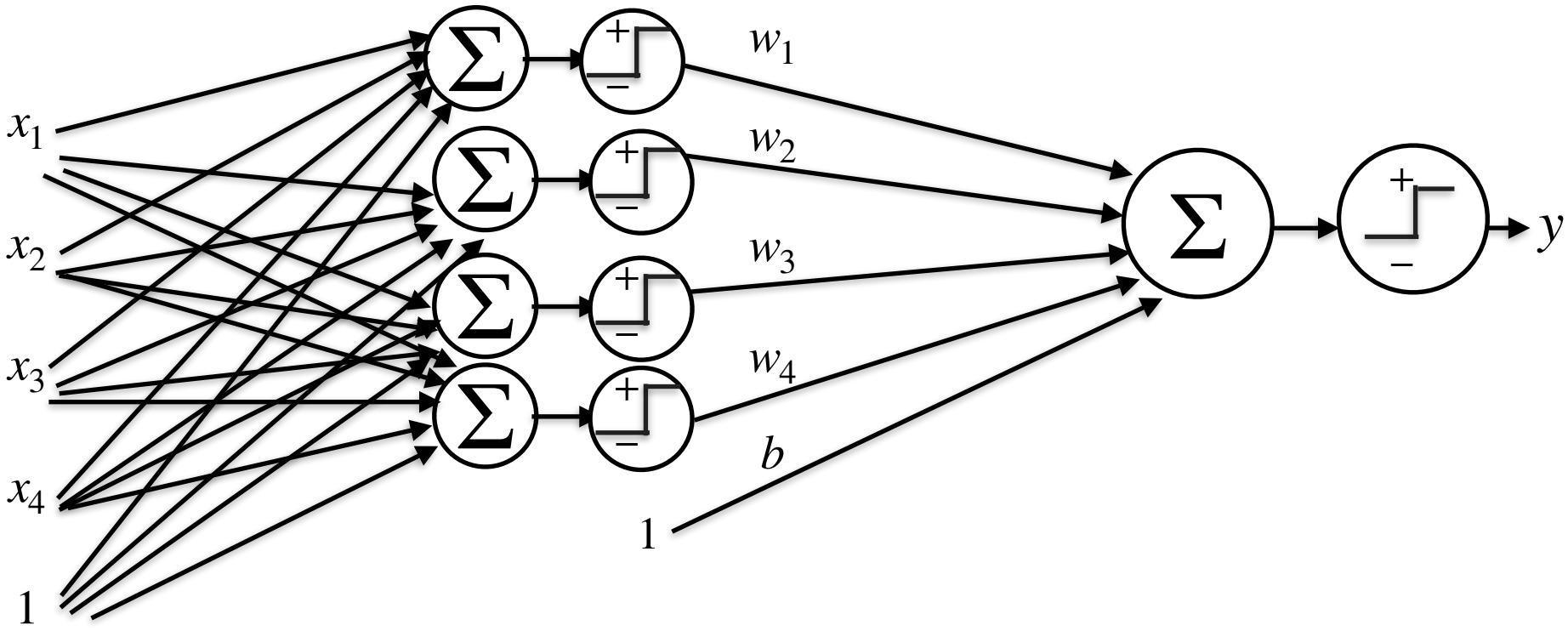


Features from classifiers



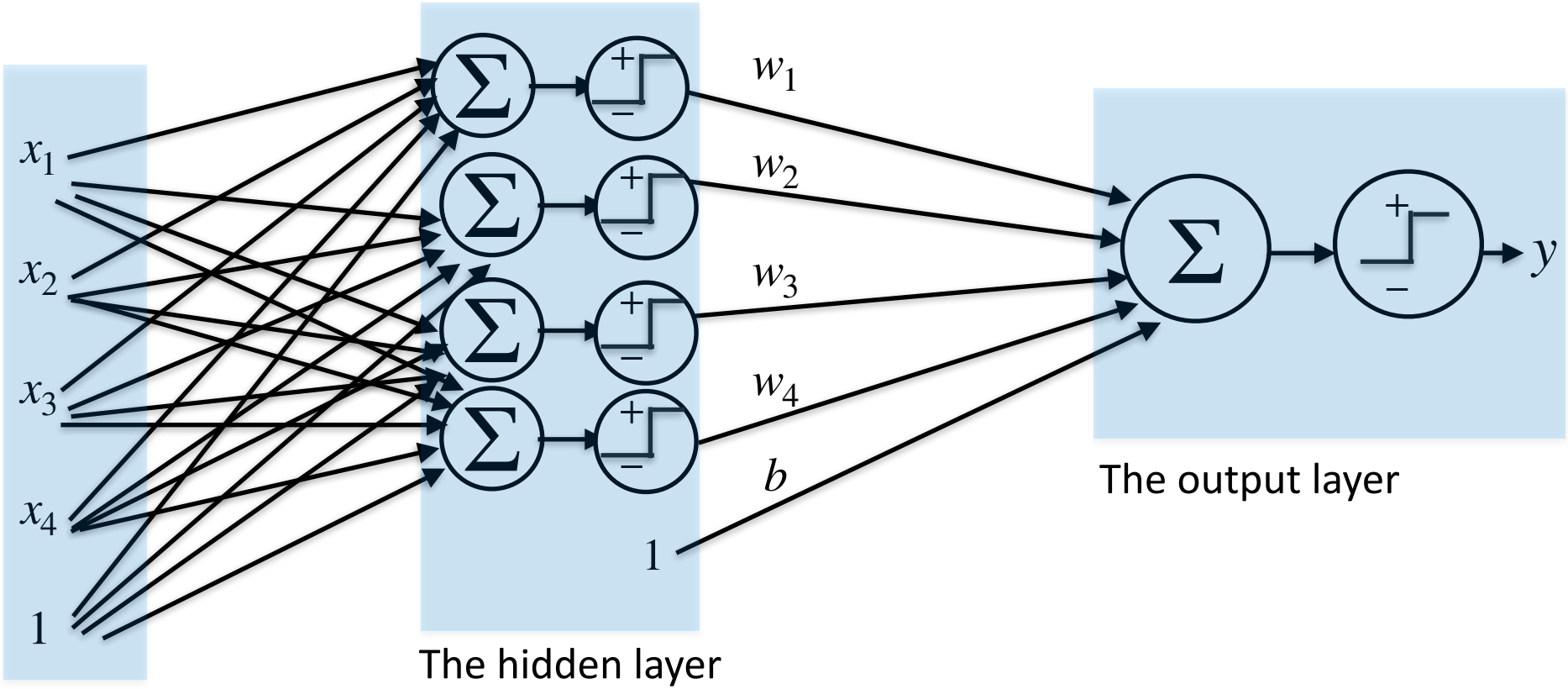
Features from classifiers

This is a two-layer feed forward neural network



Features from classifiers

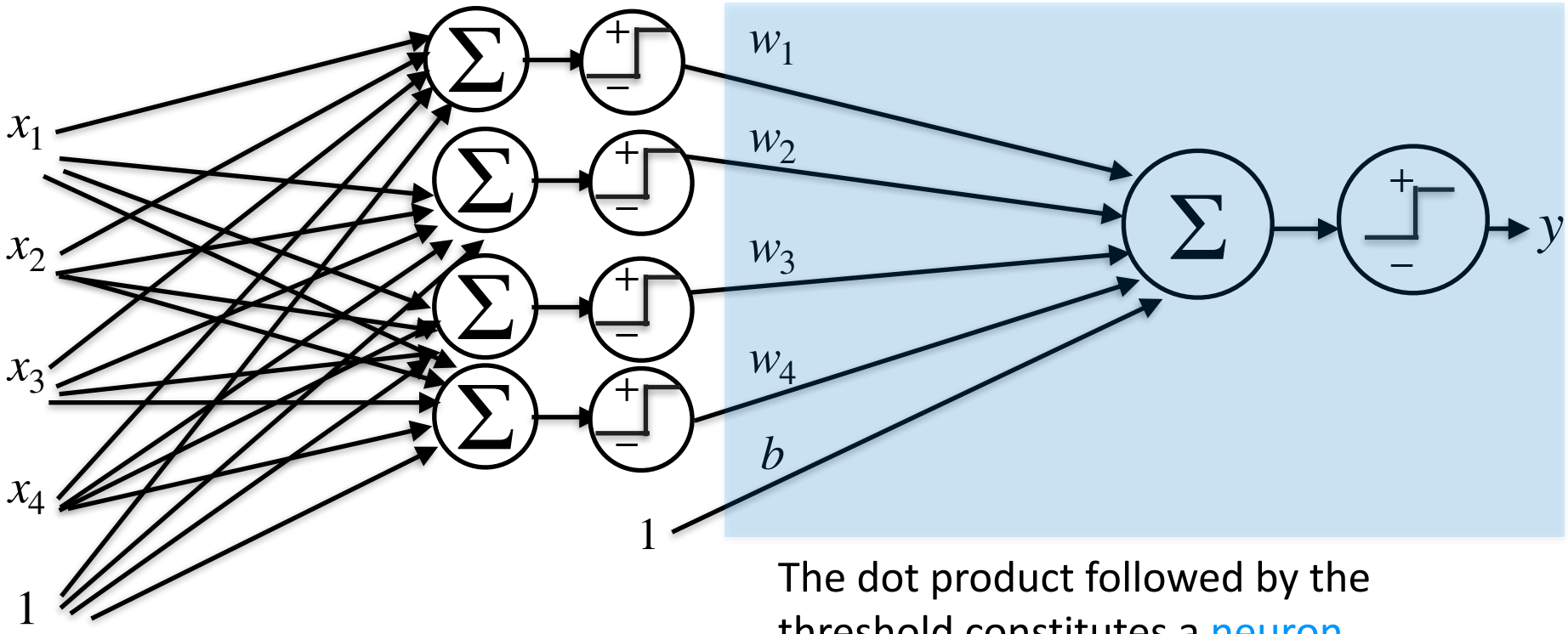
This is a two-layer feed forward neural network



Think of the hidden layer as learning a good representation of the inputs

Features from classifiers

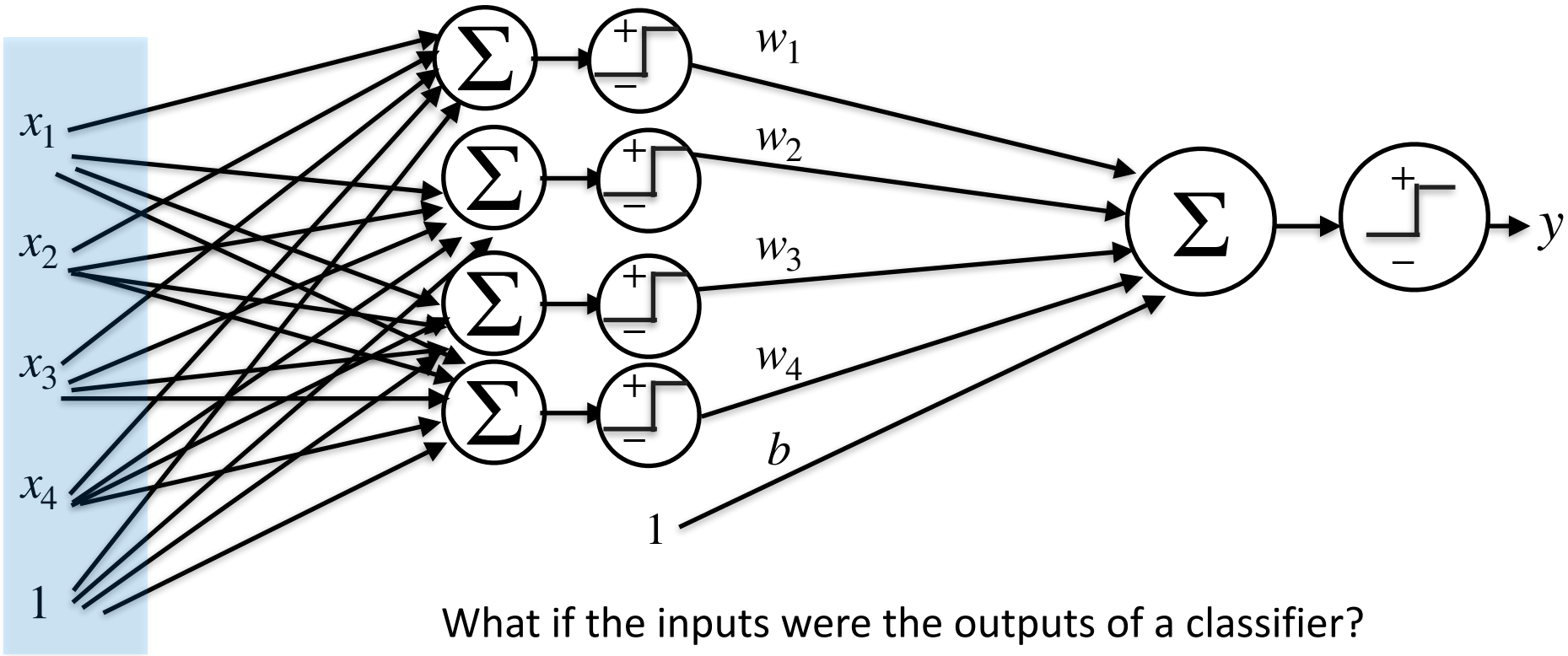
This is a two-layer feed forward neural network



The dot product followed by the threshold constitutes a neuron

Five neurons in this picture (four in hidden layer and one output)

But where do the inputs come from?



The input layer

What if the inputs were the outputs of a classifier?
We can make a three layer network … and so on.

Neural networks

Rich history, starting in early forties (McCulloch/Pitts 1943)

A robust approach for approximating real-valued, discrete-valued, or vector valued functions

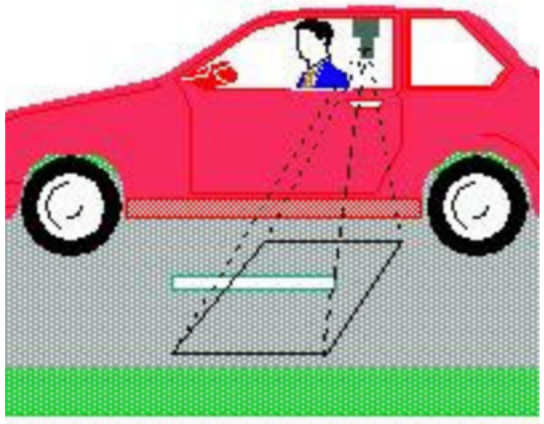
Among the most effective **general purpose** supervised learning methods currently known

- 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

- Across various application domains

Uses of neural networks



Trained to drive

- No-hands across America (Pomerleau)
- ARPA Challenge (Thrun)

Trained to recognize handwritten digits

- > 99% accuracy

Google Deepmind

- AlphaZero chess engine, AlphaGo for Go, …

Neural Networks

MORE FORMALLY

Why "Neural Network”

Brains – network of neurons – are only known example of actual intelligence

- Individual neurons are slow, boring
- Brains succeed by using massive parallelism

Idea

- use for building approximators!

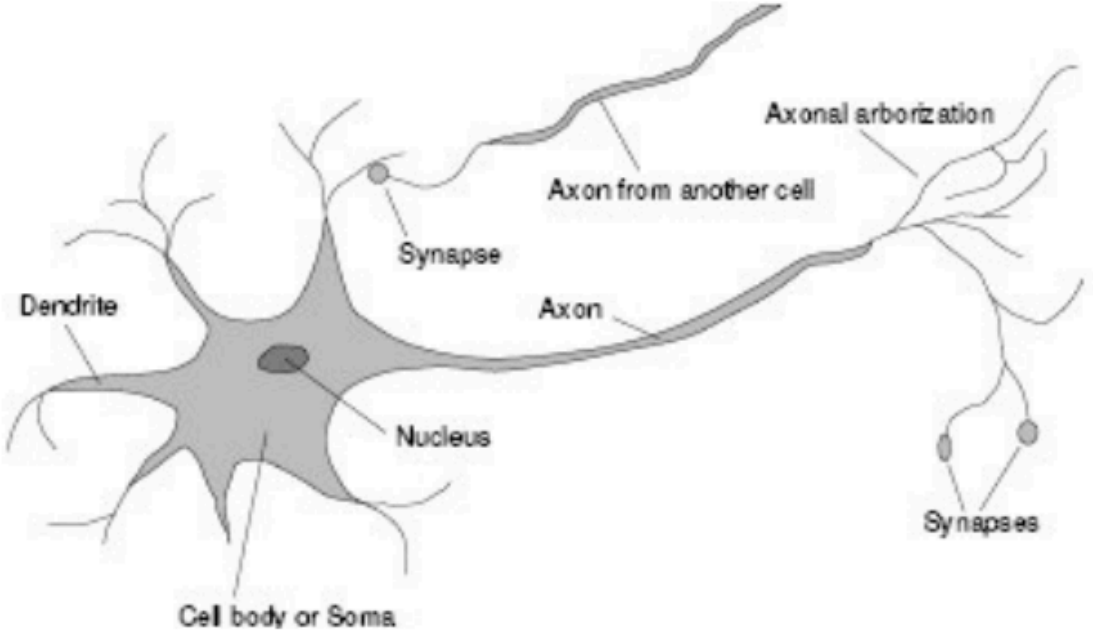
Raises many issues

- Is computational metaphor suited to computational hardware?
- How to copy the important part?
- Are we aiming too low?

Biological neurons

Neurons: core components of brain and the nervous system comprising

1. Dendrites that collect information from other neurons
2. An axon that generates outgoing spikes



Biological neurons

Neurons: core components of brain and the nervous system comprising

1. Dendrites that collect information from other neurons
2. An axon that generates outgoing spikes

Modern ***artificial*** neurons are "inspired" by biological neurons … but there are many, many fundamental differences

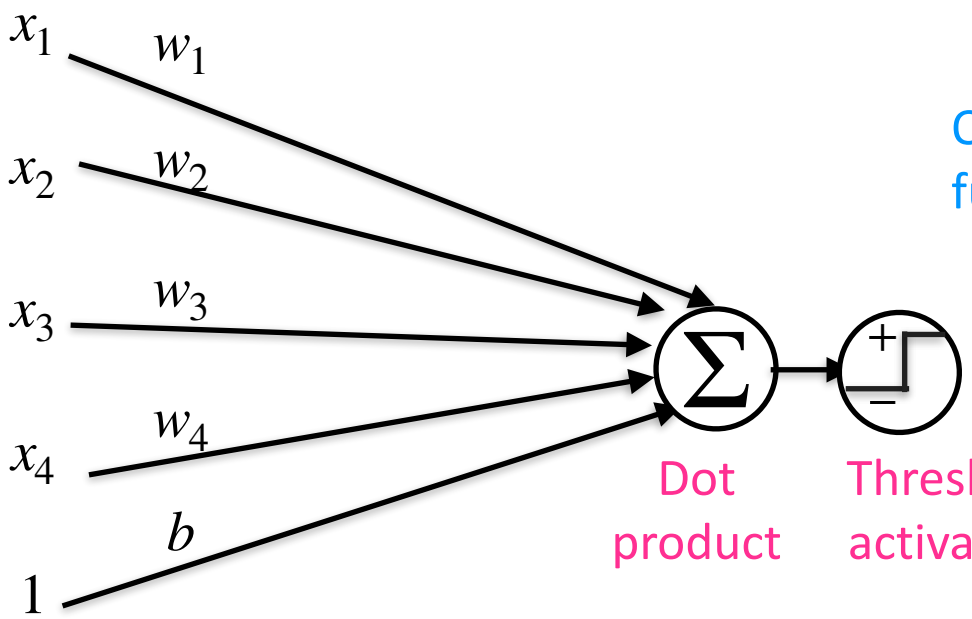
Artificial neurons

Functions that ***very loosely*** mimic a biological neuron

A neuron accepts a collection of inputs (a vector $\mathbf{x}$) and produces an output by:

1. Applying a dot product with weights $\mathbf{w}$ and adding a bias b
2. Applying a (possibly non-linear) transformation called an activation

$$output = \text{activation}(\mathbf{w}^T\mathbf{x} + b)$$



Other activation functions are possible

Common activation functions

Also called transfer functions

$$output = \text{activation}(\mathbf{w}^T\mathbf{x} + b)$$

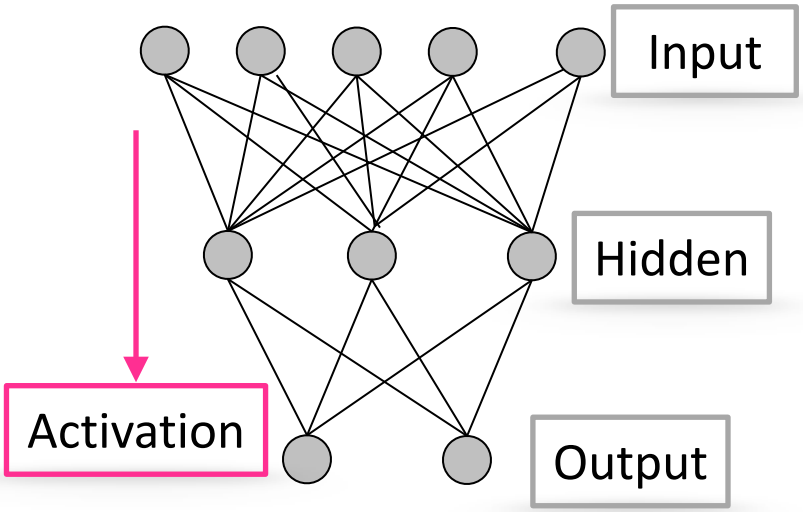
Name of the neuron	Activation function: activation(*z*)
Linear unit	*z*: (i.e., no change to the input)
Threshold/sign unit	sgn(*z*)
Sigmoid unit	1 / (1 + exp(-*z*))
Rectified linear unit (ReLU)	max(0, *z*)
Tangent hyperbolic (Tanh) unit	tanh(*z*)

Multi-layer neural network

Designed to overcome the computational (expressivity) limitation of a single threshold element

Idea

- stack several layers of units
- each layer uses output of previous layer as input
- can represent arbitrary functions

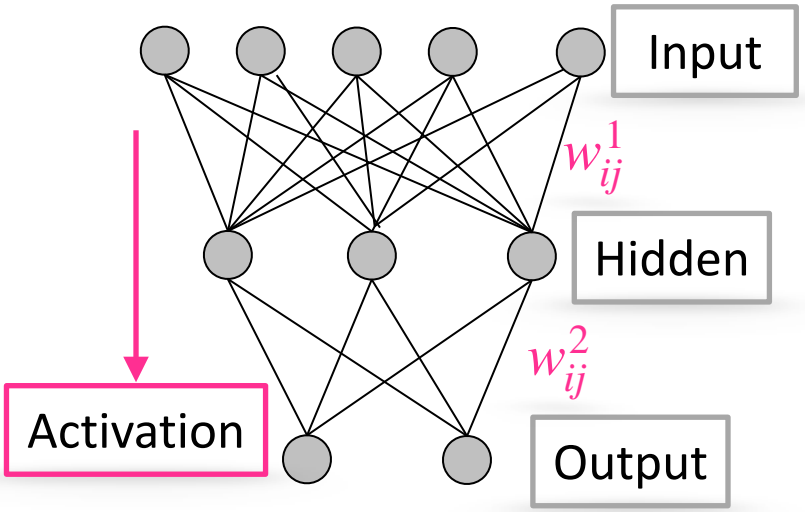


A neural network is a function that converts inputs to outputs defined by a directed acyclic graph

Basic Units

Linear Unit

- Multiple layers of linear functions $o_j = \mathbf{w} \cdot \mathbf{x}$ produce linear functions



We want to represent nonlinear functions

- but need to do in a way that facilitates learning

Threshold units: $o_j = \text{sgn}(\mathbf{w} \cdot \mathbf{x})$

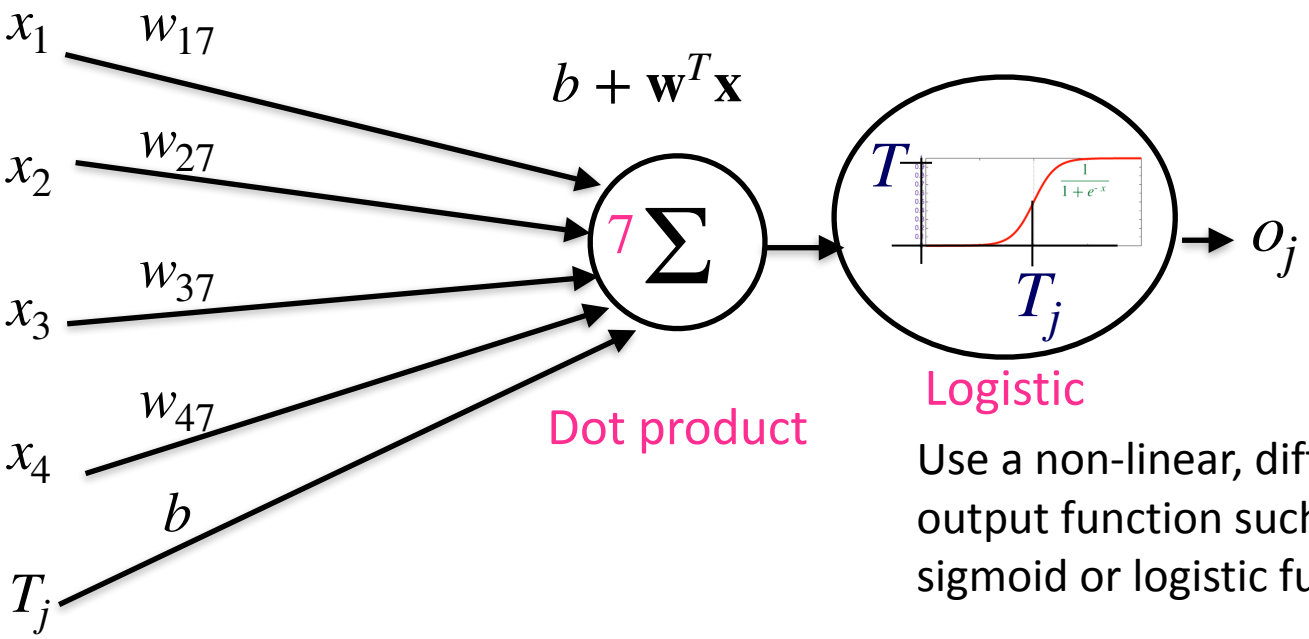
- are not differentiable, hence unsuitable for gradient descent.

Key idea

- notice that the discontinuity of the threshold element can be represents by a smooth non-linear approximation: $o_j = \dfrac{1}{[1 + \exp\{-\mathbf{w} \cdot \mathbf{x}\}]}$ (Rumelhart, Hinton, Williiam, 1986), (Linnainmaa, 1970) , see: http://people.idsia.ch/~juergen/who-invented-backpropagation.html)

Model neuron (logistic)

Neuron is modeled by a unit j connected by weighted links w_{ij} to other units i



Use a non-linear, differentiable output function such as the sigmoid or logistic function

Net input to a unit is defined as $\text{net}_j = \sum w_{ij} \cdot x_i$

Output of a unit is defined as $o_j = \dfrac{1}{1 + e^{-(\text{net}_j - T_j)}}$

To define a neural network

Specify:

- Structure of the graph: how many nodes, the connectivity
- The activation function on each node
- The edge weights

Learned from data

Architecture of the network. Typically predefined, part of the design of the classifier

A brief history of neural nets

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

1970s, 80s: Convolutional neural networks (Fukushima, LeCun), the back propagation algorithm (various)

Early 2000s-today: More compute, more data, deeper networks

See also https://people.idsia.ch//~juergen/deep-learning-overview.html