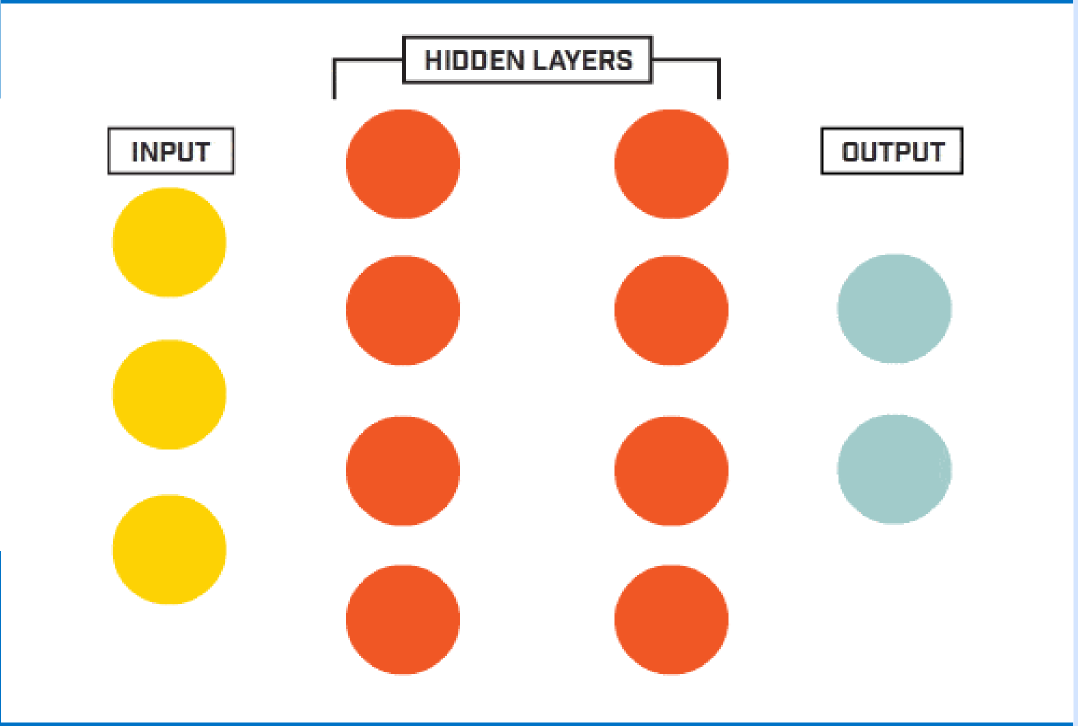
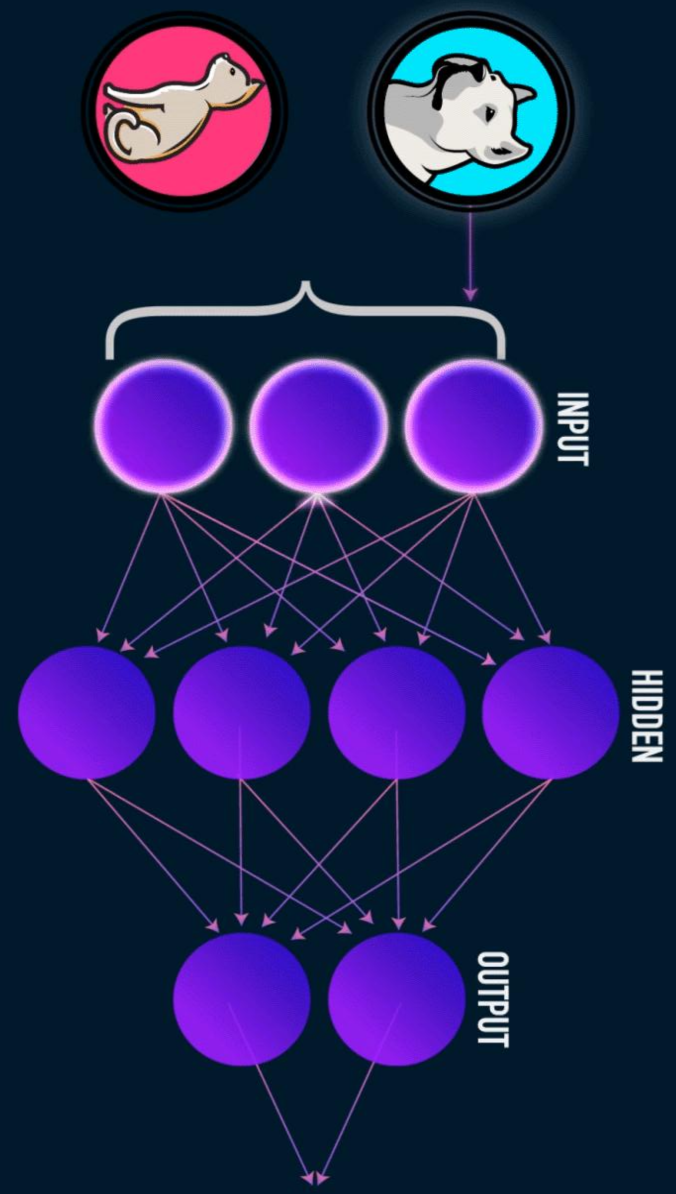


Introduction to Deep Learning (Intuitive Understanding)



Introduction to Deep Learning (Intuitive Understanding)

Outline

Part I: Introduction to Deep Learning

Part II: Tips for Training Deep Neural Network

Part III: Algorithms – CNN & Autoencoders

Part IV: Algorithms – RNN, LSTM

Deep Learning .Vs. Traditional Computing

Recognize images of cats – Traditional Method

IF (furry) AND
IF (has tail) AND
IF (has 4 legs) AND
IF (has pointy ears) AND
Etc…

- explicitly programming the computer for what features to look for.
- However, it's pretty easy to see that **these features could also apply to a dog or a fox,**
- it would be tricky to account for unusual edge cases like a **cat without a tail or only three legs.**

Deep Learning .vs. Traditional Computing

Recognize images of cats – Neural Networks

show the computer a load of examples of cats

show the computer a load of examples of not-cats

Deep Learning .vs. Traditional Computing

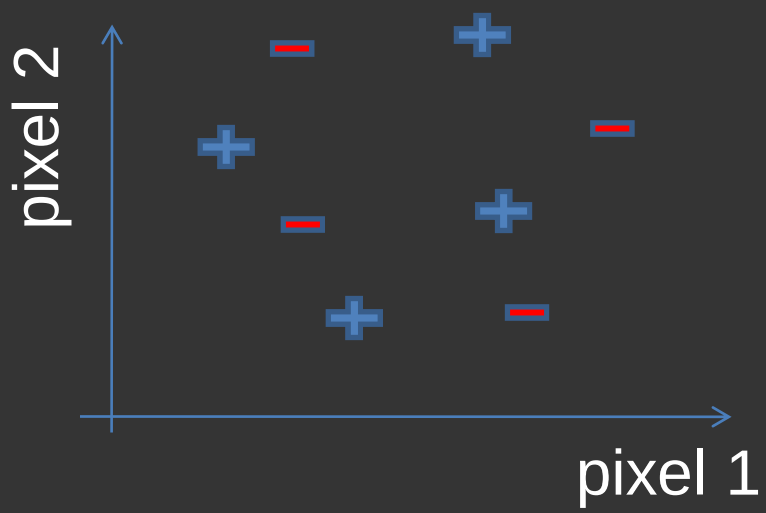
Recognize images of cats – Neural Networks

- Here, the computer works out:
  - what to look for, and
  - **what features are essential to 'cat-ism'.**
- This is actually a lot closer to the way humans learn to distinguish objects, and is why we call them 'neural' networks'
  - they are inspired by biology and how our own brains work.

What is Machine Learning?

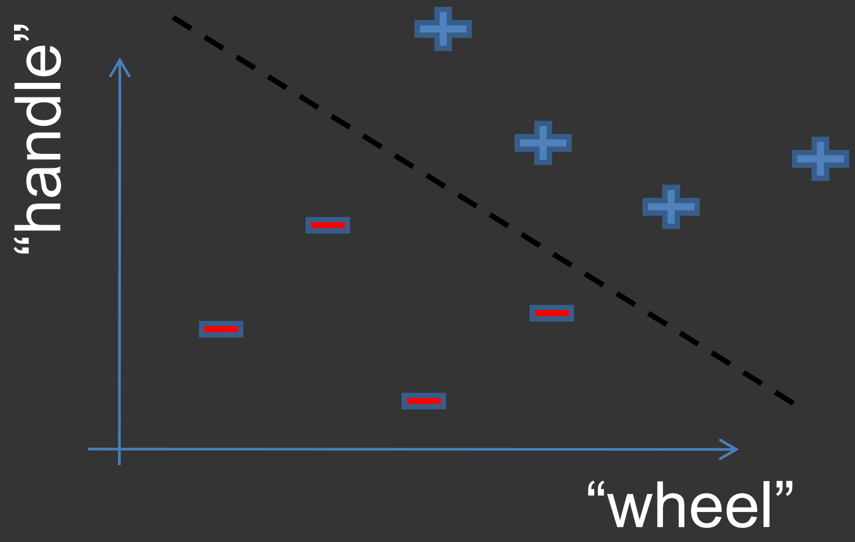
Input → Feature Extraction → Machine Learning algorithm

Input space



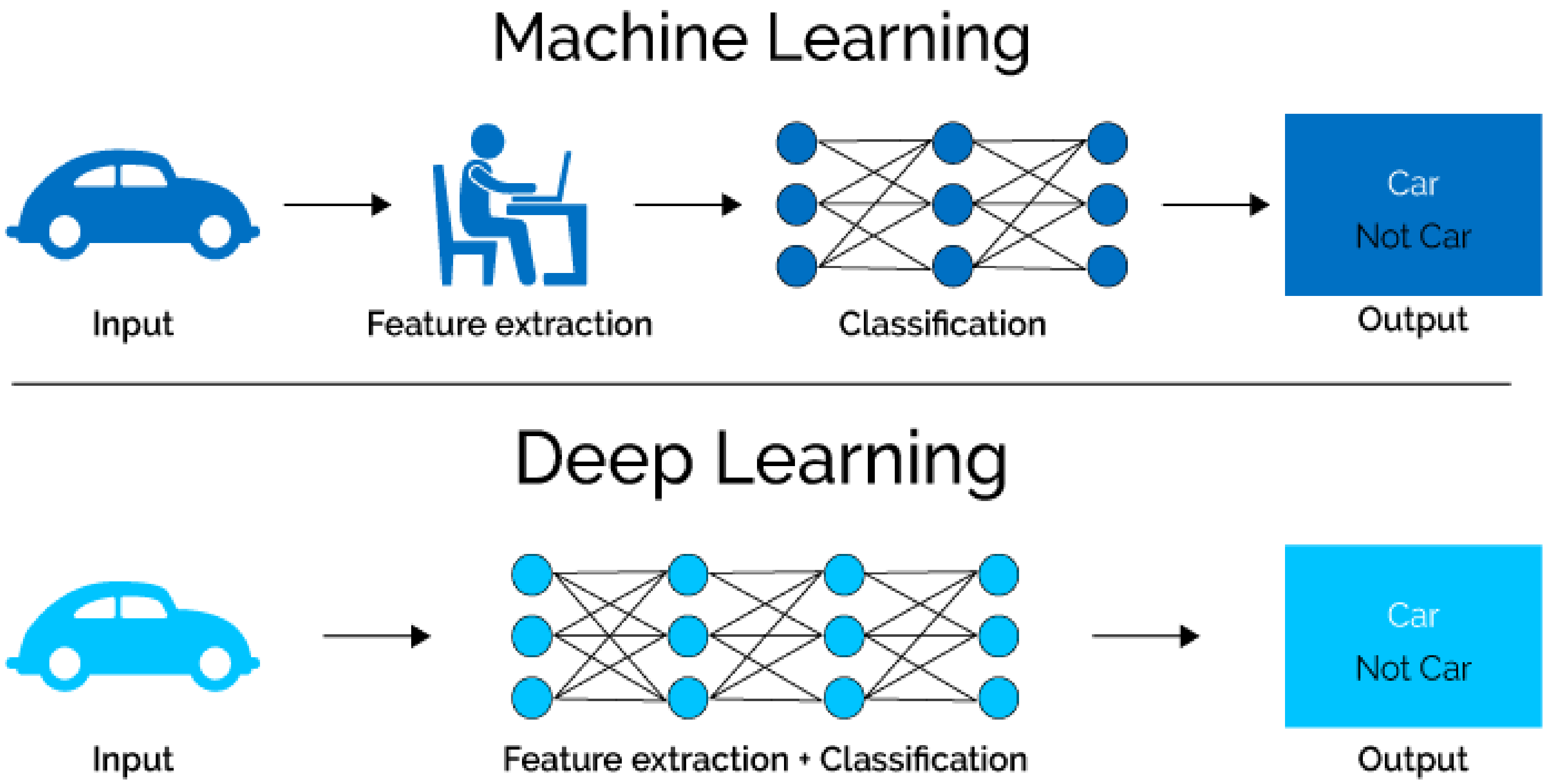
+ Motorbikes
− “Non”-Motorbikes

Feature space



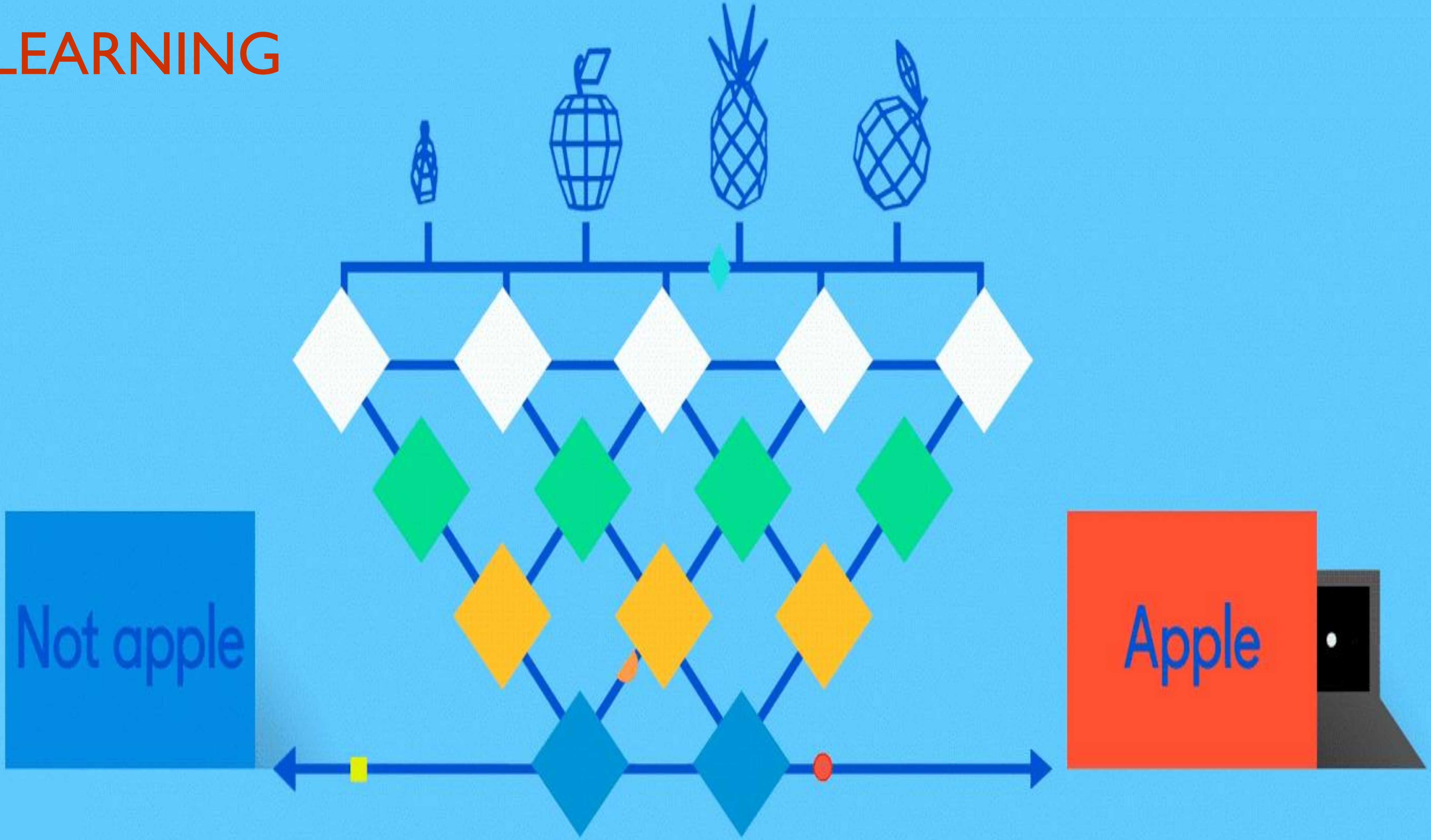
Machine learning .vs. deep learning

- A machine learning requires **representations** of data to learn and predict.
- Deep learning algorithms attempt to learn (multiple levels of) representation on its own by using a **hierarchy of multiple layers**
- If you provide the system **tons of information** it begins to understand it and respond in useful ways.



https://www.xenonstack.com/blog/static/public/uploads/media/machine-learning-vs-deep-learning.png

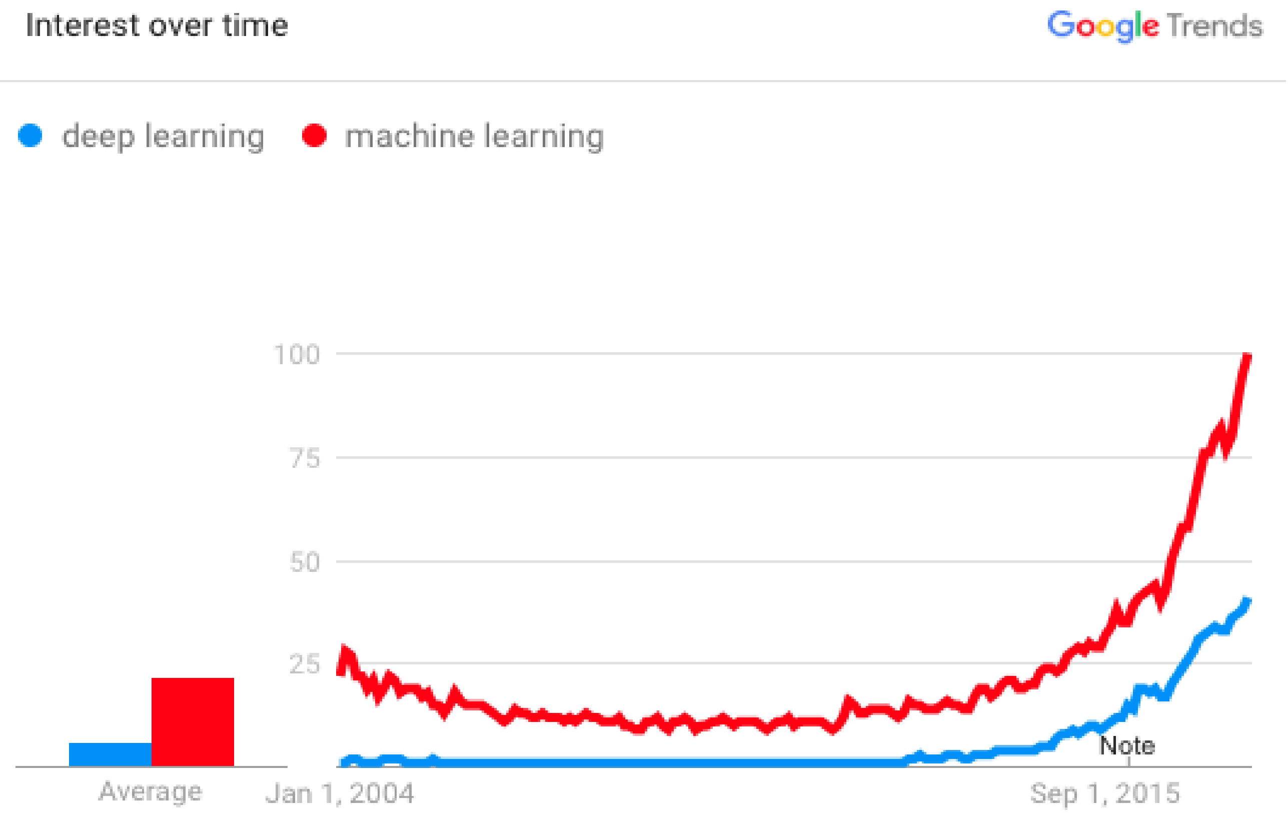
DEEP LEARNING



Why Is DL Useful?

- Manually designed features are often **over-specified**, **incomplete** and take a **long time to design** and validate
- Learned Features are **easy to adapt**, **fast** to learn
- Deep learning provides a very **flexible**, (almost?) **universal**, learnable framework for representing world, visual and linguistic information.
- Can learn both unsupervised and supervised
- Effective **end-to-end** joint system learning
- Utilize large amounts of training data

In ~2010 DL started outperforming other ML techniques
first in speech and vision, then NLP



What We Learn?

Challenges

A big topic…difficult to know where to start…

Unfamiliar terminology…

Hard to abstract…

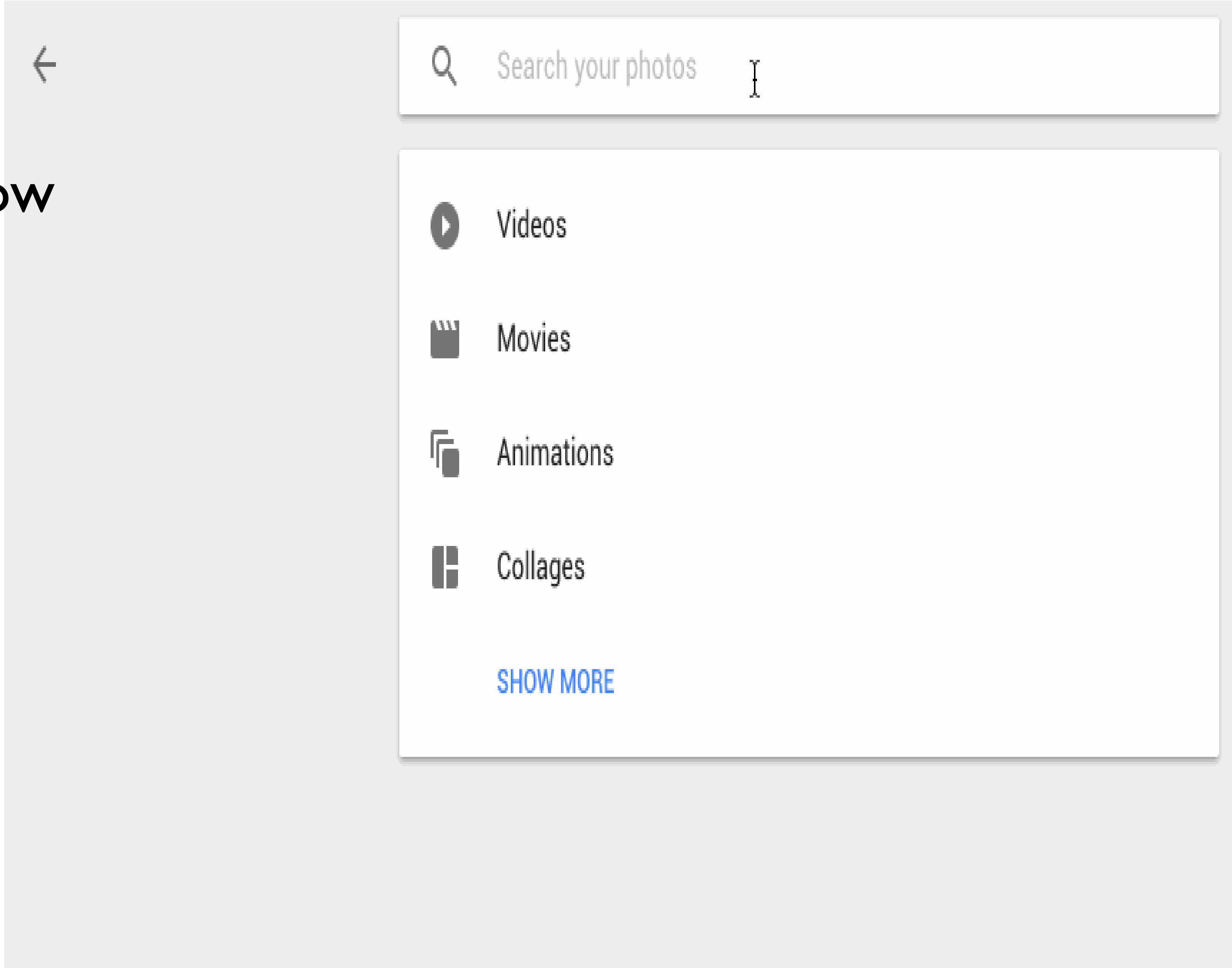
Our Approach

Simplify…

Demystify…

Intuitions…

Miss out…

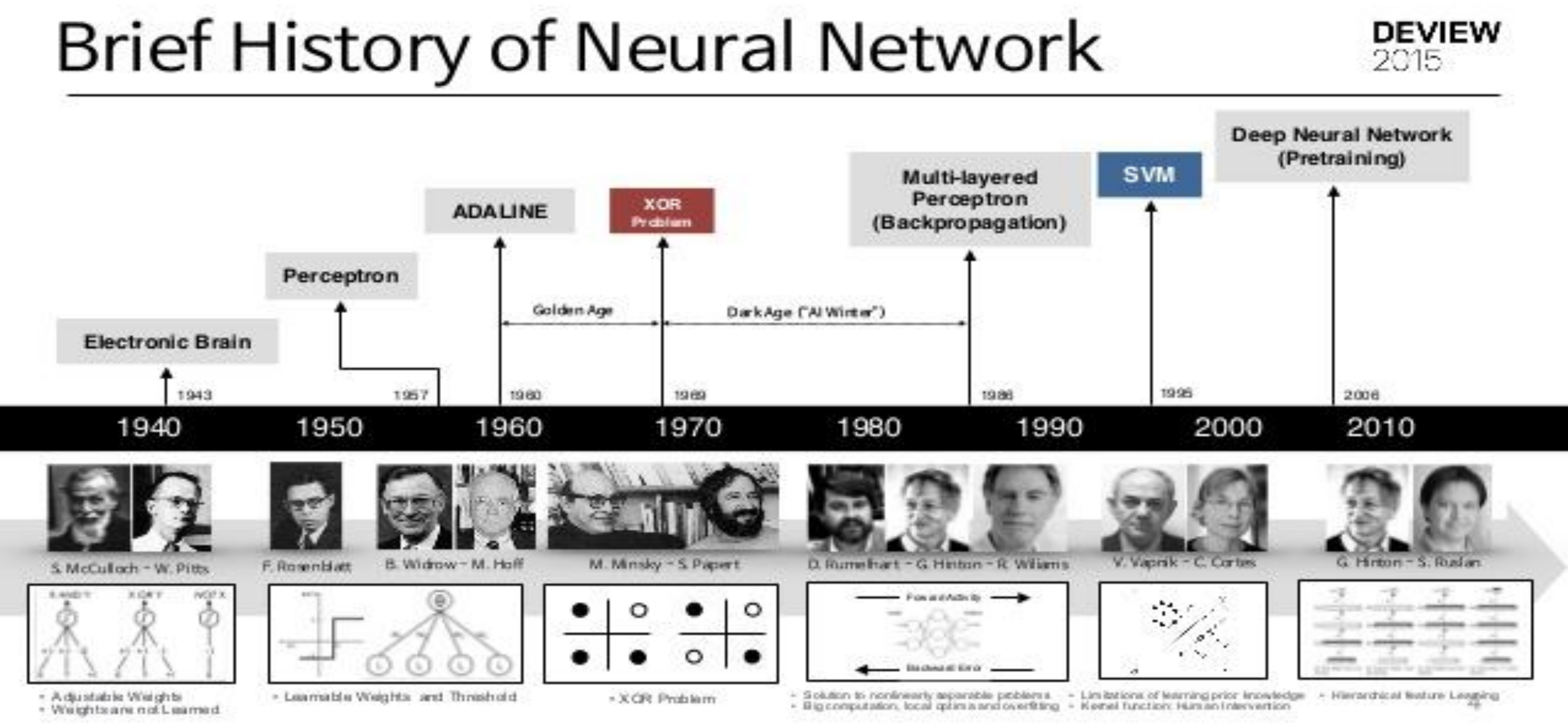


Deep learning isn't magic.

But, it is very good at finding patterns…to make predictions.

Introduction To Deep Learning

History



Brief History of Neural Network
DEVIEW 2015
Electronic Brain
Perceptron
ADALINE
XOR Problem
Multi-layered Perceptron (Backpropagation)
SVM
Deep Neural Network (Pretraining)
Golden Age
Dark Age ("AI Winter")
1943
1957
1960
1969
1986
1995
2006
1940
1950
1960
1970
1980
1990
2000
2010
S. McCulloch - W. Pitts
F. Rosenblatt
B. Widrow - M. Hoff
M. Minsky - S. Papert
D. Rumelhart - G. Hinton - R. Wiliams
V. Vapnik - C. Cortes
G. Hinton - S. Ruslan
• Adjustable Weights
• Weights are not Learned
• Learnable Weights and Threshold
• XOR Problem
• Solution to nonlinearly separable problems
• Big computation, local optima and overfitting
• Limitations of learning prior knowledge
• Kernel function: Human Intervention
• Hierarchical feature Learning

Deep Learning

Dominant technology inspired by Biology...

Neuron

Neural

Network

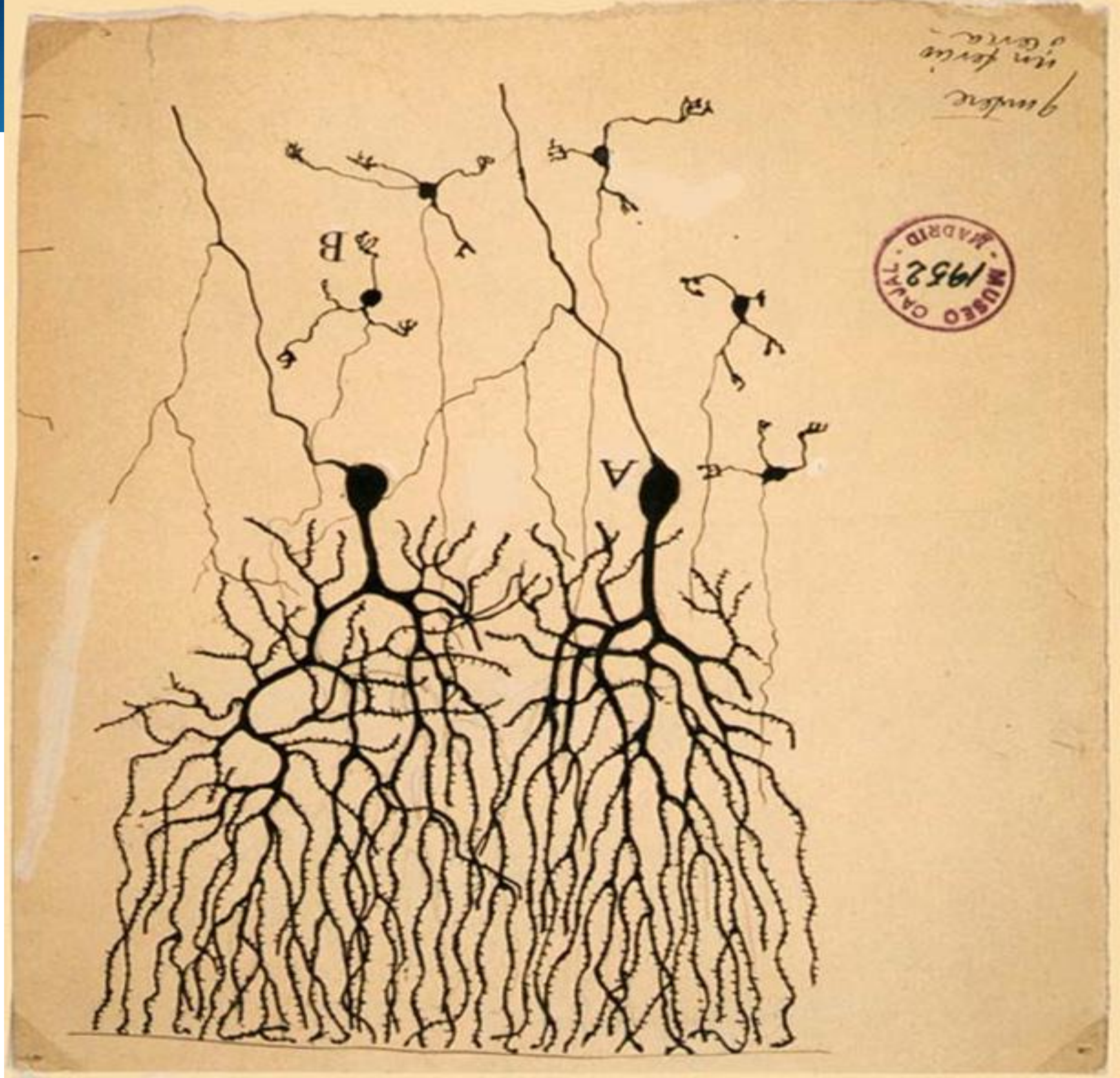


Image Credits: https://en.wikipedia.org/wiki/Neuron

Neural Network (Biology)

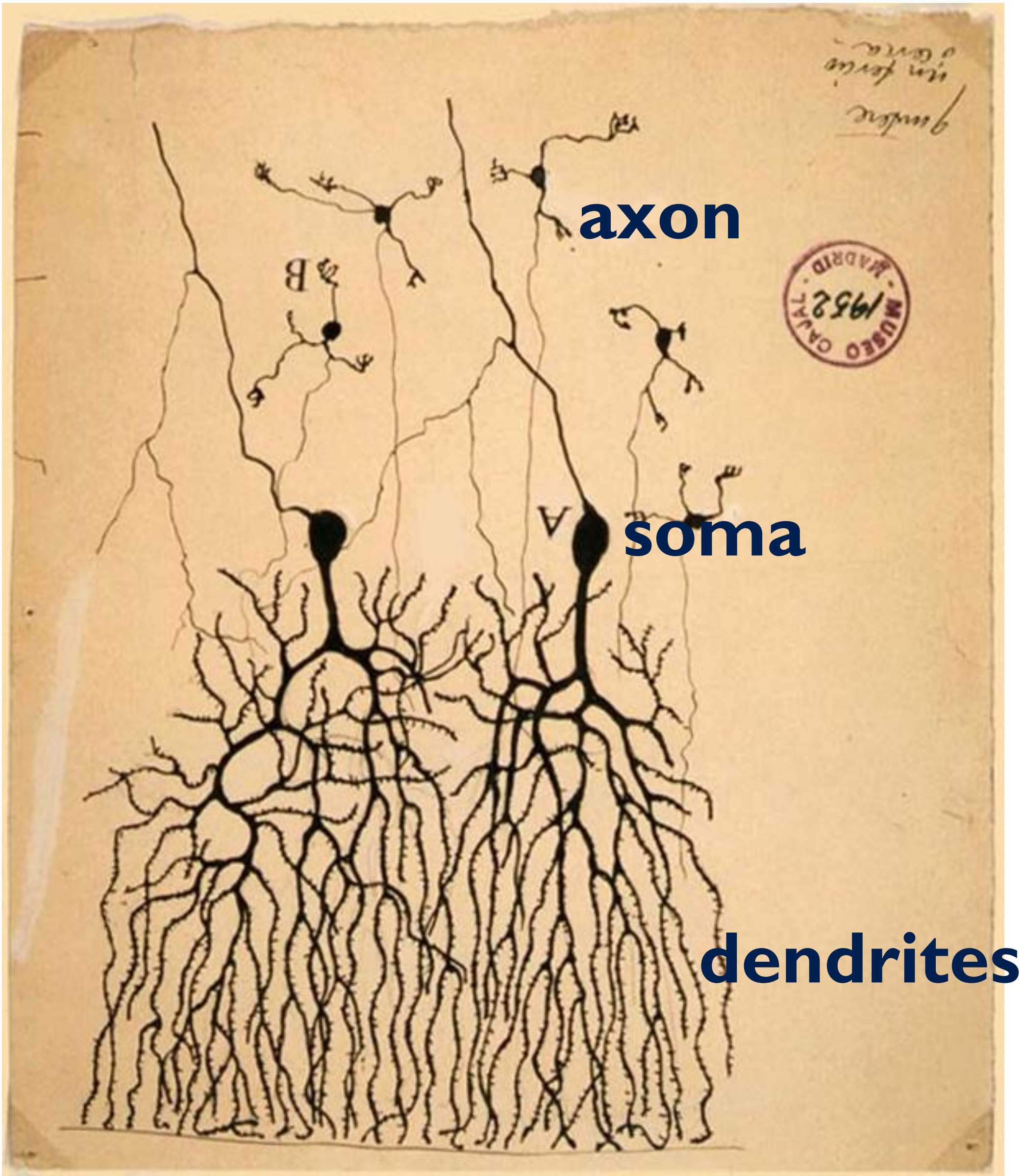
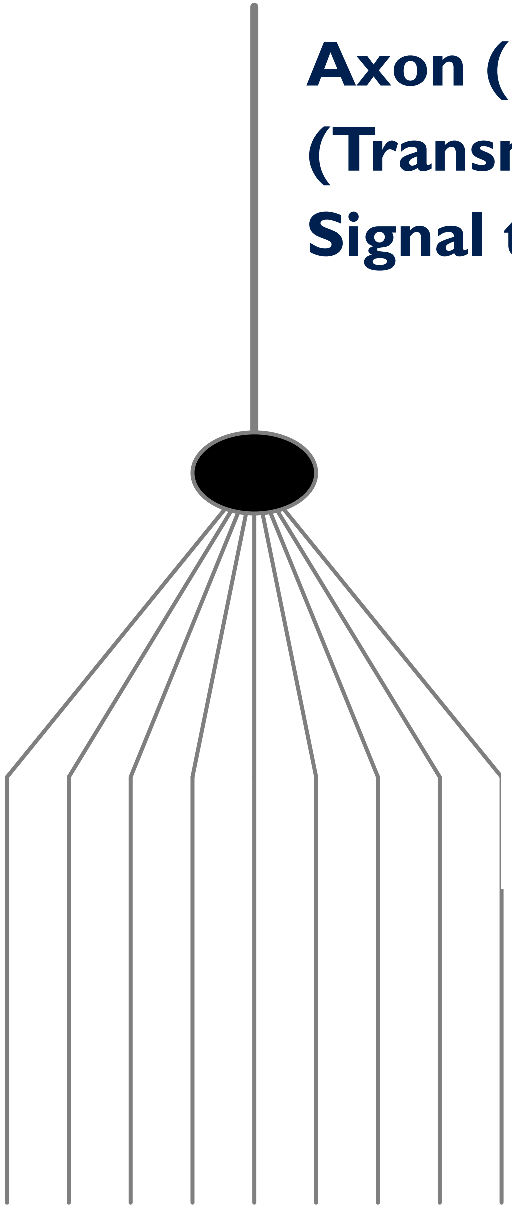


Image Credits: https://en.wikipedia.org/wiki/Neuron



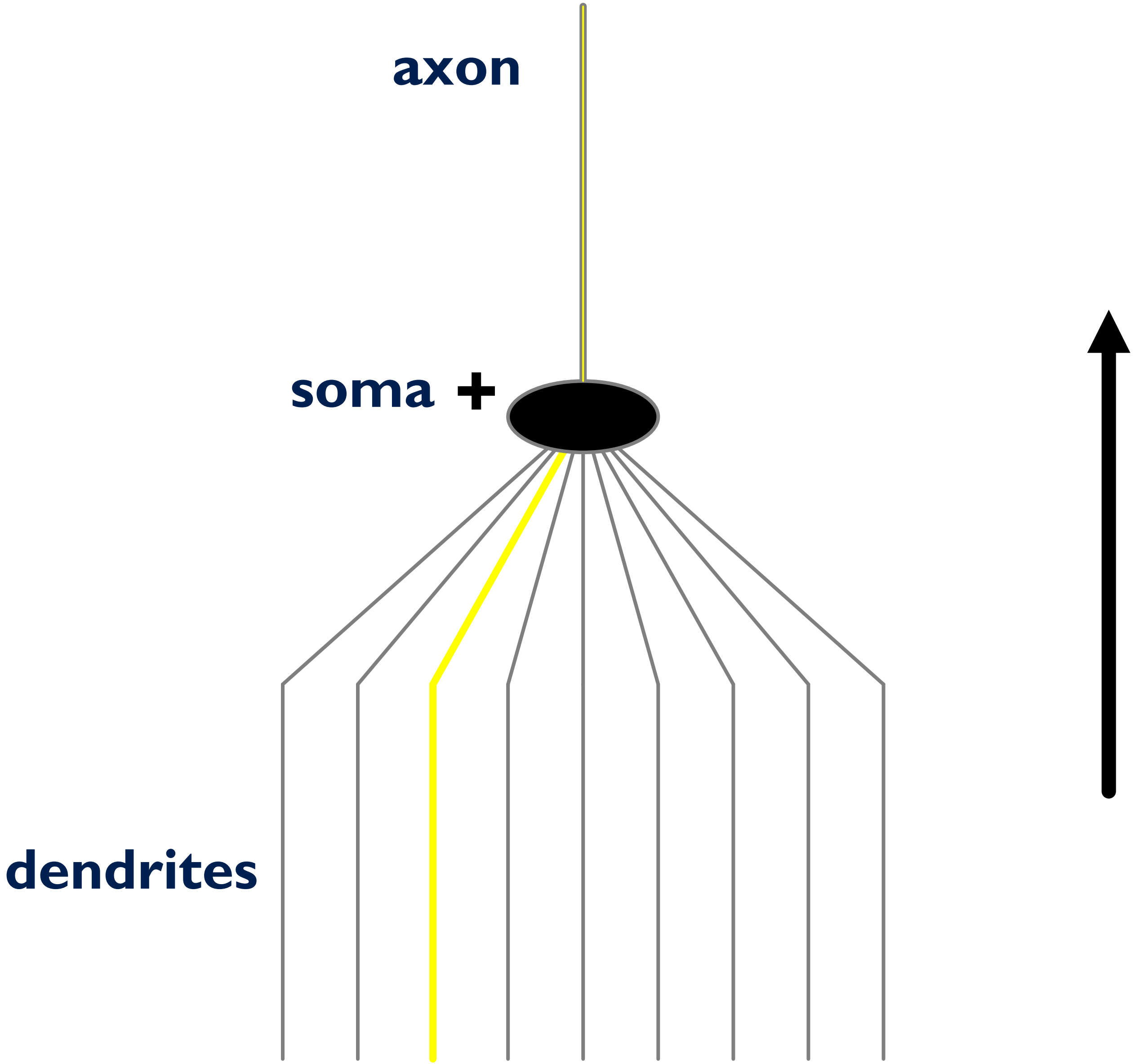
Axon (Output)
(Transmits the output
Signal to other neurons)

Soma
(weighted Summation
Function of positive and
Negative signals from
Dendrites)

Dendrites
(Input Vector)

Neural Network (Biology)…

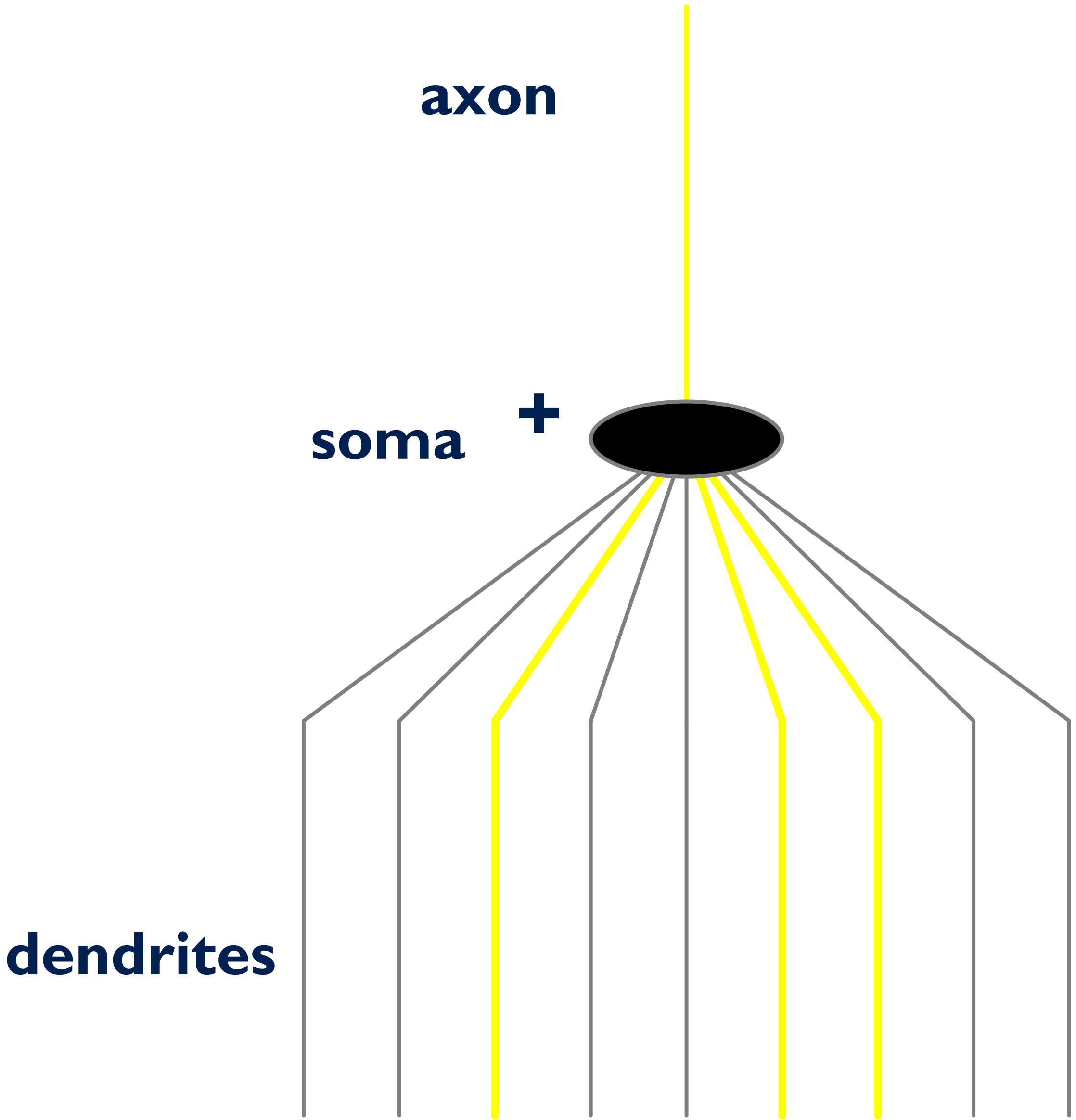
“Soma” combines dendrites activities and passes it to axon.



Neural Network (Biology)…

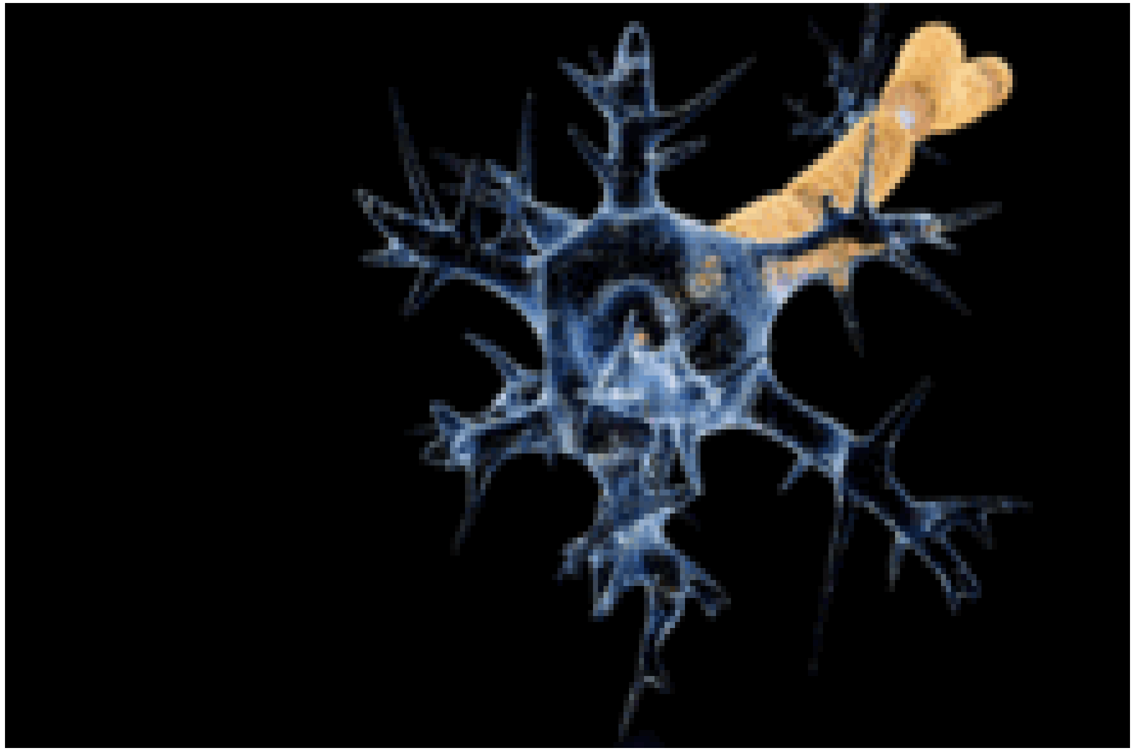
More dendrite activity

→ more axon activity



NEURAL NETWORK (BIOLOGY)...

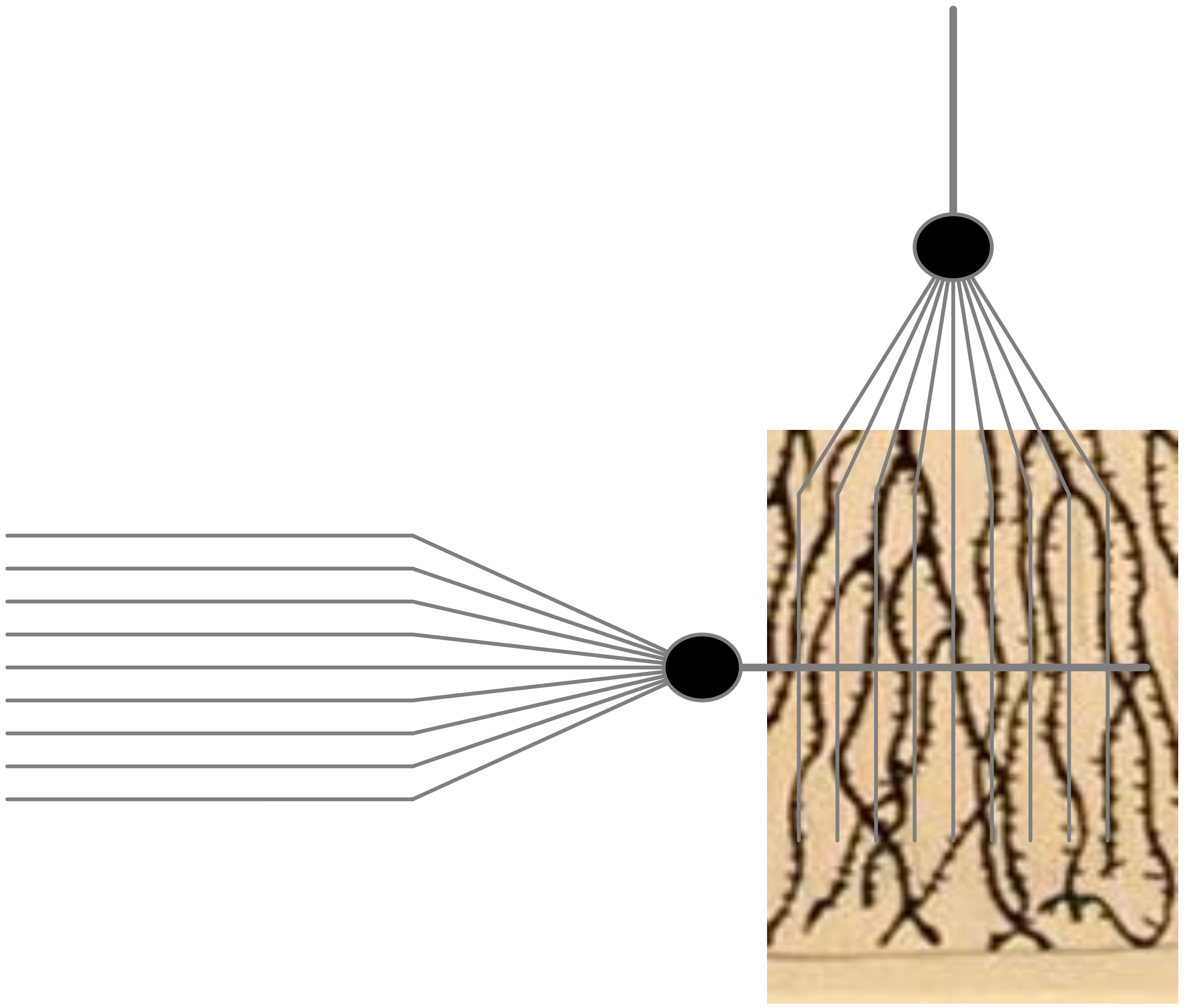
Connection between axon of one neuron and dendrites of another



Credits: https://en.wikipedia.org/wiki/Neuron

Neural Network (Biology)…

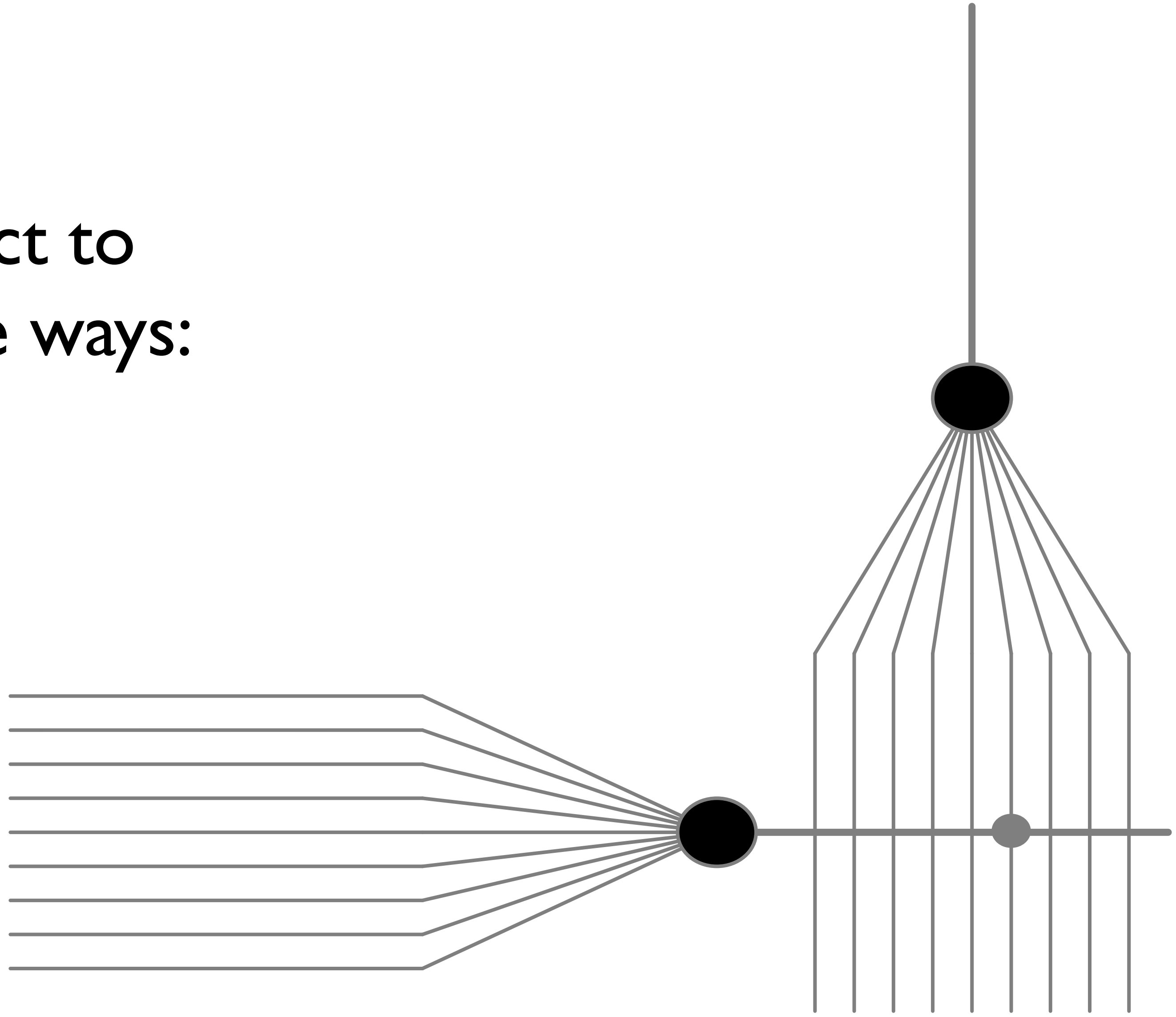
Connection between axon of one neuron and dendrites of another



Neural Network (Biology)…

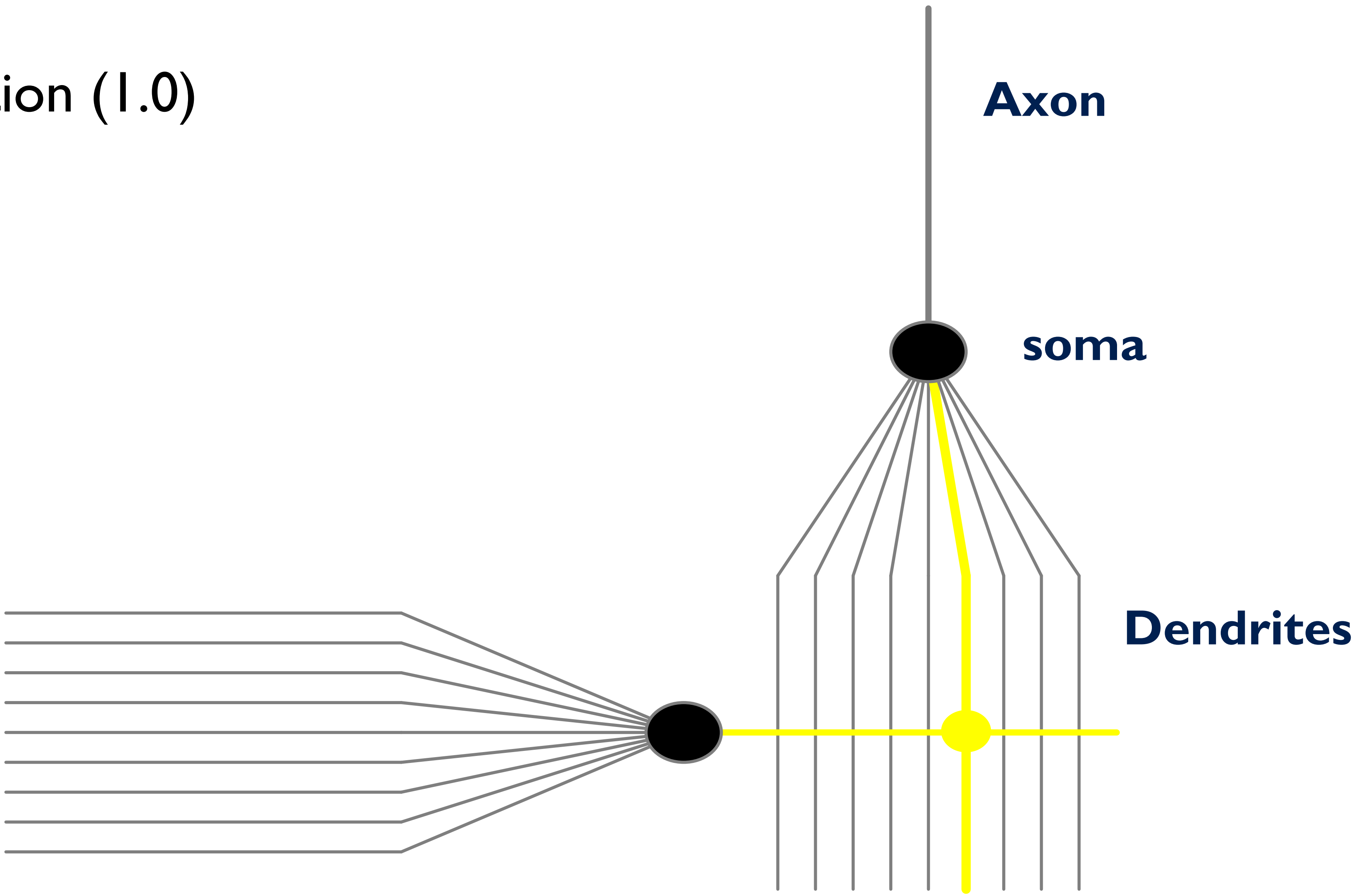
Axons can connect to dendrites in three ways:

- Strongly
- Weakly,
- Medium.



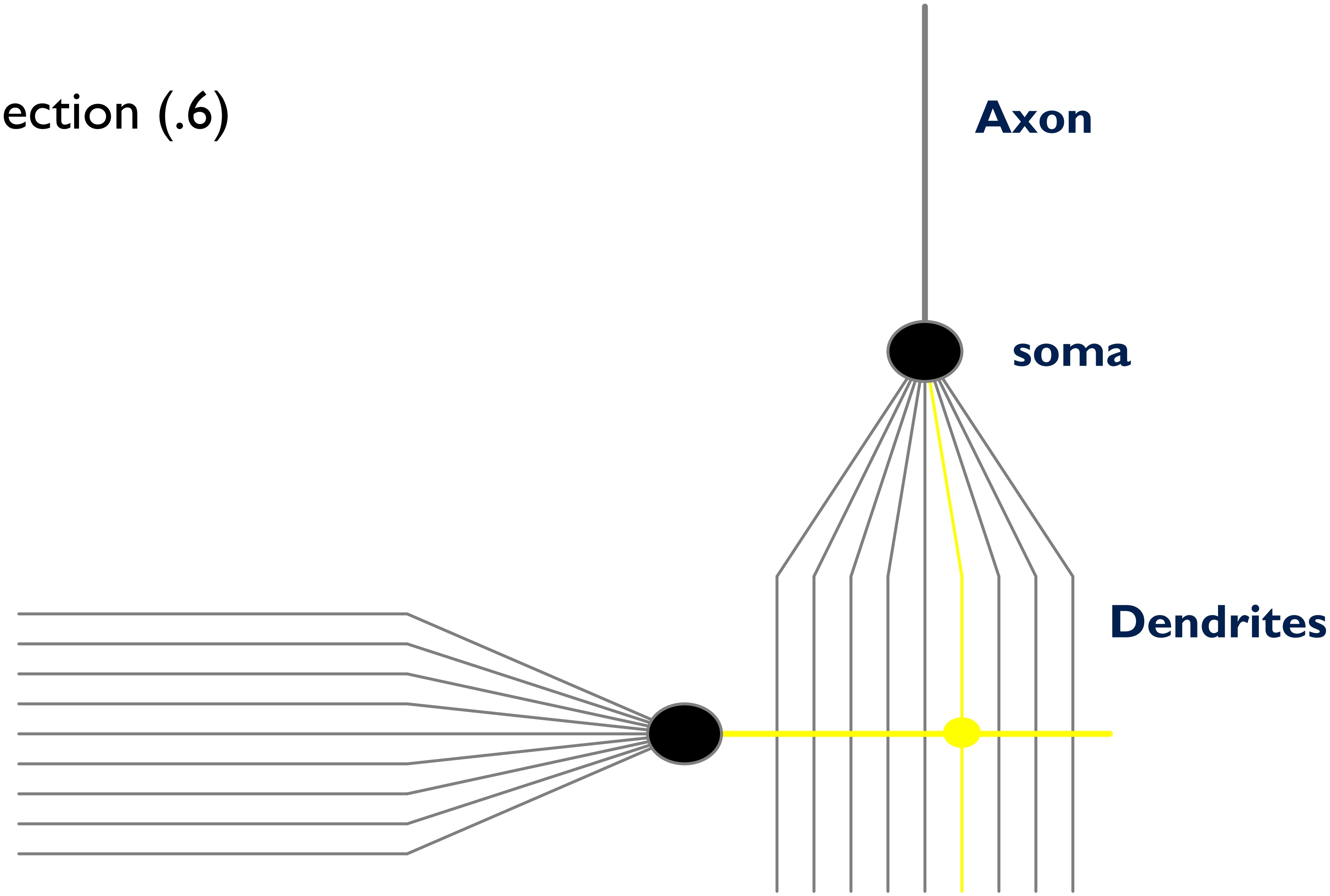
Neural Network (Biology)…

Strong connection (1.0)



Neural Network (Biology)…

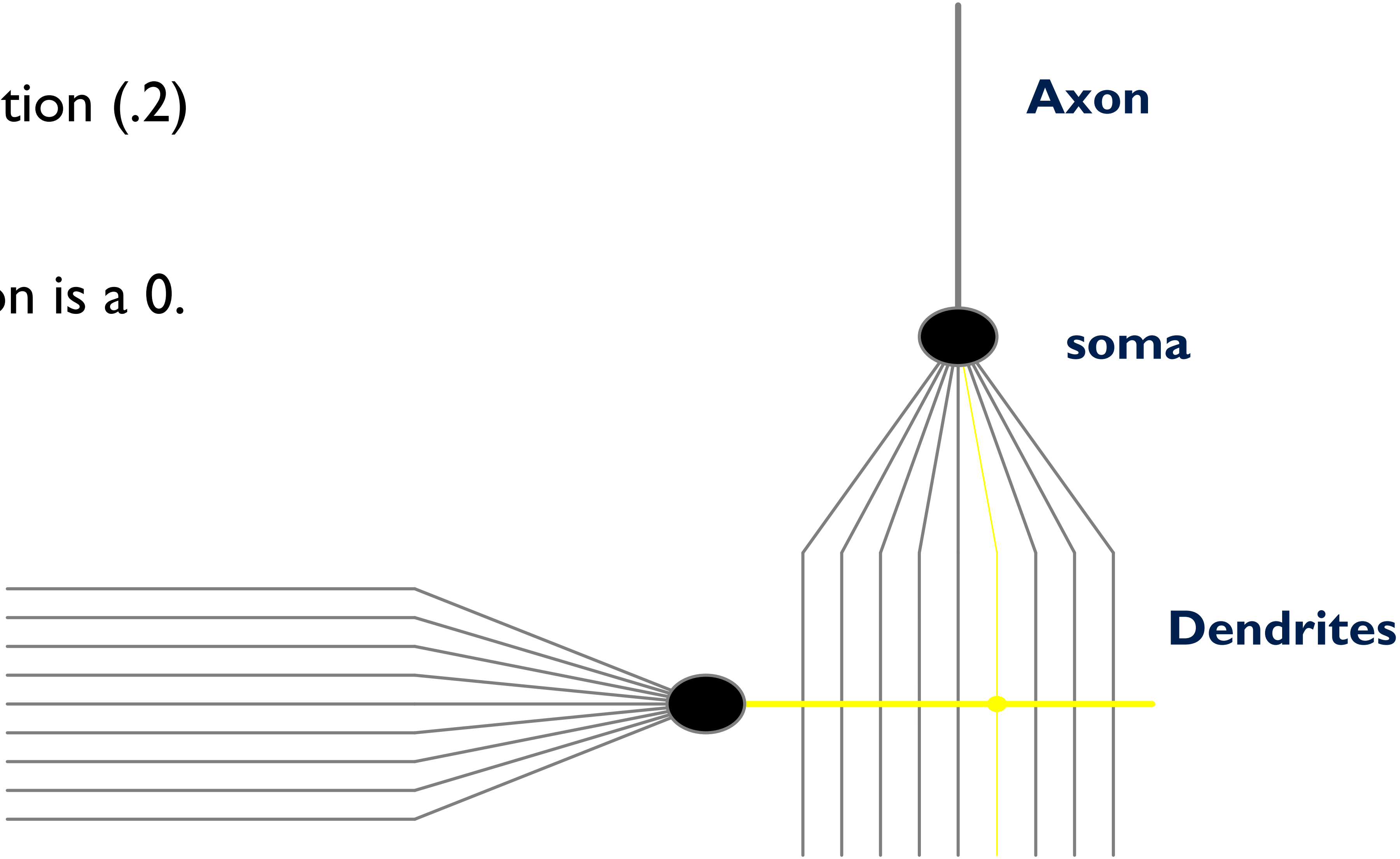
Medium connection (.6)



Neural Network (Biology)…

Weak connection (.2)

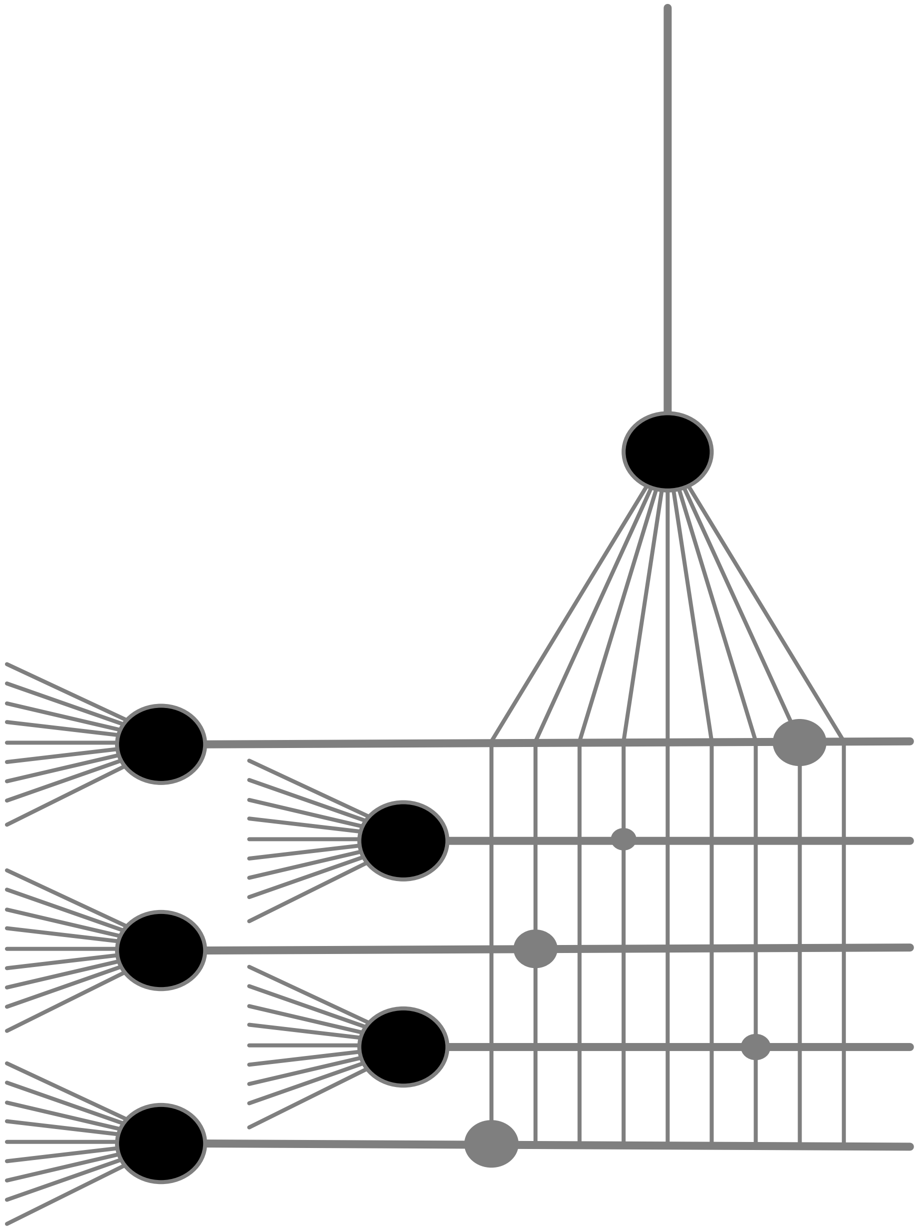
No connection is a 0.



Neural Network (Biology)...

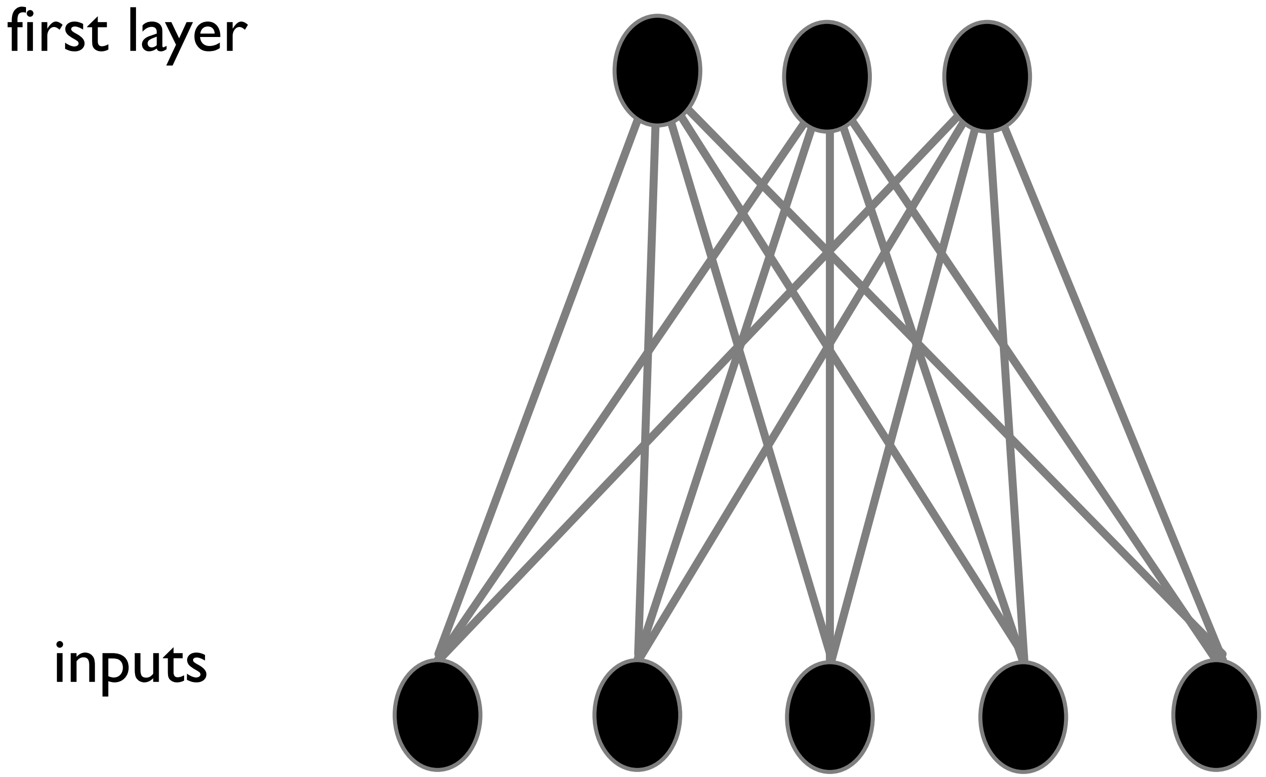
Lots of axons connect with the dendrites of one neuron.

Each has its own connection strength.



Neural Network

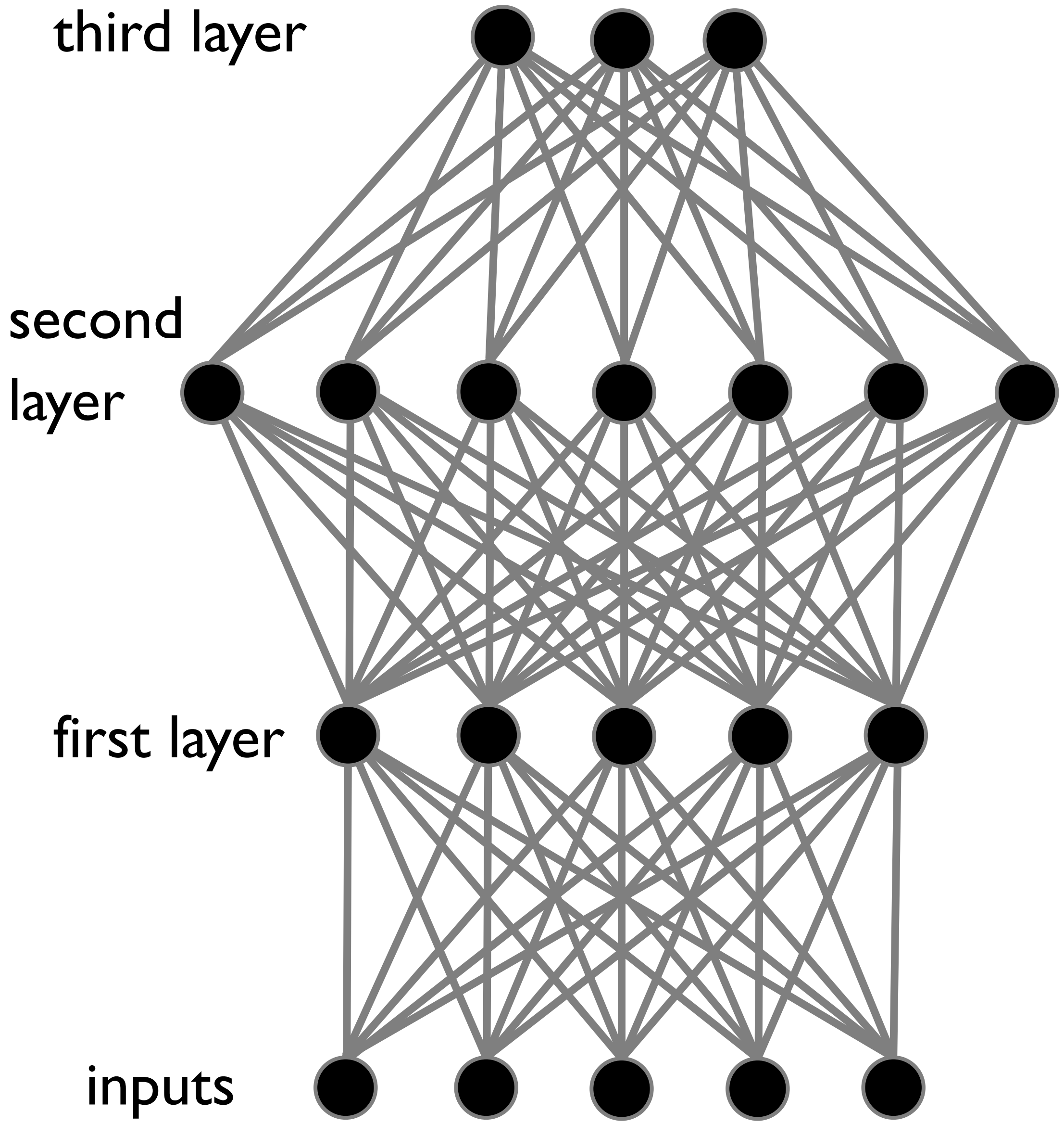
Each node represents a pattern, a combination of the neurons on the previous layer.



DEEP NEURAL NETWORK

If a network has more than three layers, **it's deep.**

Some 10 or more layers.



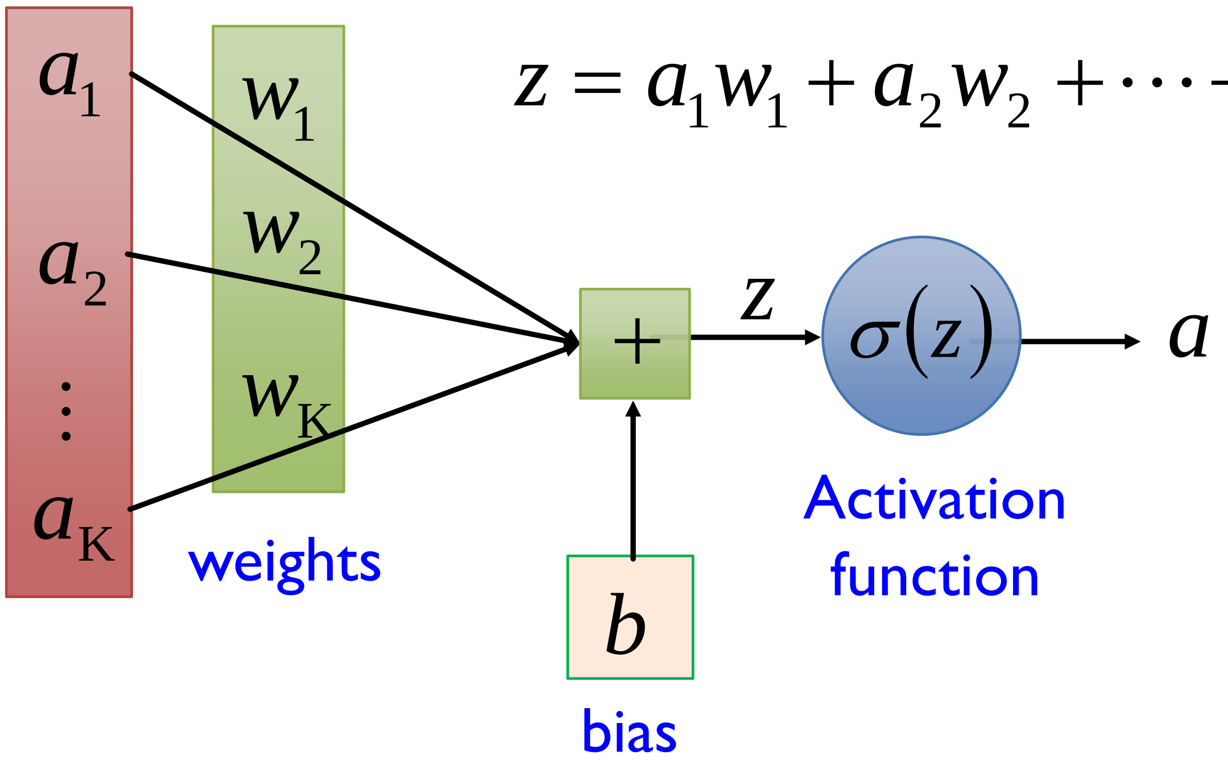
Algorithms

- Single Perceptron
- Multi Layer Perceptron (MLP) – Feed Forward NN
- Backpropagation
- Convolution Neural Networks (CNN)
- Auto encoders (For Dimension Reduction, Unsupervised)
- Recurrent Neural Networks (Simple RNN, LSTM)

Single Perceptron

Representation Of A Neuron

Neuron $f: R^K \rightarrow R$



$$z = a_1 w_1 + a_2 w_2 + \cdots + a_K w_K + b$$

What happens in a neuron?

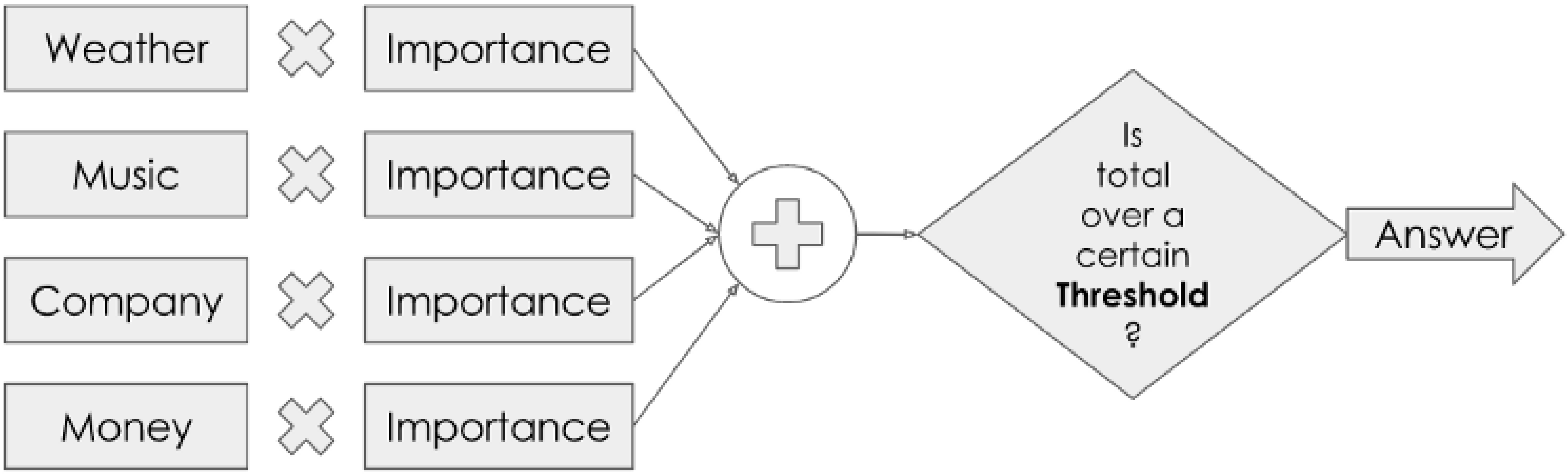
Problem: Whether to go to a cricket match or not?

- Will the weather be nice?
- What's the music like?
- Do I have anyone to go with?
- Can I afford it?
- Do I need to write my exam?
- Will I like the food in the food stalls at stadium?

Just take the first four for simplicity

What happens in a neuron?

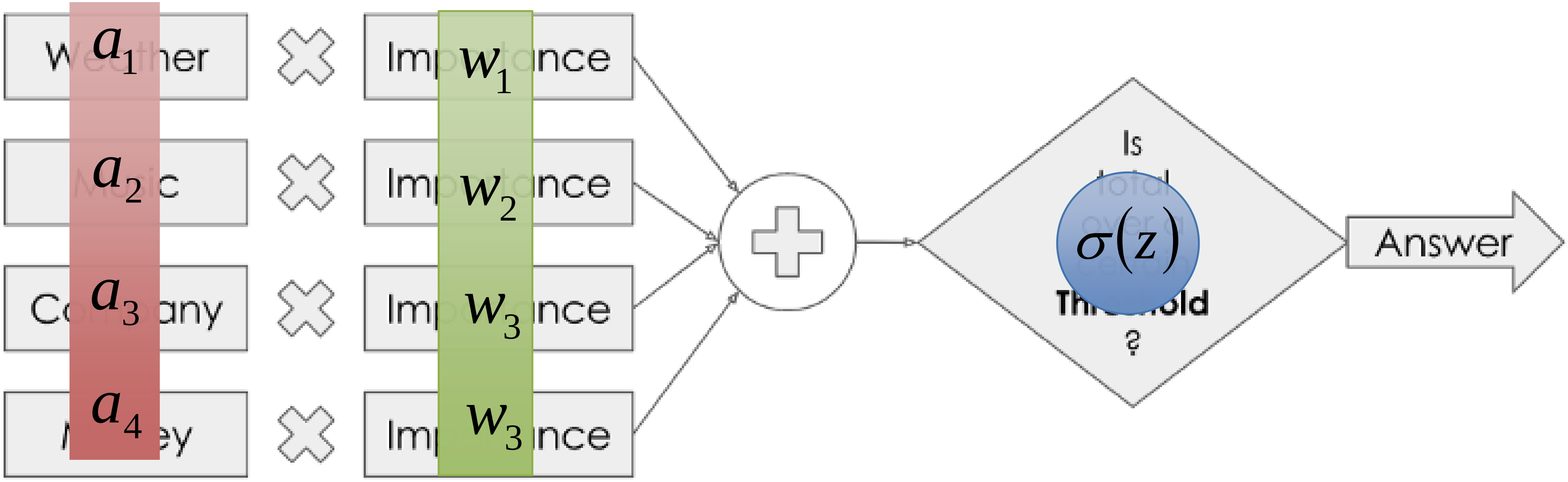
Problem: Whether to go to a cricket match or not?



To make my decision, **I would consider how important each factor was to me, weigh them** all up, and see if the result was over a certain threshold. If so, I will go to the match! It's a bit like the process we often use of weighing up pros and cons to make a decision.

What happens in a neuron?

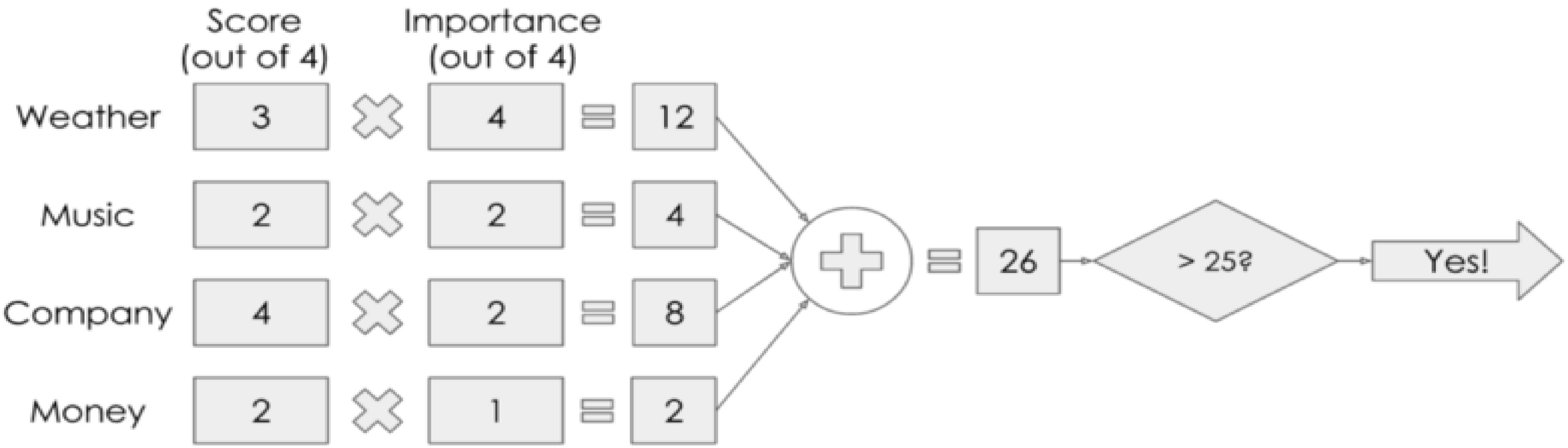
Problem: Whether to go to a cricket match or not?



To make my decision, **I would consider how important each factor was to me, weigh them** all up, and see if the result was over a certain threshold. If so, I will go to the match! It's a bit like the process we often use of weighing up pros and cons to make a decision.

What happens in a neuron?

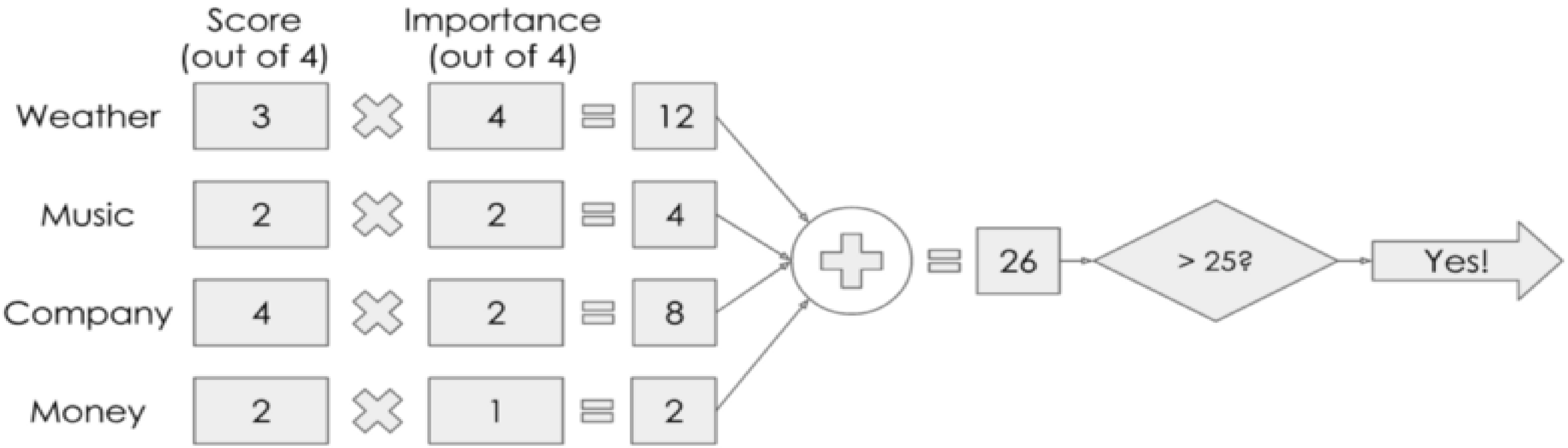
Problem: Whether to go to a cricket match or not?



Give some Attributes and Weights…use a single perceptron…

What happens in a neuron?

Problem: Whether to go to a cricket match or not?

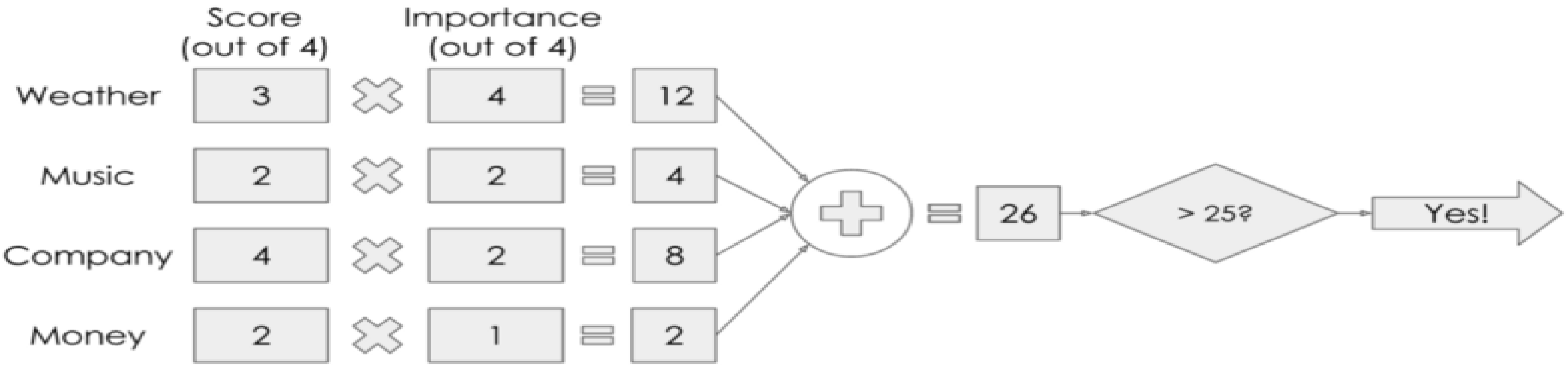


Scores:

- **weather** is looking pretty good but not perfect - 3 out of 4.
- **music** is ok but not my favourite - 2 out of 4.
- **Company -** my best friend has said he/she'll come with me so I know the company will be great, - 4 out of 4.
- **Money** - little pricey but not completely unreasonable - 2 out of 4.

WHAT HAPPENS IN A NEURON?

Problem: Whether to go to a cricket match or not?

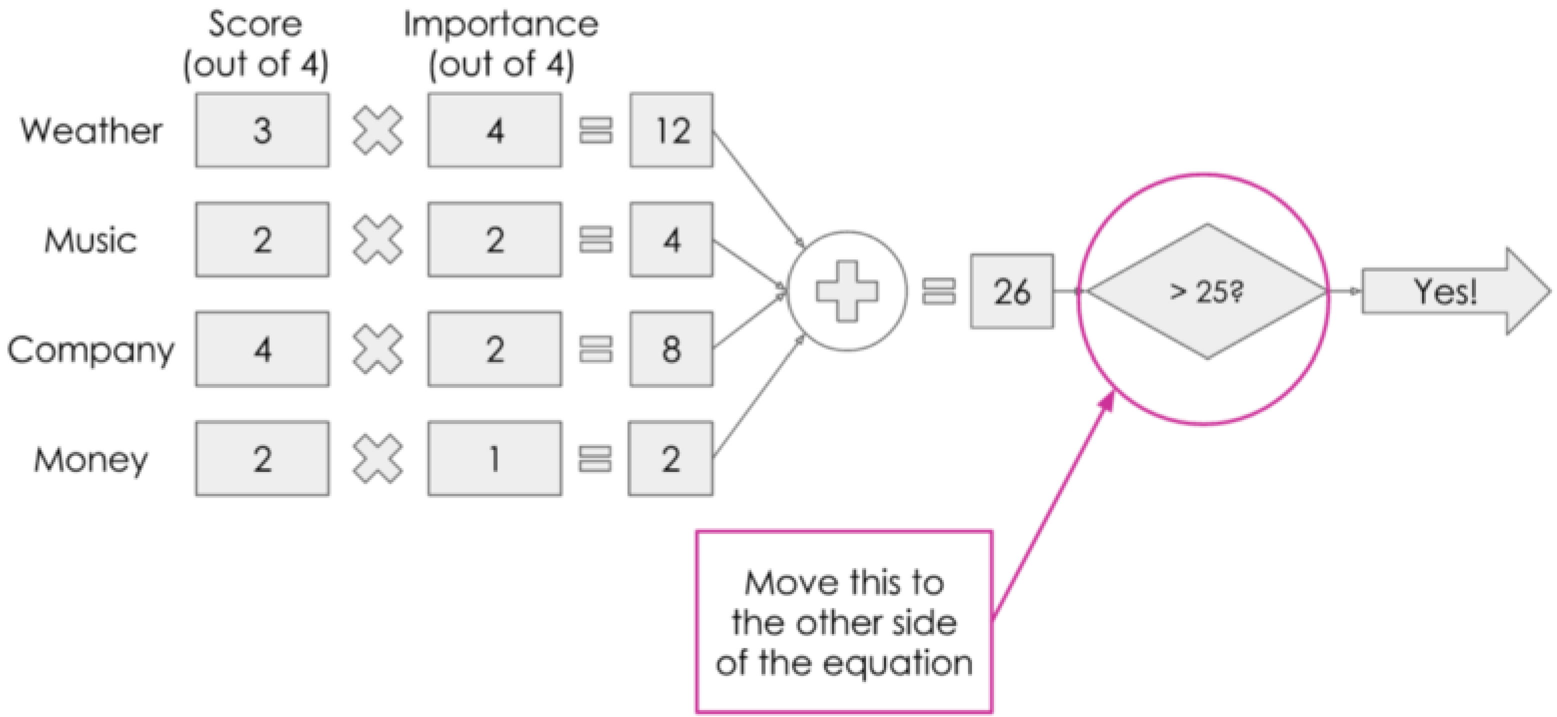


Now for the importance:

- **Weather** - I really want to go if it's sunny, very imp - I give it a full **4 out of 4**.
- **Music** - I'm happy to listen to most things, not super important - **2 out of 4.**
- **Company** - I wouldn't mind too much going to the match on my own, so company can have a **2 out of 4** for importance.
- **Money** - I'm not too worried about money, so I can give it just a **1 out of 4**.

What happens in a neuron?

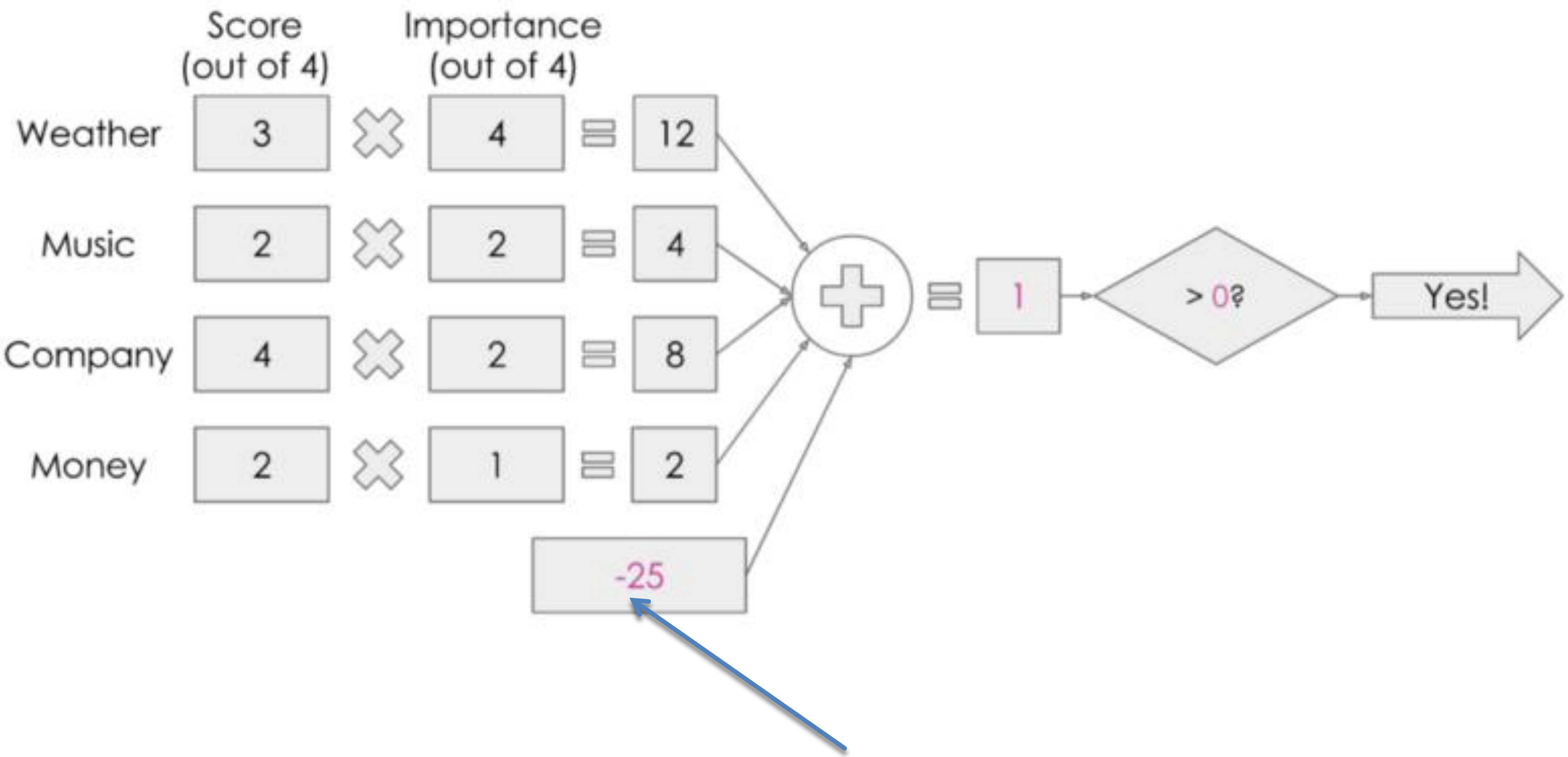
Problem: Whether to go to a cricket match or not?



Total Score = Sum of (Score x Weight) = 26
Threshold = 25
Score (26) > Threshold (25) → Going to Cricket Match

What happens in a neuron?

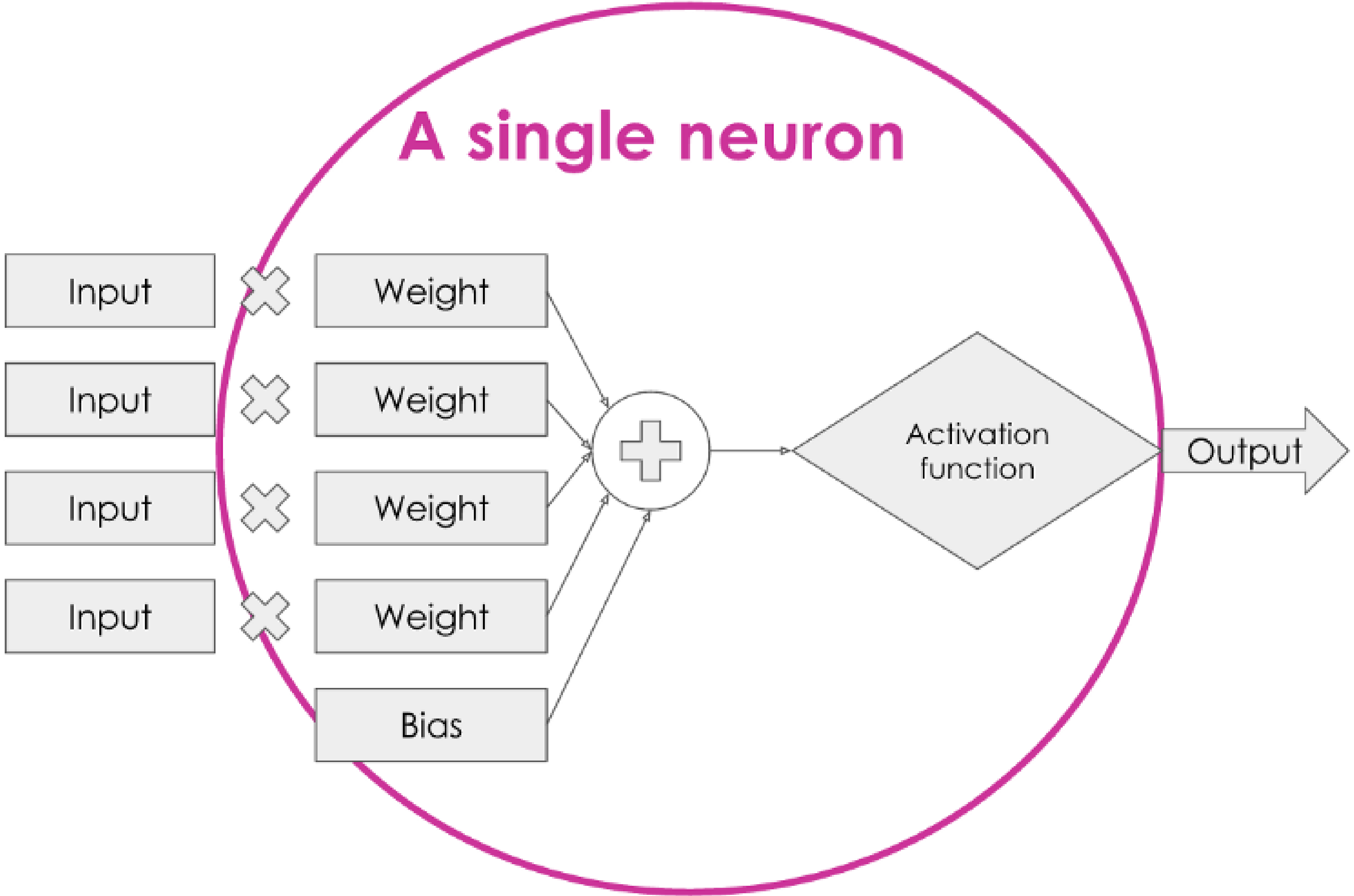
Problem: Whether to go to a cricket match or not?



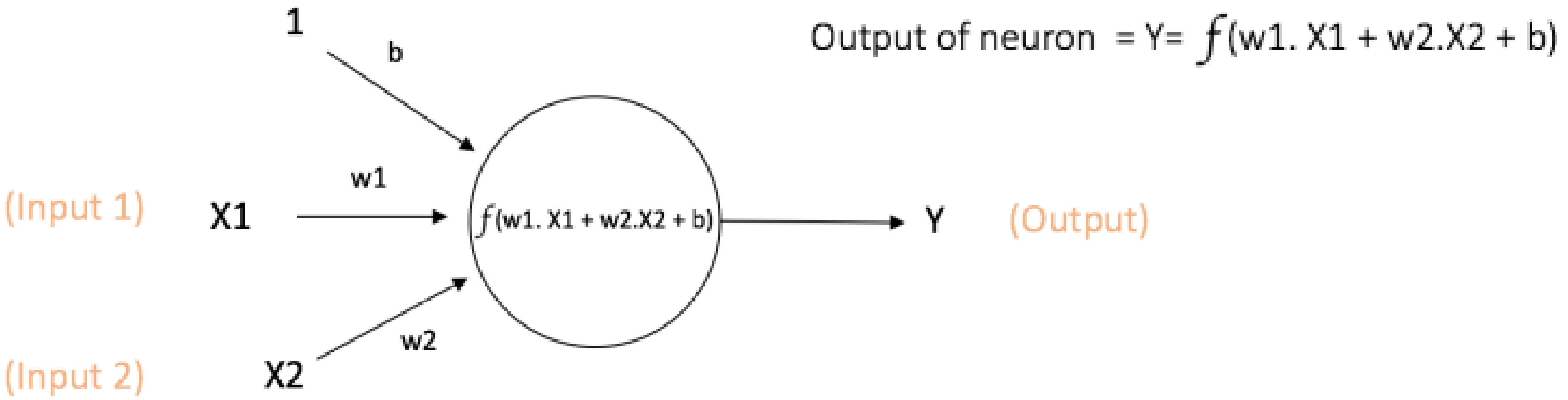
This number is what is known as the **'bias'**

What happens in a neuron?

A general form of a Neuron/Single Perceptron



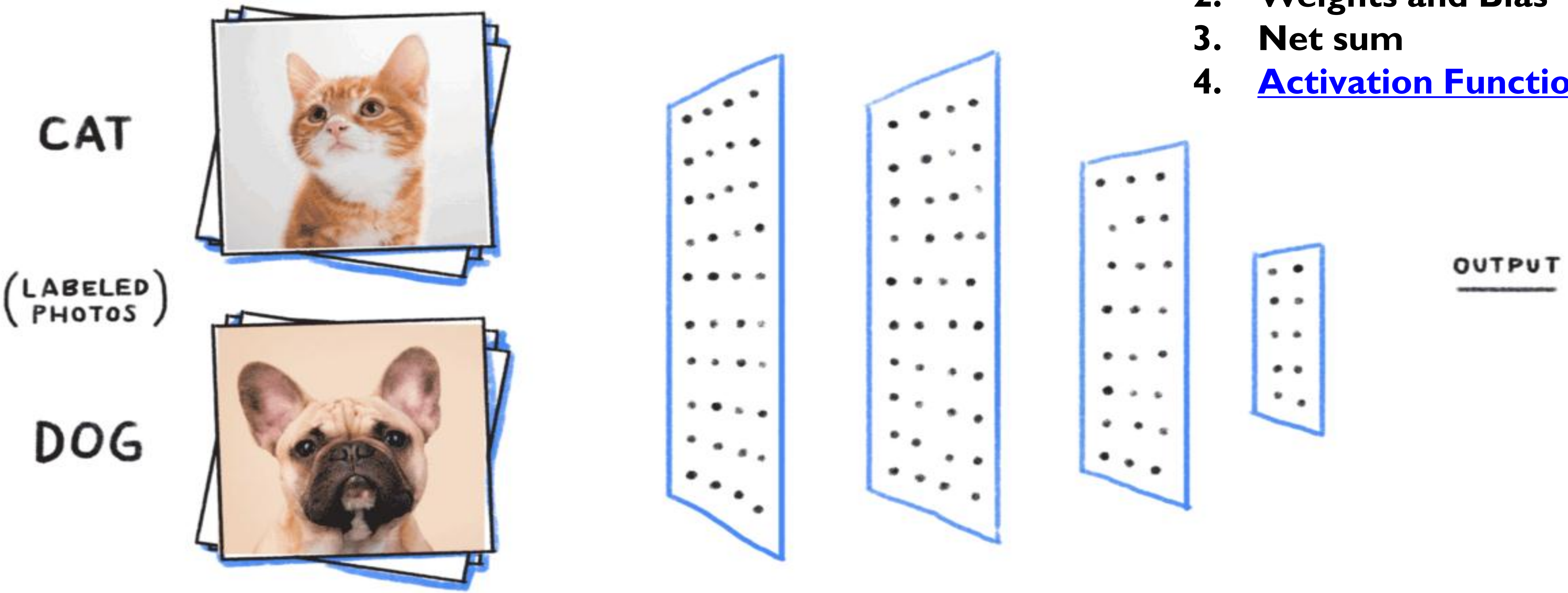
A perceptron



- The basic unit of computation in a neural network is the **neuron**, often called a **node** or **unit**.
- receives input from some other nodes, or from an external source and computes an output.
- Each input has an associated **weight** (w), which is assigned on the basis of its relative importance
- The node applies a function ***f*** (defined below) to the weighted sum of its inputs

A perceptron

1. **Input values or One input layer**
2. **Weights and Bias**
3. **Net sum**
4. **Activation Function**



- *A multi-layer perceptron is called Neural Network.*

A perceptron…how it works?

1. All the inputs ***x*** are multiplied with their weights ***w***. Let's call it ***k.***

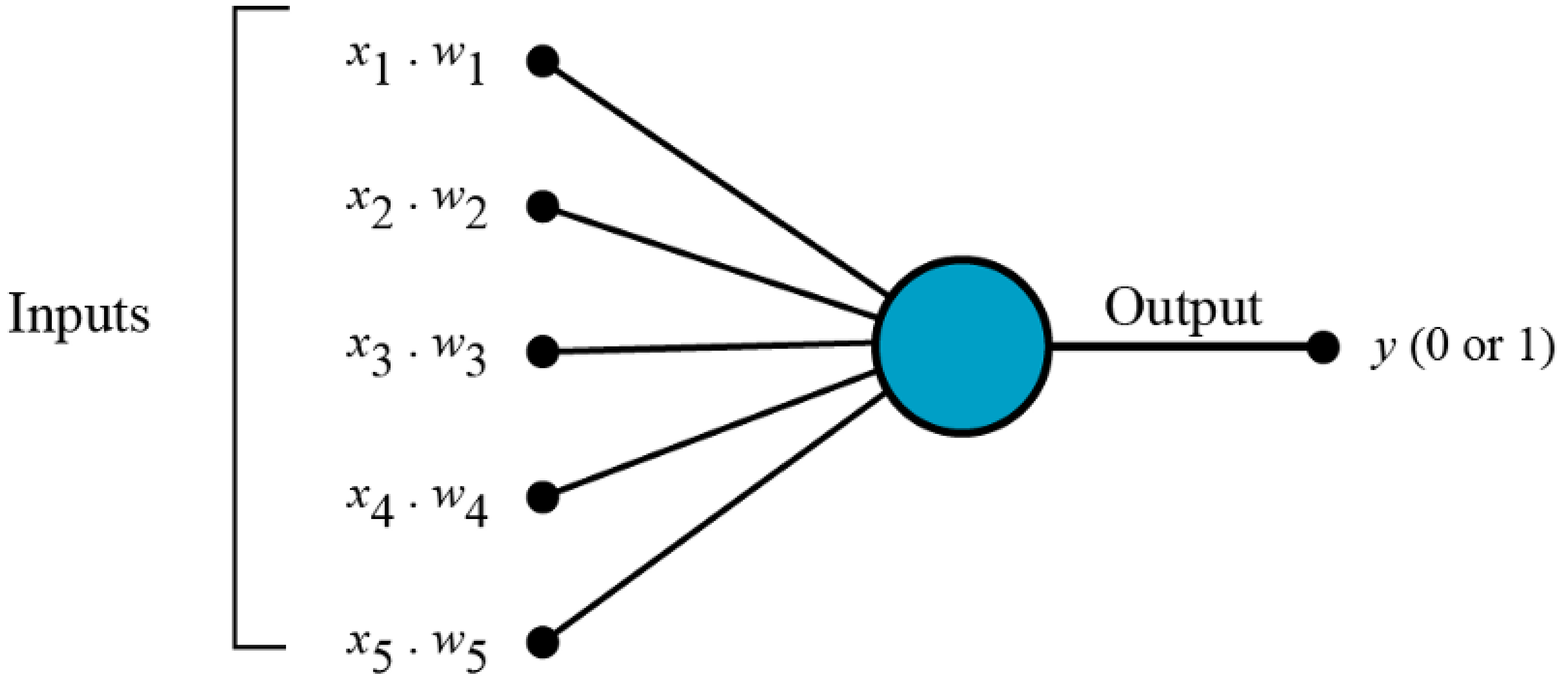


Fig: Multiplying inputs with weights for 5 inputs

A perceptron…how it works?

2. ***Add*** all the multiplied values and call them ***Weighted Sum.***

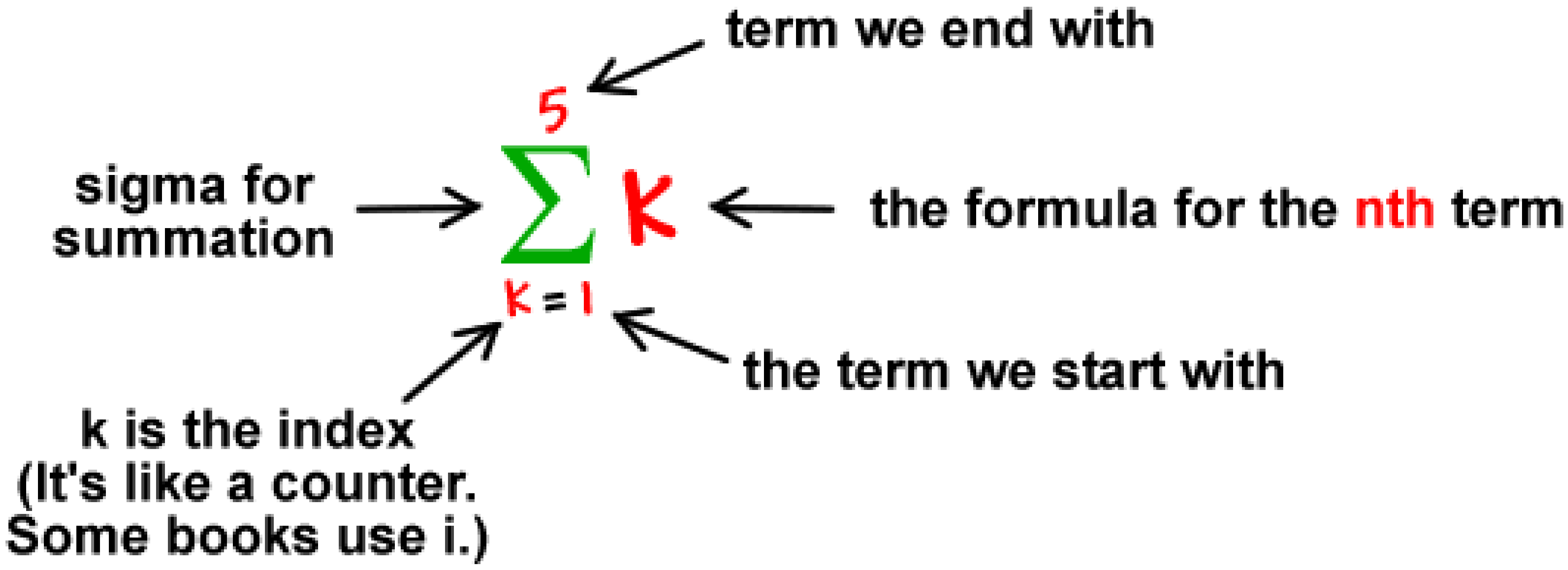


Fig: Adding with Summation

Three components of perceptron

a) Weights (W)

b) Bias (b)

c) Activation Function

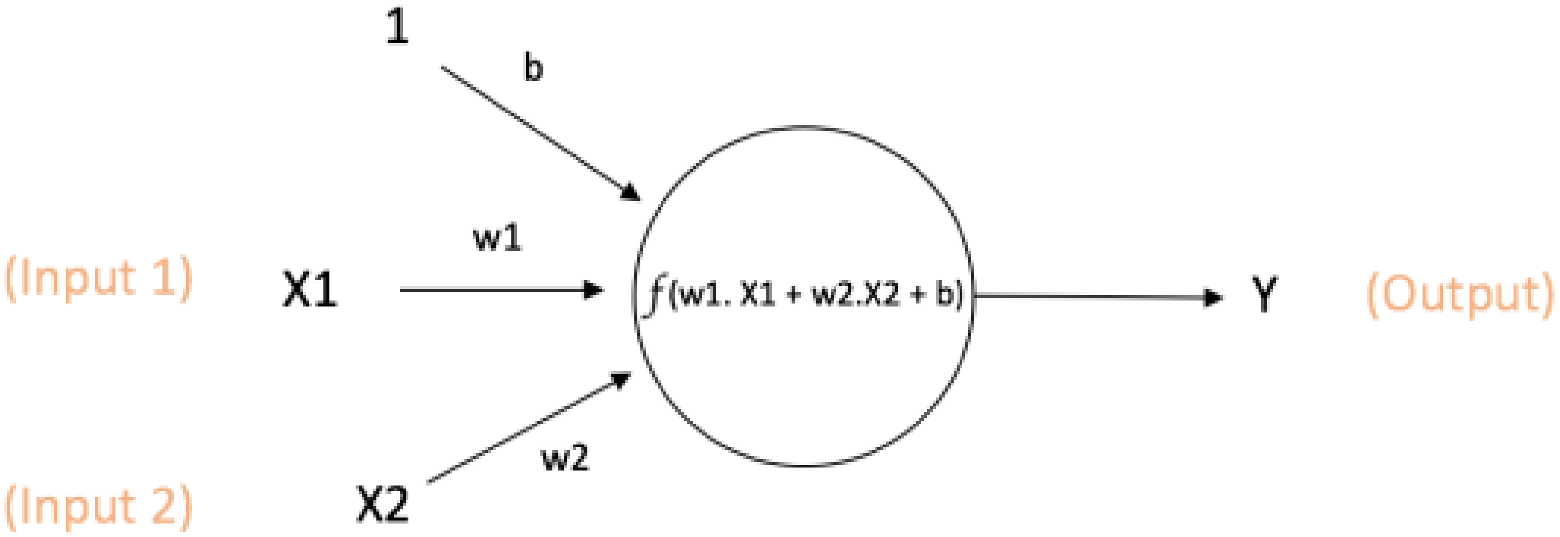
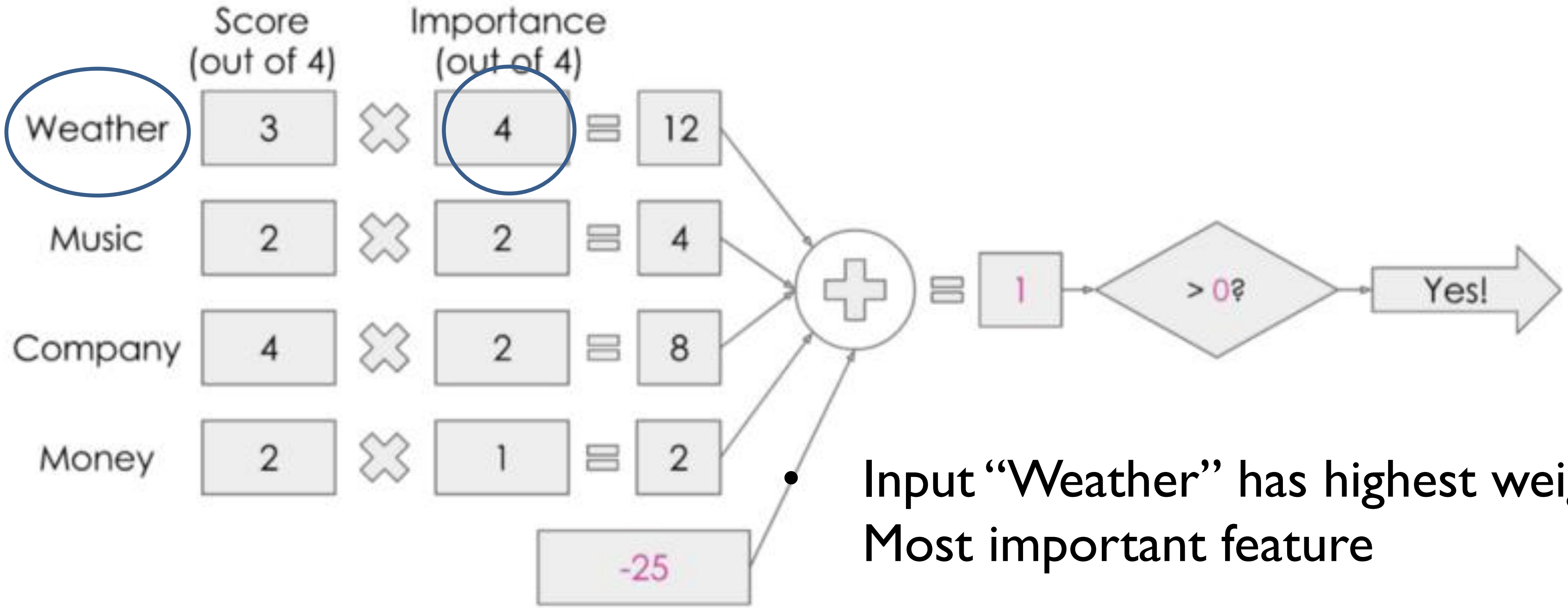


Fig: Adding with Summation

Parameters – a) Weights

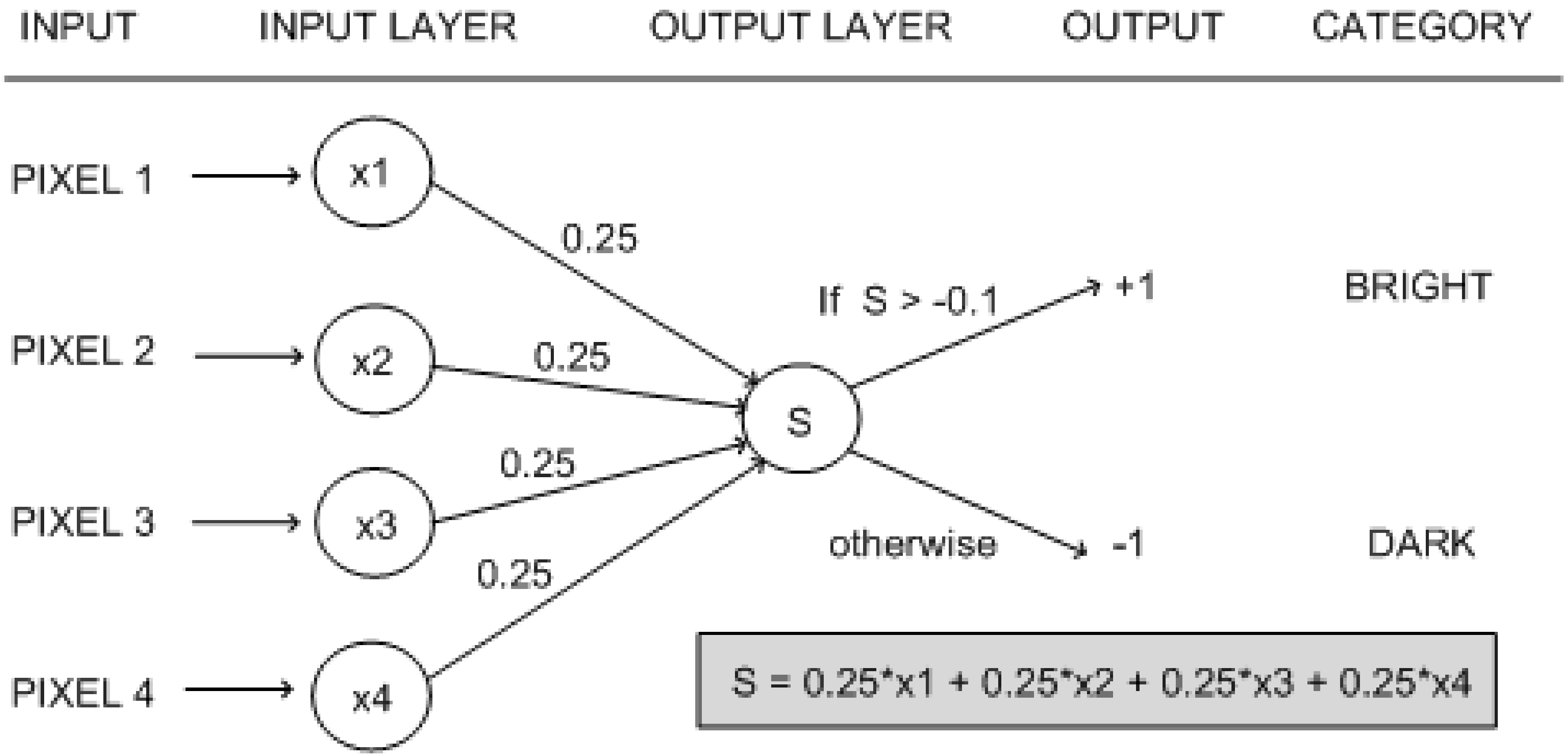
- *Weights show the strength of the particular node/input.*
- If a neuron has two inputs, then each input will have has an associated weight assigned to it.
- Initialize the weights randomly and update weights with model training.
- Input with higher weight is more important.



- Input "Weather" has highest weight → Most important feature

Parameters – b) Bias

- *A bias value allows you to shift the activation function to the left or right.*
- Another linear component is applied to the input.



c) Activation function

Why do we need “activation function”?

- ***Their main purpose is to convert a input signal of a node in a A-NN to an output signal.***
- *In short, the activation functions are used to* ***map the input between the required values like (0, 1) or (-1, 1).***
- *It is also known as* ***Transfer Function.***
- ***They introduce non-linear properties to our Network.***
- That output signal now is used as a input in the next layer in the stack.

The Activation Functions can be basically divided into 2 types-

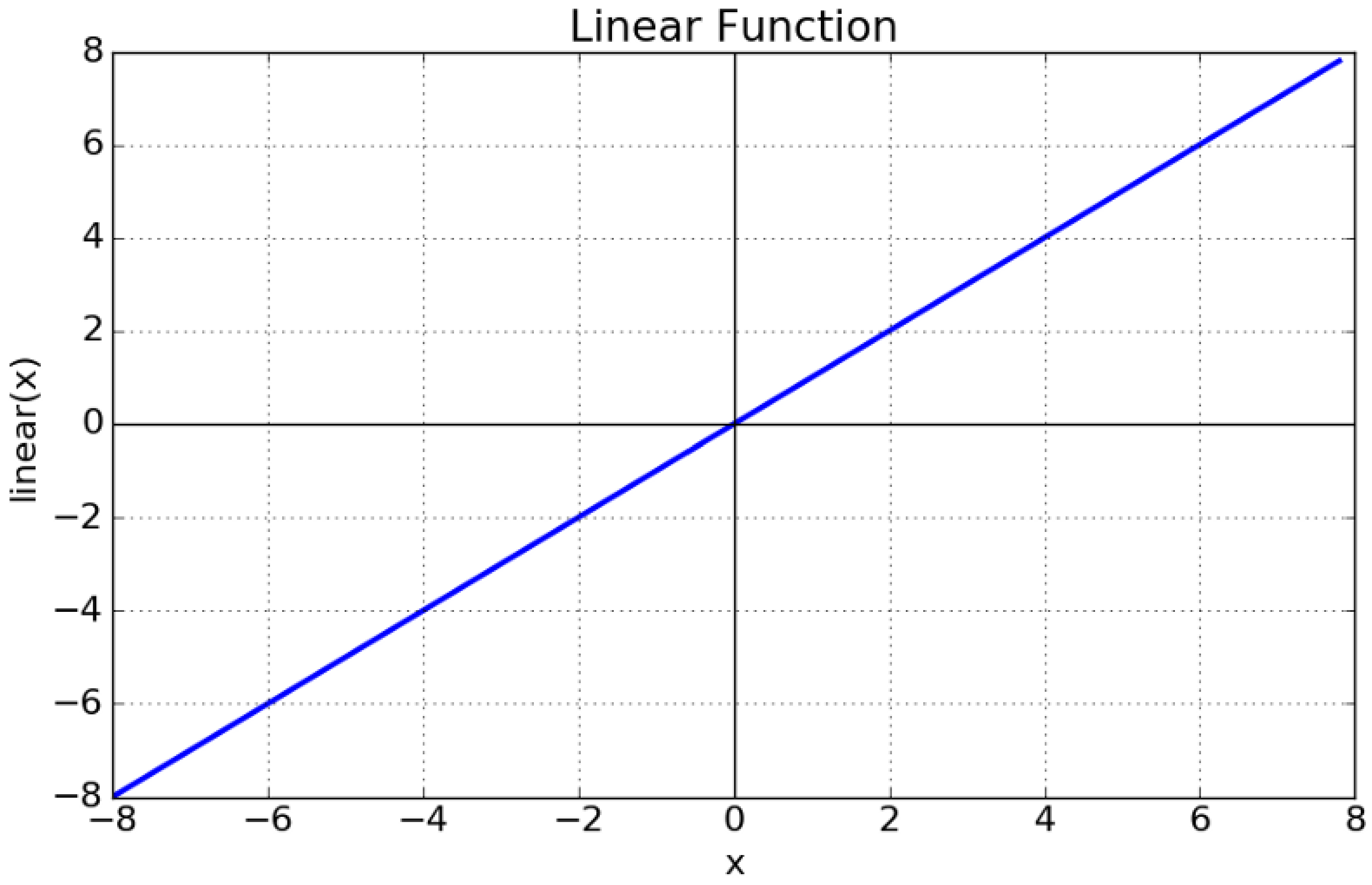
- Linear Activation Function (We don’t use it in DL)
- Non-linear Activation Functions

Linear activation function

Equation : $f(x) = x$

Range : (-infinity to infinity)

It doesn’t help with the complexity or various parameters of usual data that is fed to the neural networks.



why can’t we do it without activating the input signal?

- Linear Activation functions are just a function of a polynomial of Degree 1.
- They have less power to solve complex problems.
- A NN with linear activation function can be treated as a linear regression model
- without activation function our Neural network would not be able to learn and model other complicated kinds of data such as images, videos , audio , speech etc.
- **Examples:**
  - Sigmoid or Logistic
  - *Tanh - Hyperbolic tangent*
  - *ReLu - Rectified linear units*

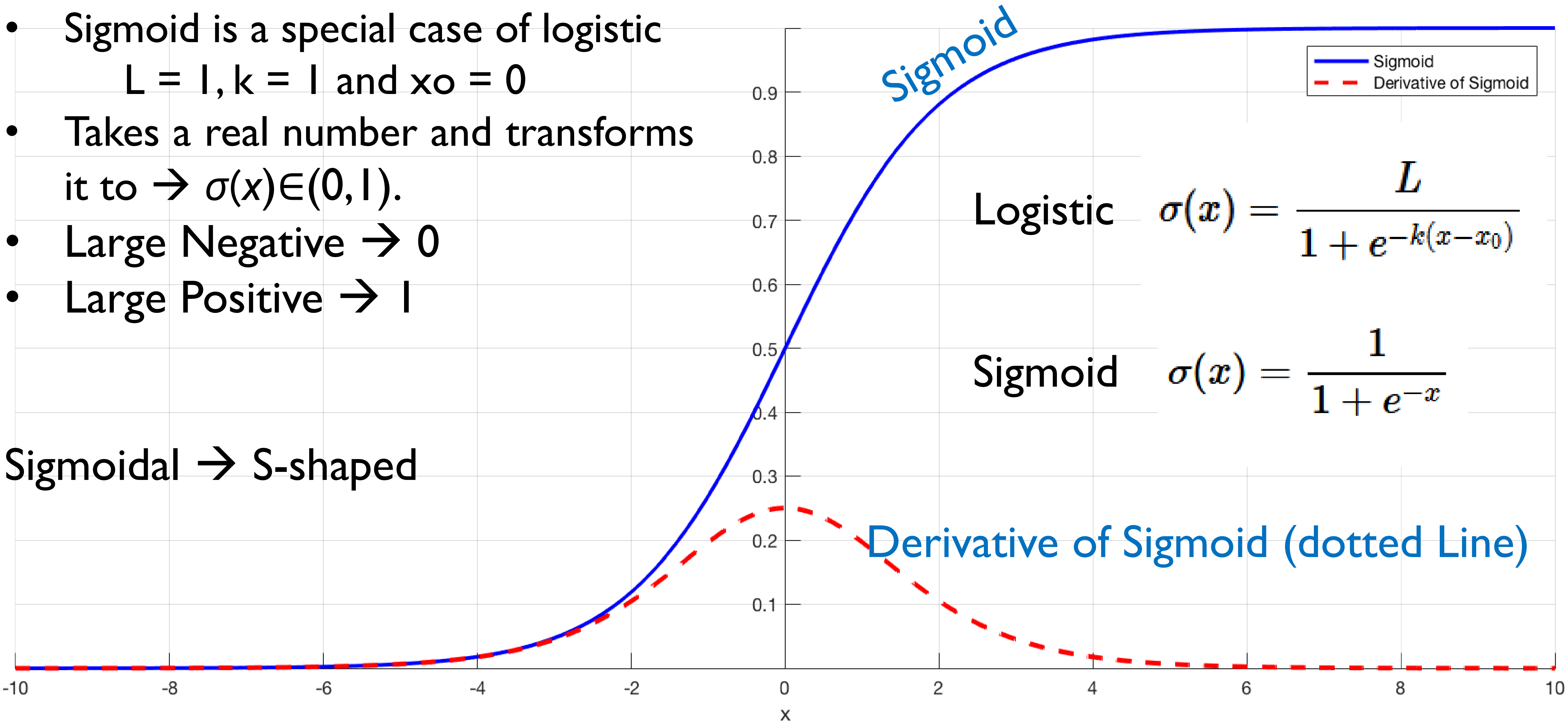
Sigmoid

- Sigmoid is a special case of logistic
  L = 1, k = 1 and xo = 0
- Takes a real number and transforms it to → $\sigma(x) \in (0,1)$.
- Large Negative → 0
- Large Positive → 1

Sigmoidal → S-shaped

Logistic $$\sigma(x) = \frac{L}{1 + e^{-k(x - x_0)}}$$

Sigmoid $$\sigma(x) = \frac{1}{1 + e^{-x}}$$



Sigmoid – Disadvantages

1) Sigmoids saturate and kill gradients

- neuron's activation saturates at either tail of 0 or 1

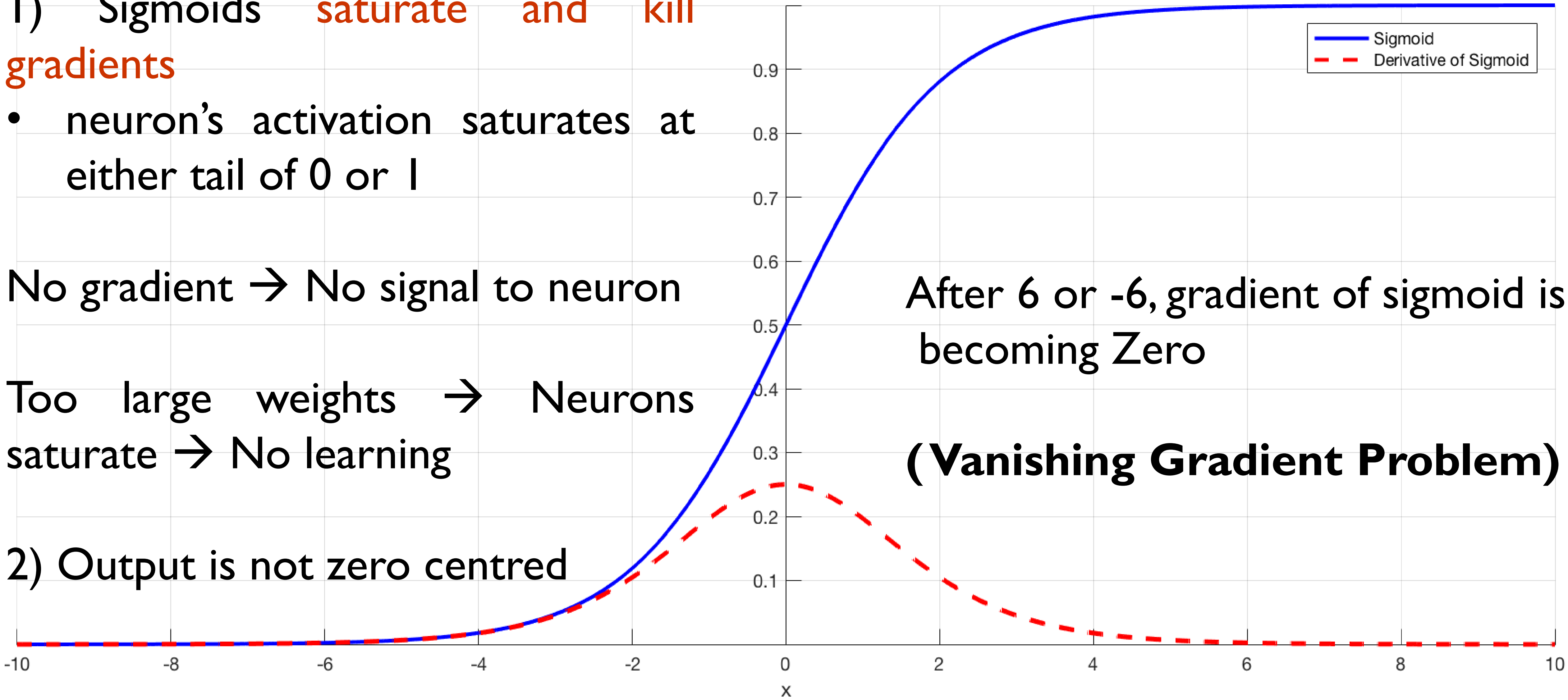
No gradient → No signal to neuron

Too large weights → Neurons saturate → No learning

2) Output is not zero centred

After 6 or -6, gradient of sigmoid is becoming Zero

(Vanishing Gradient Problem)



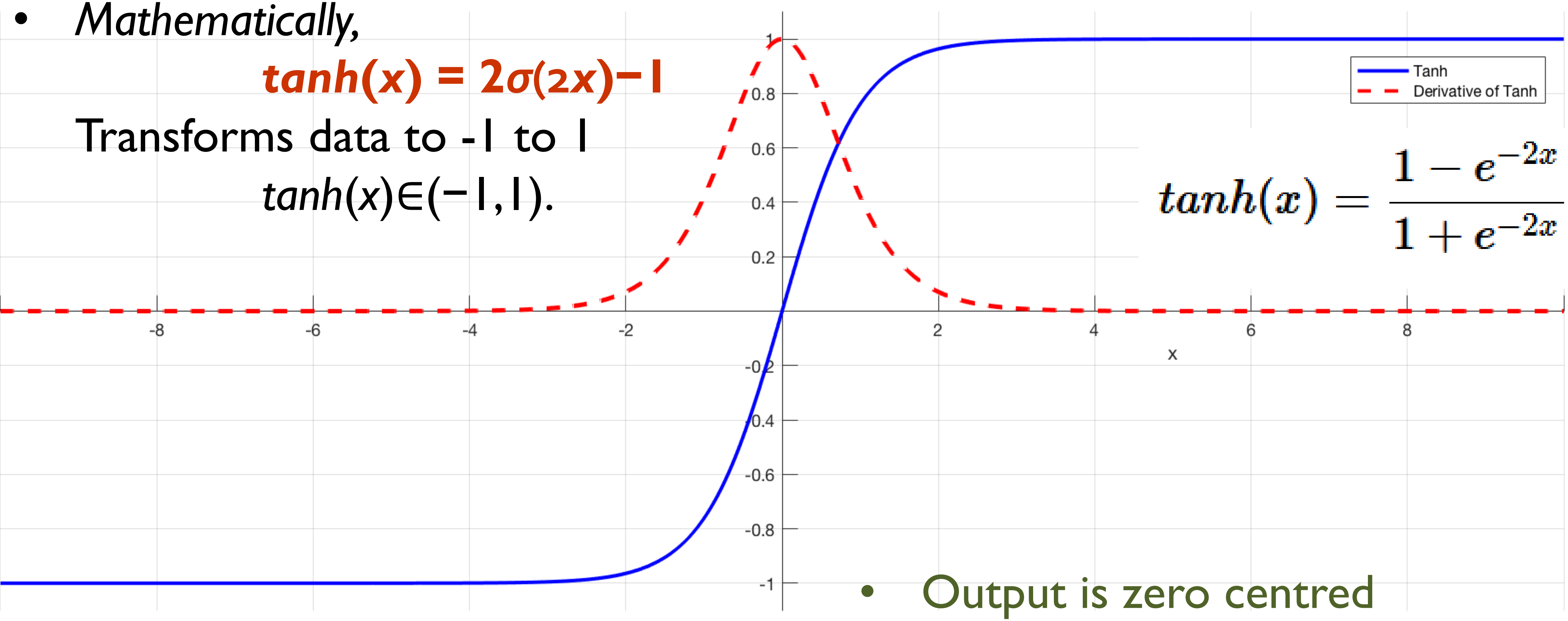
Hyperbolic Tangent function - Tanh

- It is also sigmoidal
- *Mathematically,*

$$tanh(x) = 2\sigma(2x) - 1$$

Transforms data to -1 to 1

$$tanh(x) \in (-1, 1).$$

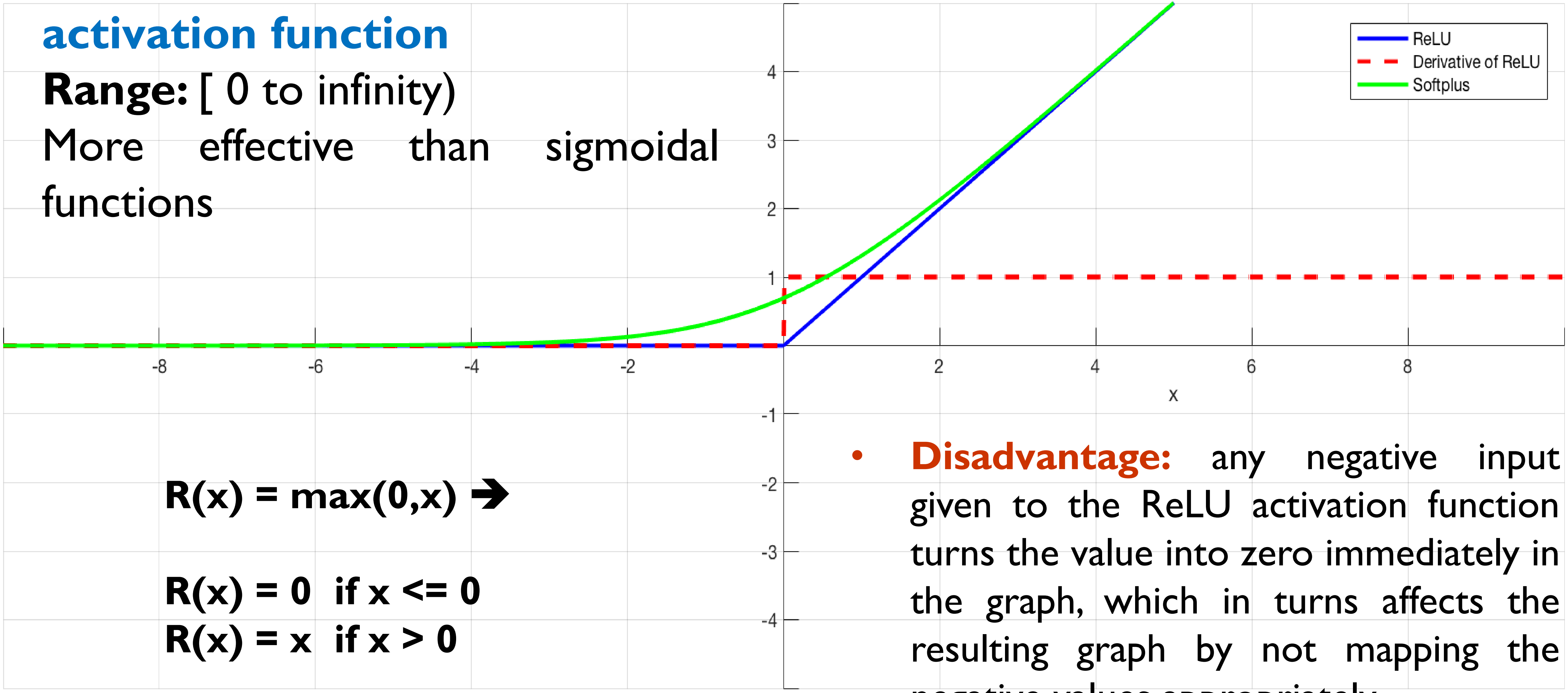


$$tanh(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}}$$

- Output is zero centred
- It also saturates and kills gradient

ReLU (Rectified Linear Unit) Activation Function

- **The ReLU is the most used activation function**
- **Range:** [0 to infinity)
- More effective than sigmoidal functions

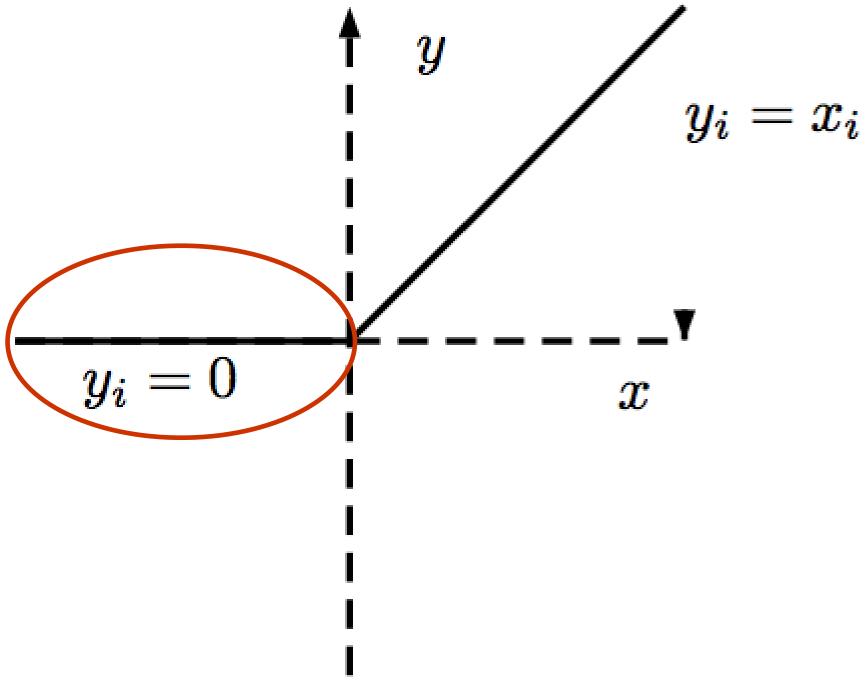


R(x) = max(0,x) ➔

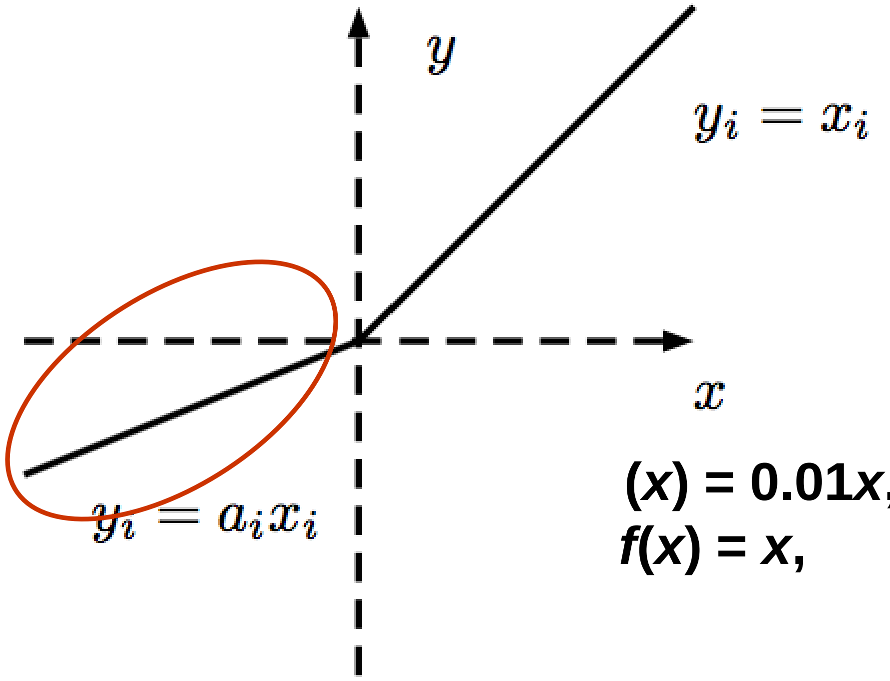
R(x) = 0 if x <= 0
R(x) = x if x > 0

- **Disadvantage:** any negative input given to the ReLU activation function turns the value into zero immediately in the graph, which in turns affects the resulting graph by not mapping the negative values appropriately.

Leaky Relu



ReLU



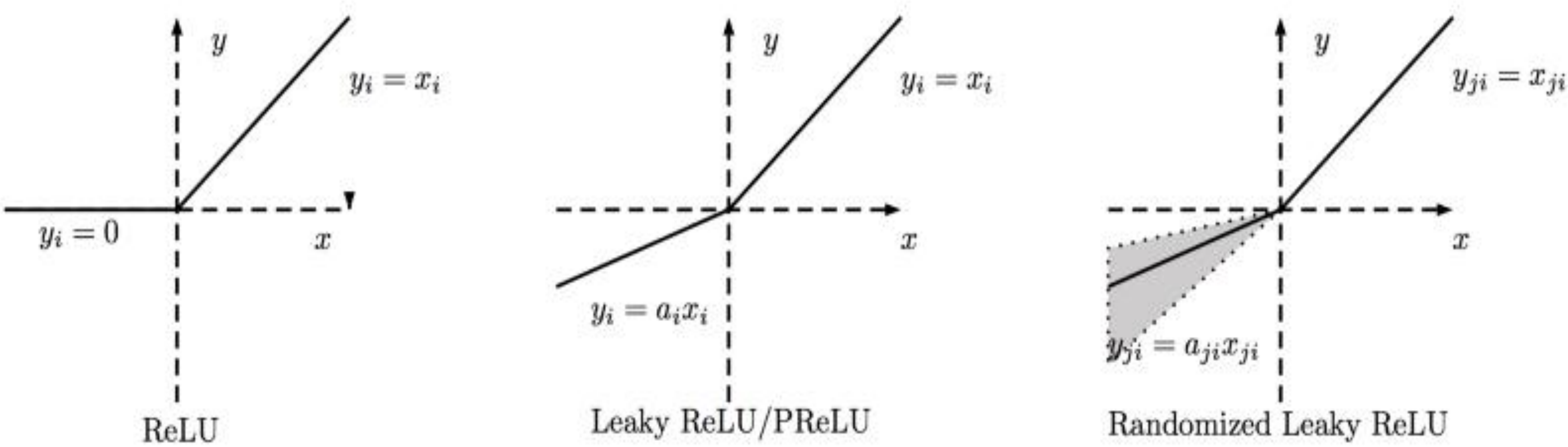
Leaky ReLU/PReLU

(*x*) = 0.01*x*, if (*x*<0)
***f*(*x*) = *x*, if (*x*≥0)**

fix the "dying ReLU" problem by
introducing a
Small negative slope (0.01 or so)

Randomized RELU

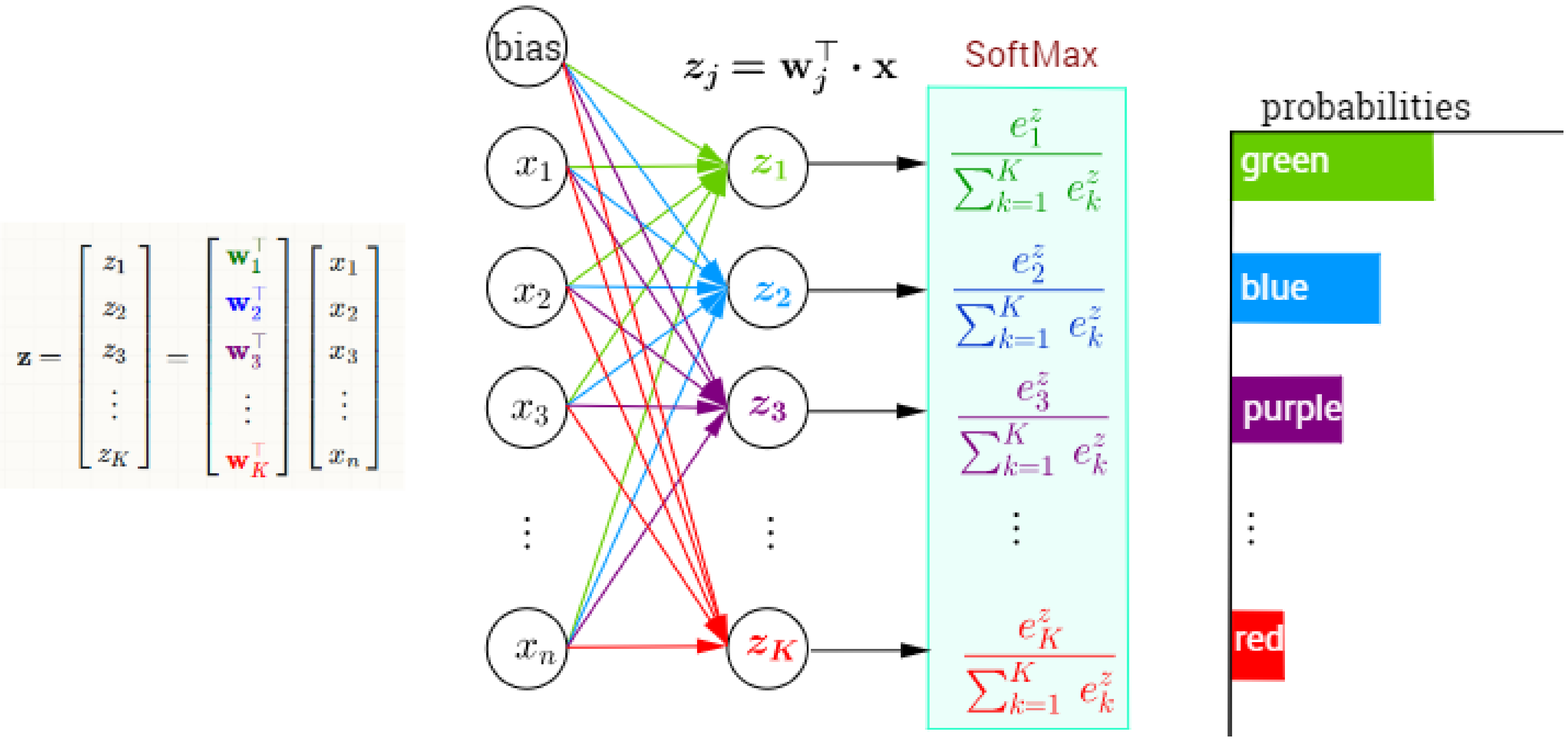
- Fix the problem of dying neurons
- It introduces a small slope to keep the updates alive.
- The leak helps to increase the range of the ReLU function. Usually, the value of **a** is 0.01 or so.
- When **a is not 0.01** then it is called **Randomized ReLU**.



Softmax

- Transforms a real valued vector to [0, 1]
- Output of softmax is the list of probabilities of k different classes/categories

Multi-Class Classification with NN and SoftMax Function

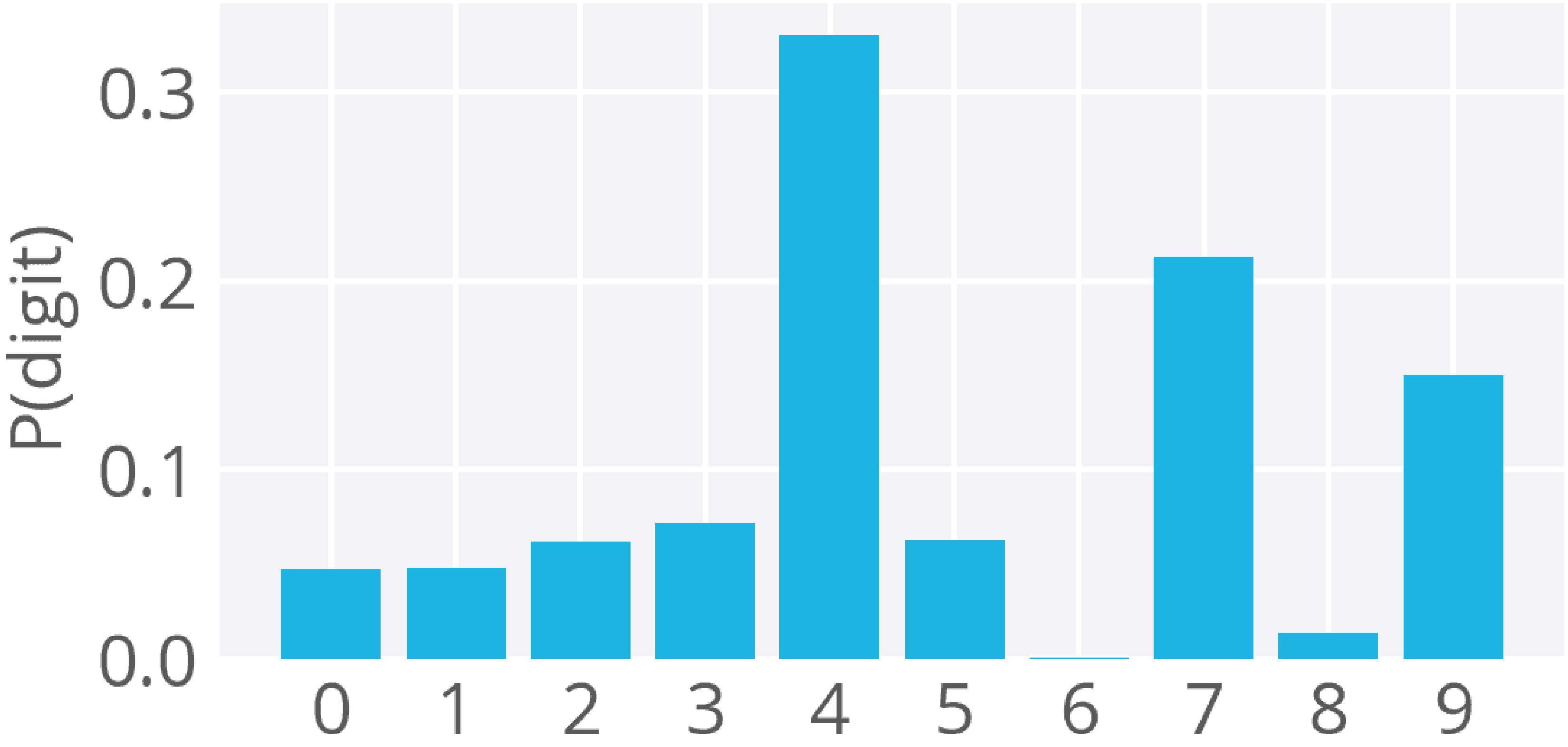


$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$

j = 1, 2, ..., K.

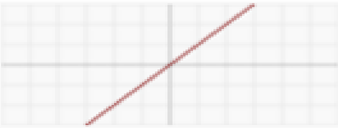
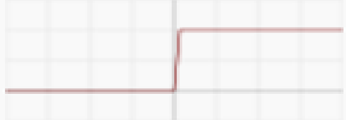
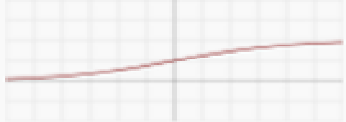
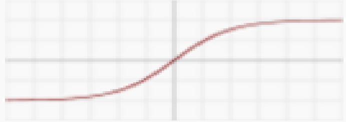
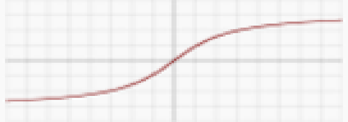
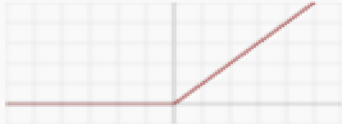
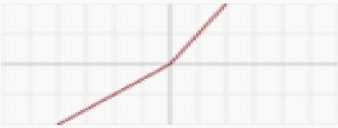
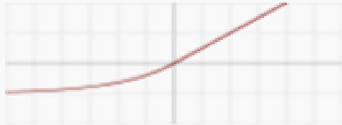
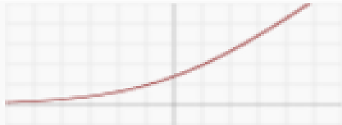
Softmax – Classification of digits

- Transforms a real valued vector to [0, 1]
- Output of softmax is the list of probabilities of 10 digits



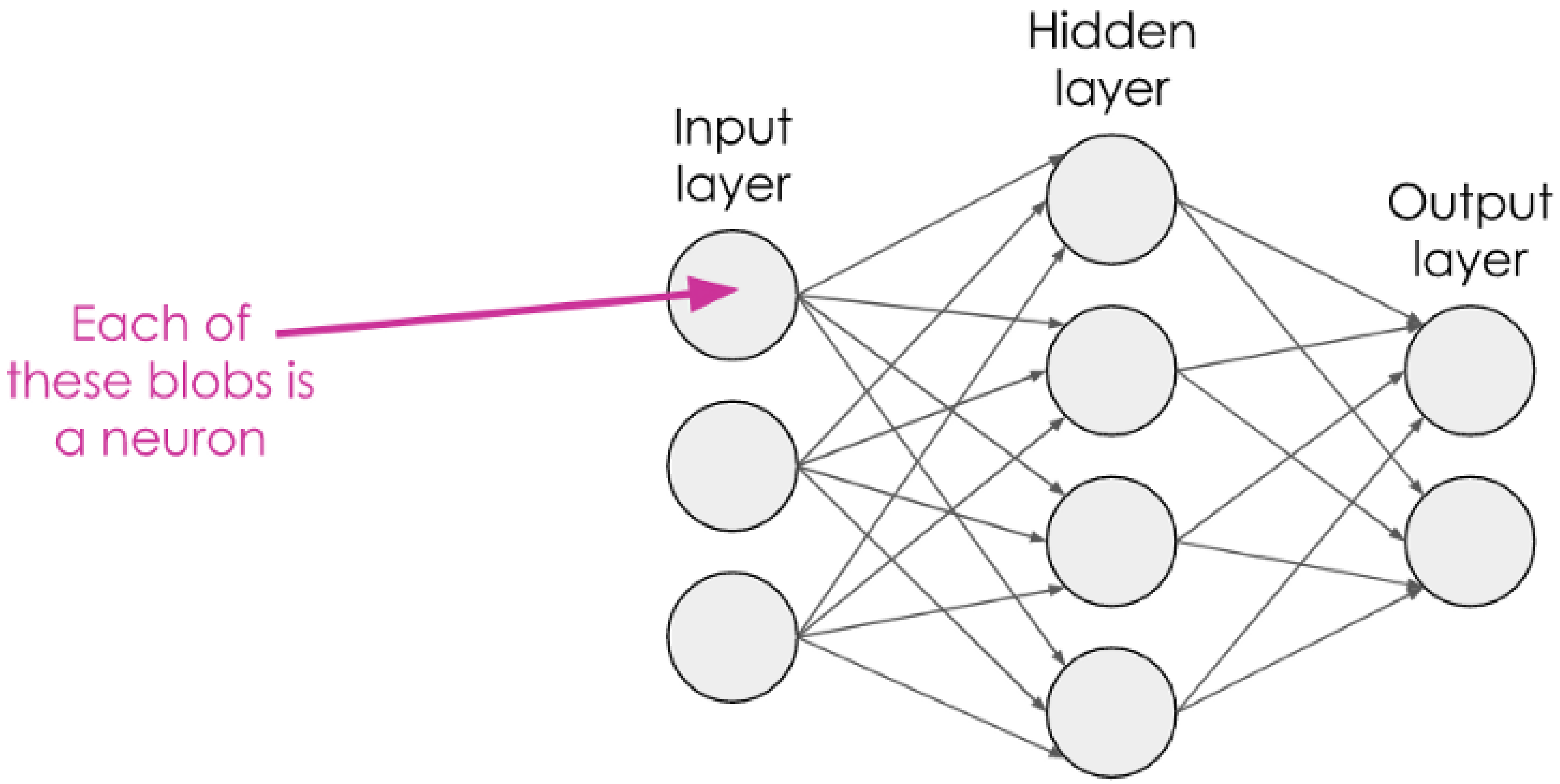
Which activation function to use?

- use **ReLu** which should only be applied to the hidden layers.
- And if our Model suffers form dead neurons during training we should use **leaky ReLu**.
- It's just that *Sigmoid and Tanh* should not be used nowadays due to the **vanishing Gradient Problem** which causes a lots of problems to train a Neural Network Model.
- **Softmax** is used in the last layer (Because it gives output probabilities).

Name	Plot	Equation	Derivative
Identity		$f(x) = x$	$f'(x) = 1$
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$
Logistic (a.k.a Soft step)		$f(x) = \frac{1}{1+e^{-x}}$	$f'(x) = f(x)(1 - f(x))$
TanH		$f(x) = \tanh(x) = \frac{2}{1+e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2+1}$
Rectified Linear Unit (ReLU)		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Parameteric Rectified Linear Unit (PReLU)[2]		$f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Exponential Linear Unit (ELU)[3]		$f(x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} f(x) + \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
SoftPlus		$f(x) = \log_e(1 + e^x)$	$f'(x) = \frac{1}{1+e^{-x}}$

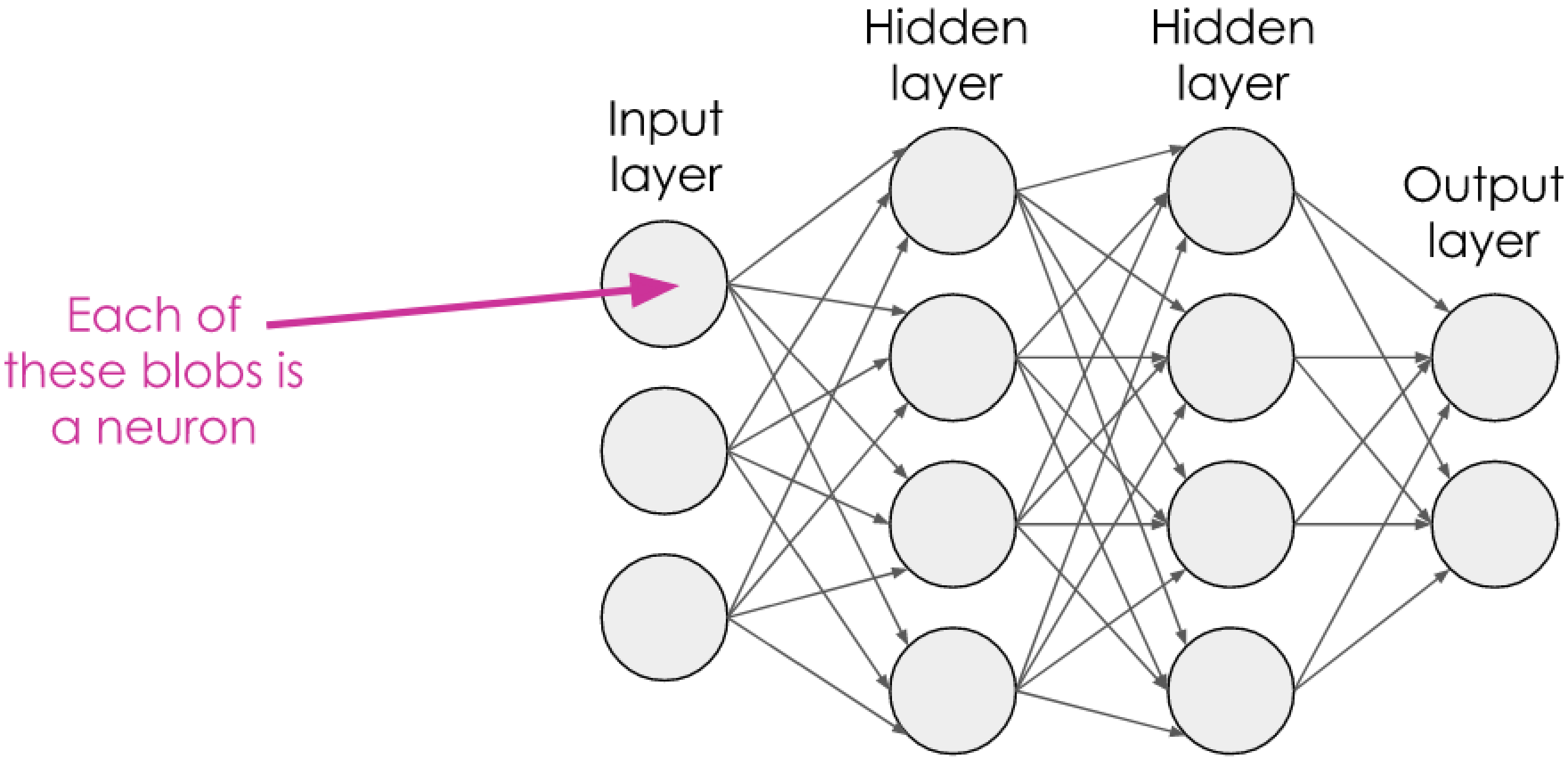
A Neuron to Deep Networks…

Single neuron to a network



- The outputs of the neurons in one layer flow through to become the inputs of the next layer, and so on.

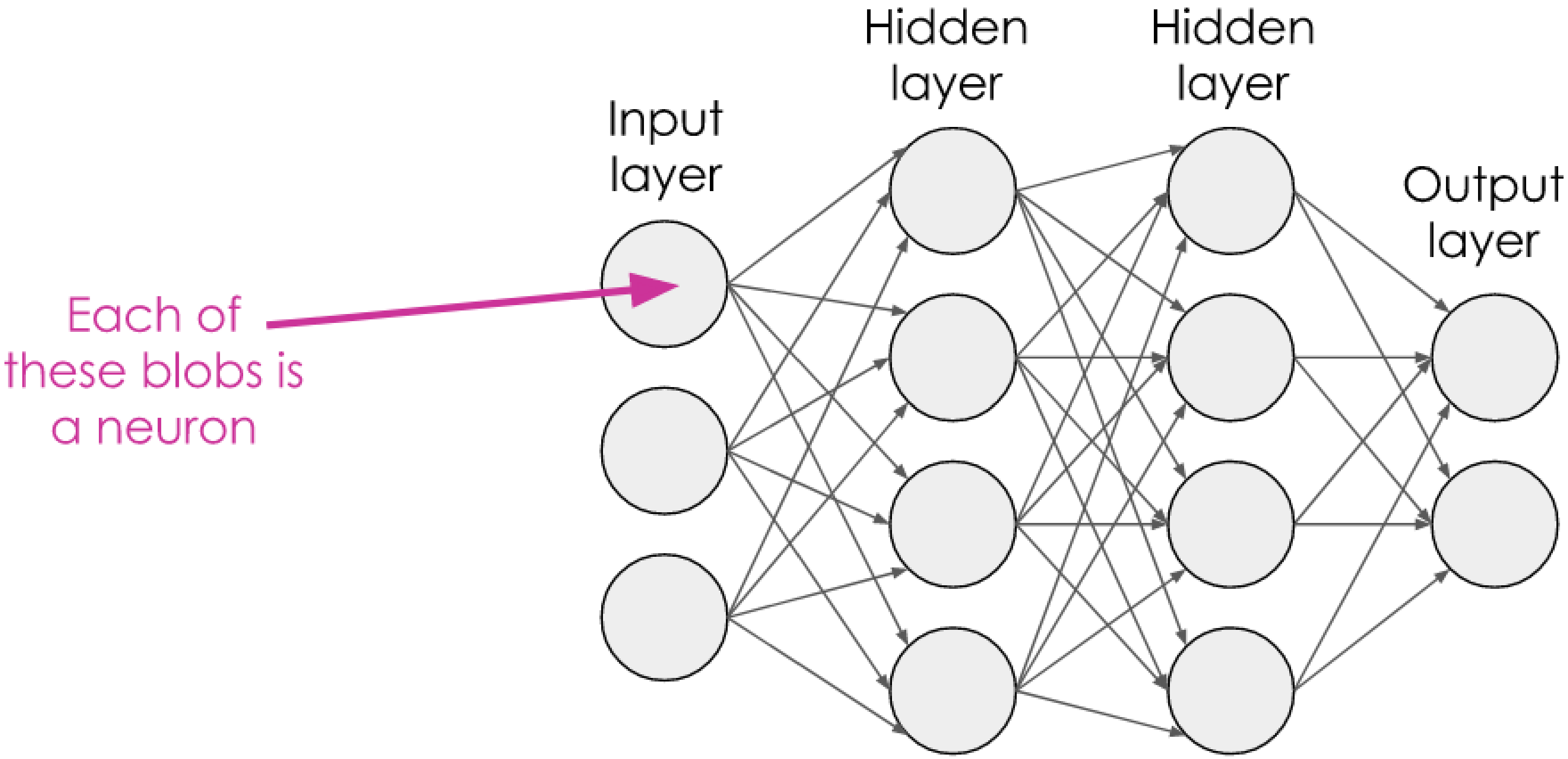
A network to a "deep network"



Add another hidden layer….

a neural network can be considered 'deep' if it contains more hidden layers...

Feedforward Neural Network

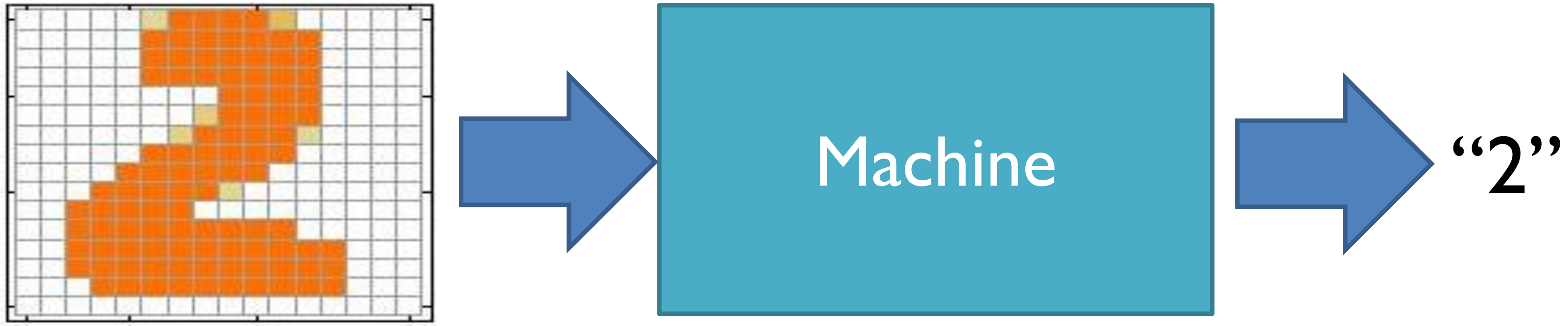


Add another hidden layer….

a neural network can be considered 'deep' if it contains more hidden layers...

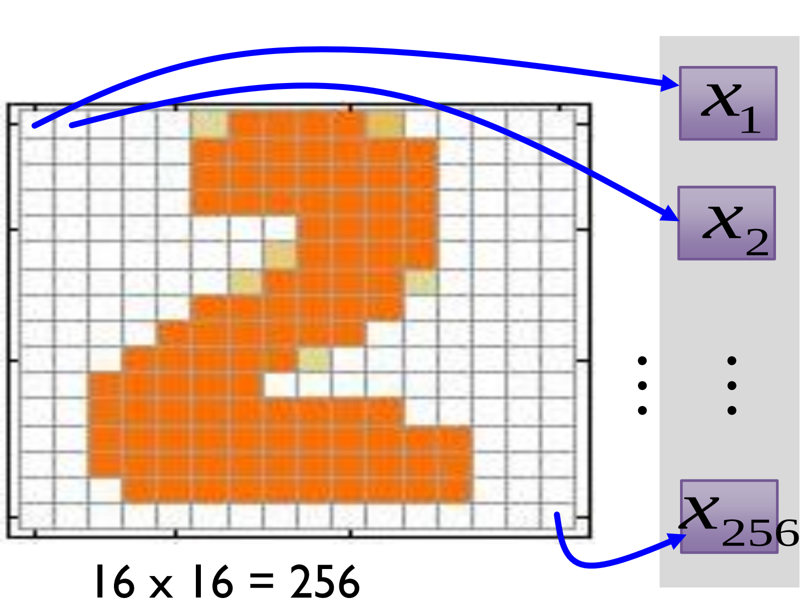
Deep Network - Example Application

- Handwriting Digit Recognition



Deep Network - Example Application

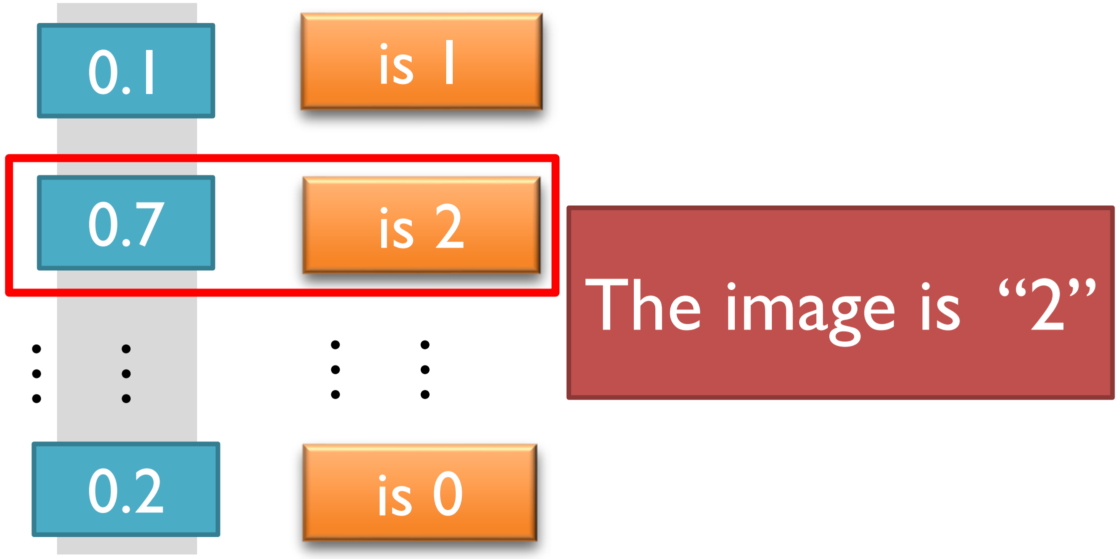
- Handwriting Digit Recognition



16 x 16 = 256

Ink → 1

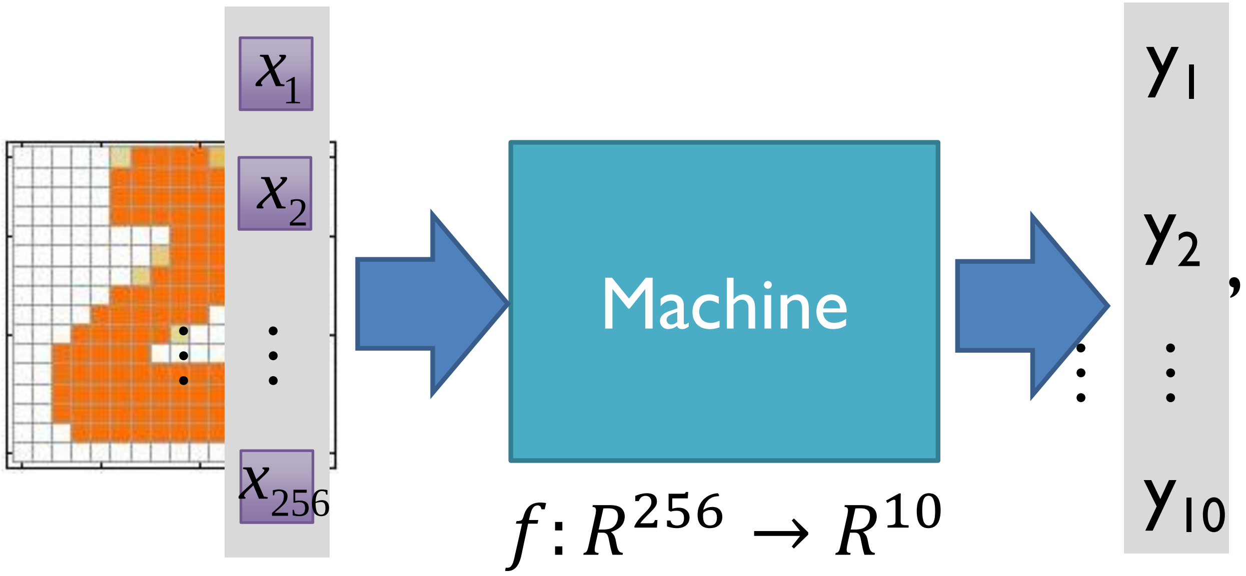
No ink → 0



Each dimension represents the confidence of a digit.

Deep Network - Example Application

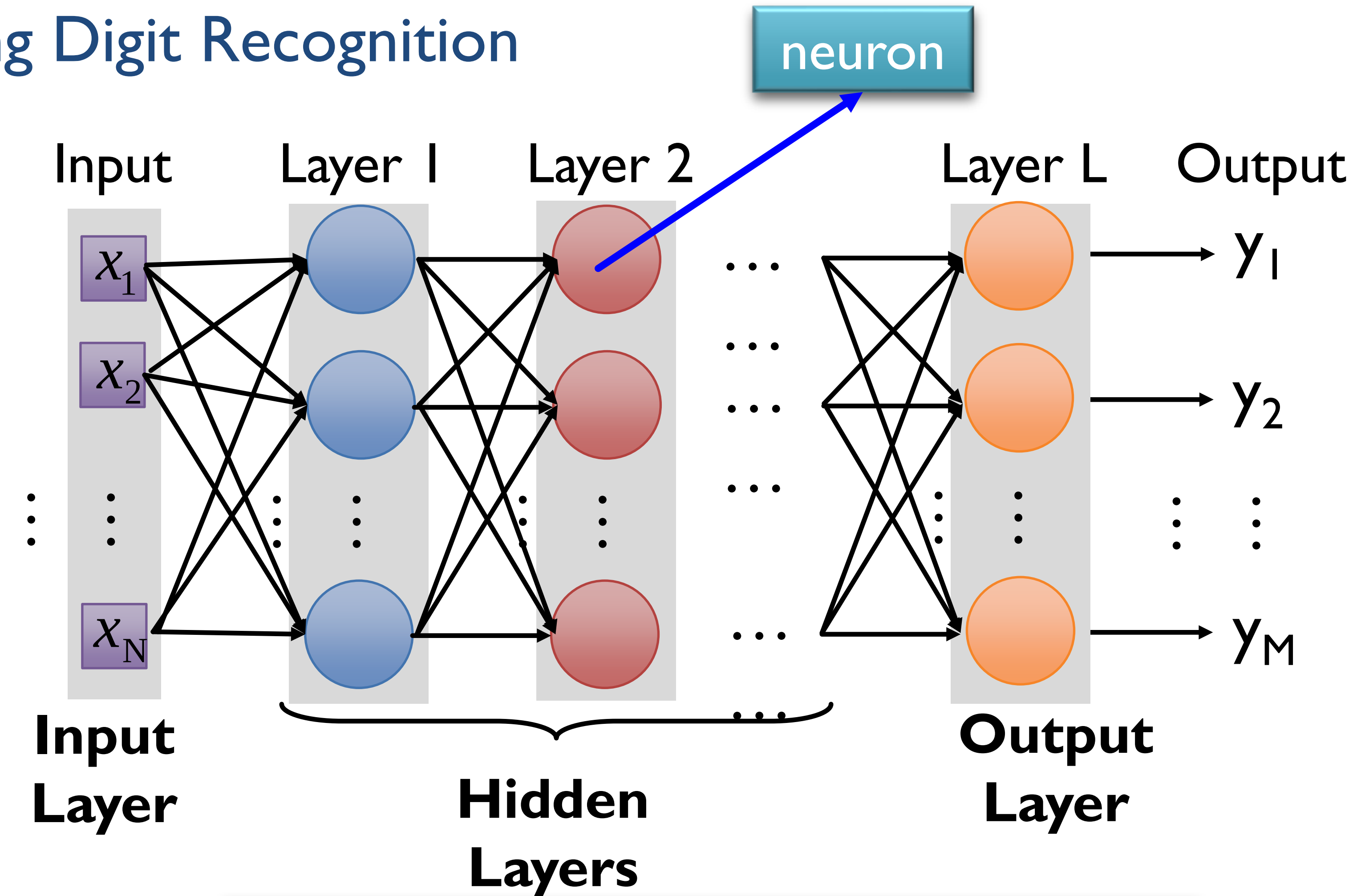
- Handwriting Digit Recognition



In deep learning, the function f is represented by neural network

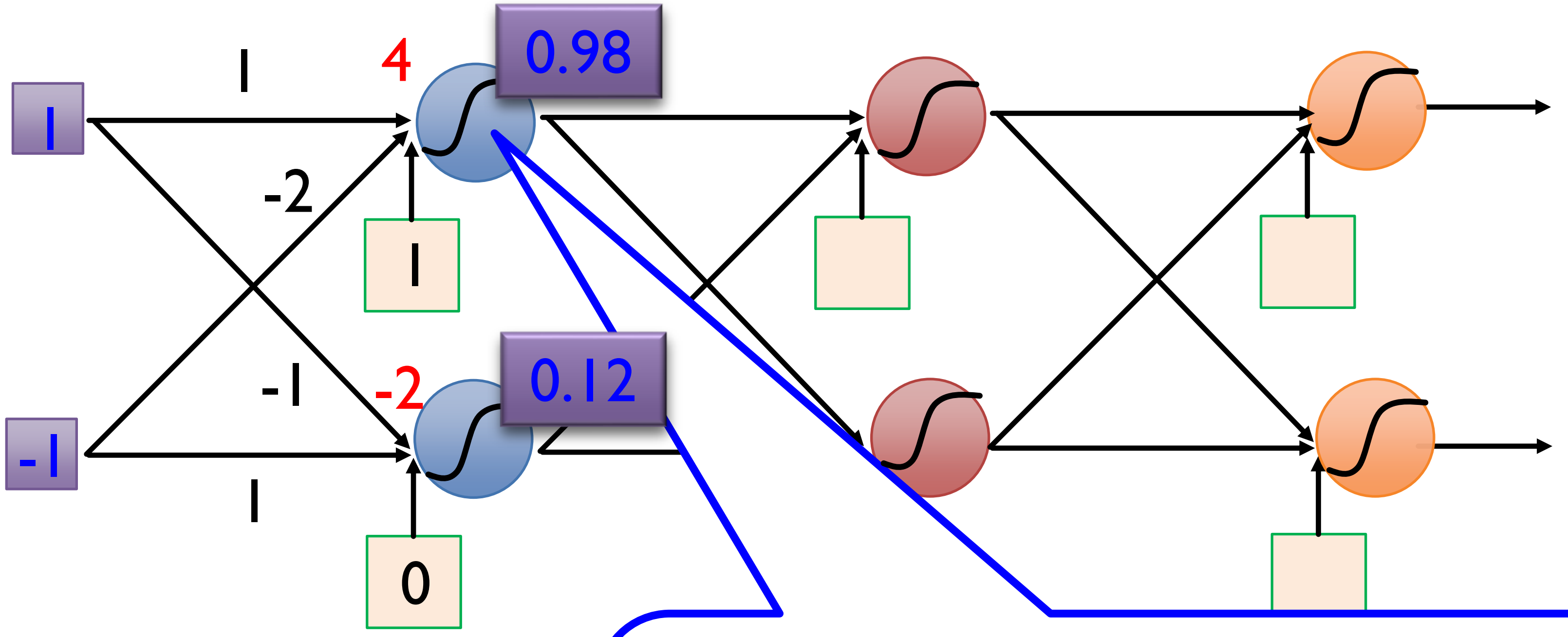
Deep Network – Representation

- Handwriting Digit Recognition



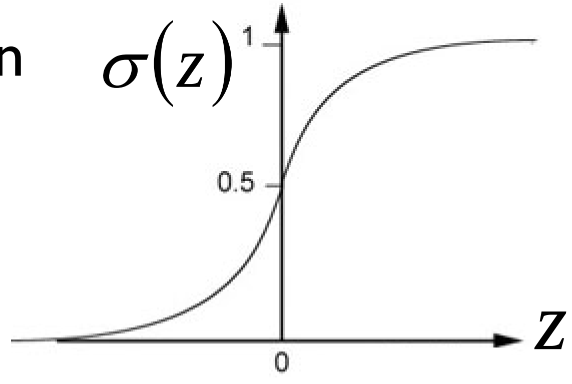
Deep means many hidden layers

Deep Network – Representation

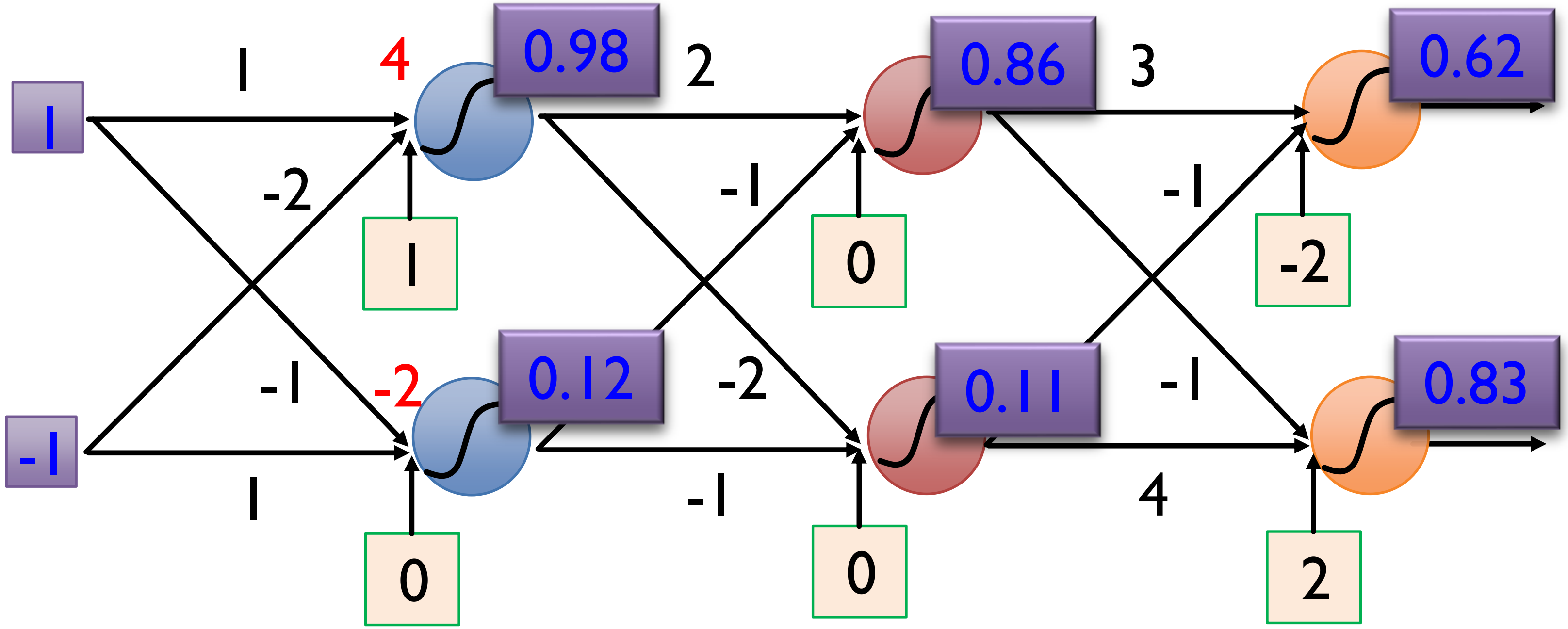


Sigmoid Function

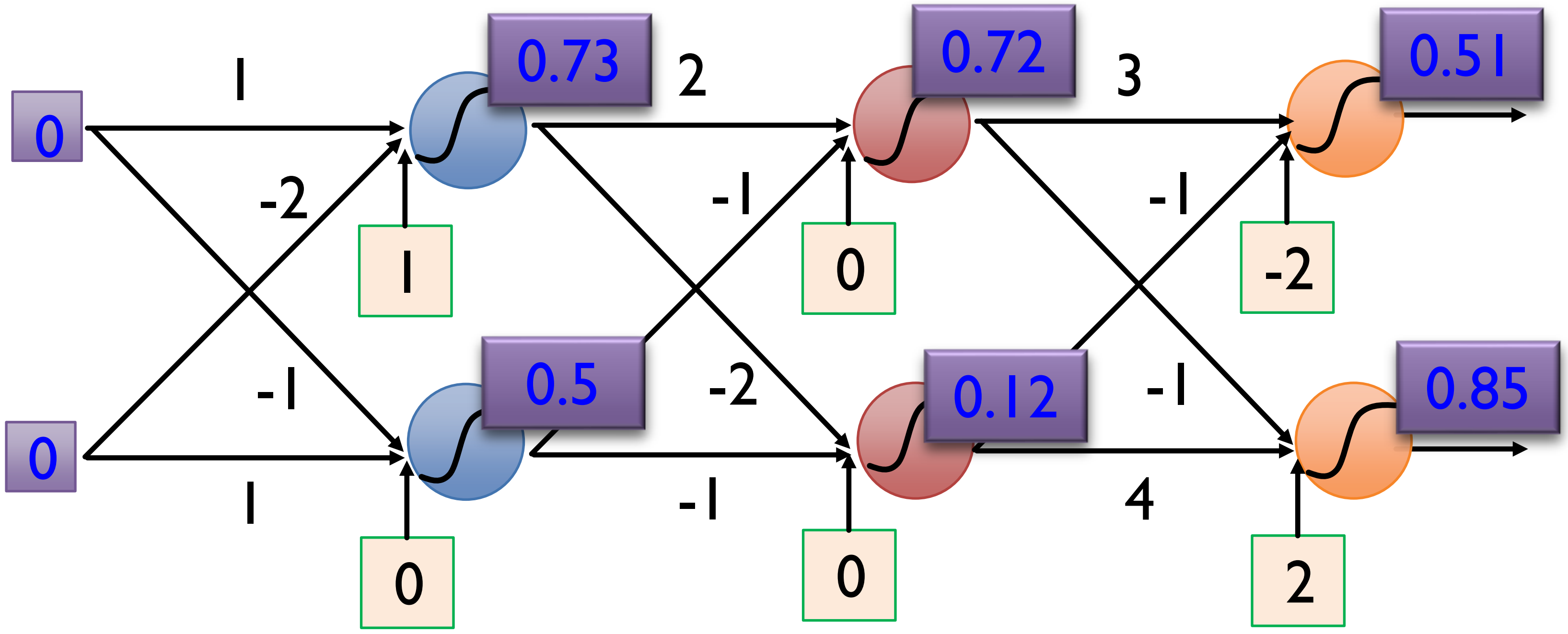
$$\sigma(z) = \frac{1}{1+e^{-z}}$$



Deep Network – Representation



Deep Network – Representation



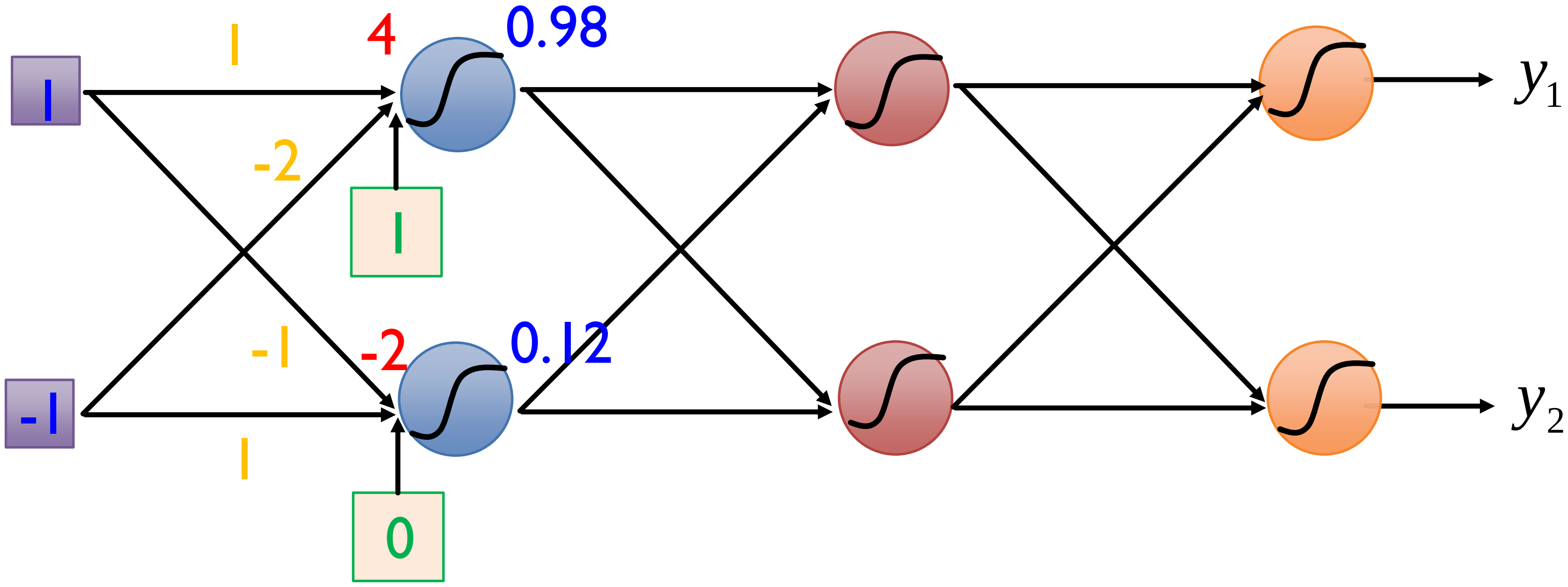
$$f: R^2 \rightarrow R^2$$

$$f\left(\begin{bmatrix} 1 \\ -1 \end{bmatrix}\right) = \begin{bmatrix} 0.62 \\ 0.83 \end{bmatrix} \quad f\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}\right) = \begin{bmatrix} 0.51 \\ 0.85 \end{bmatrix}$$

Different parameters define different function

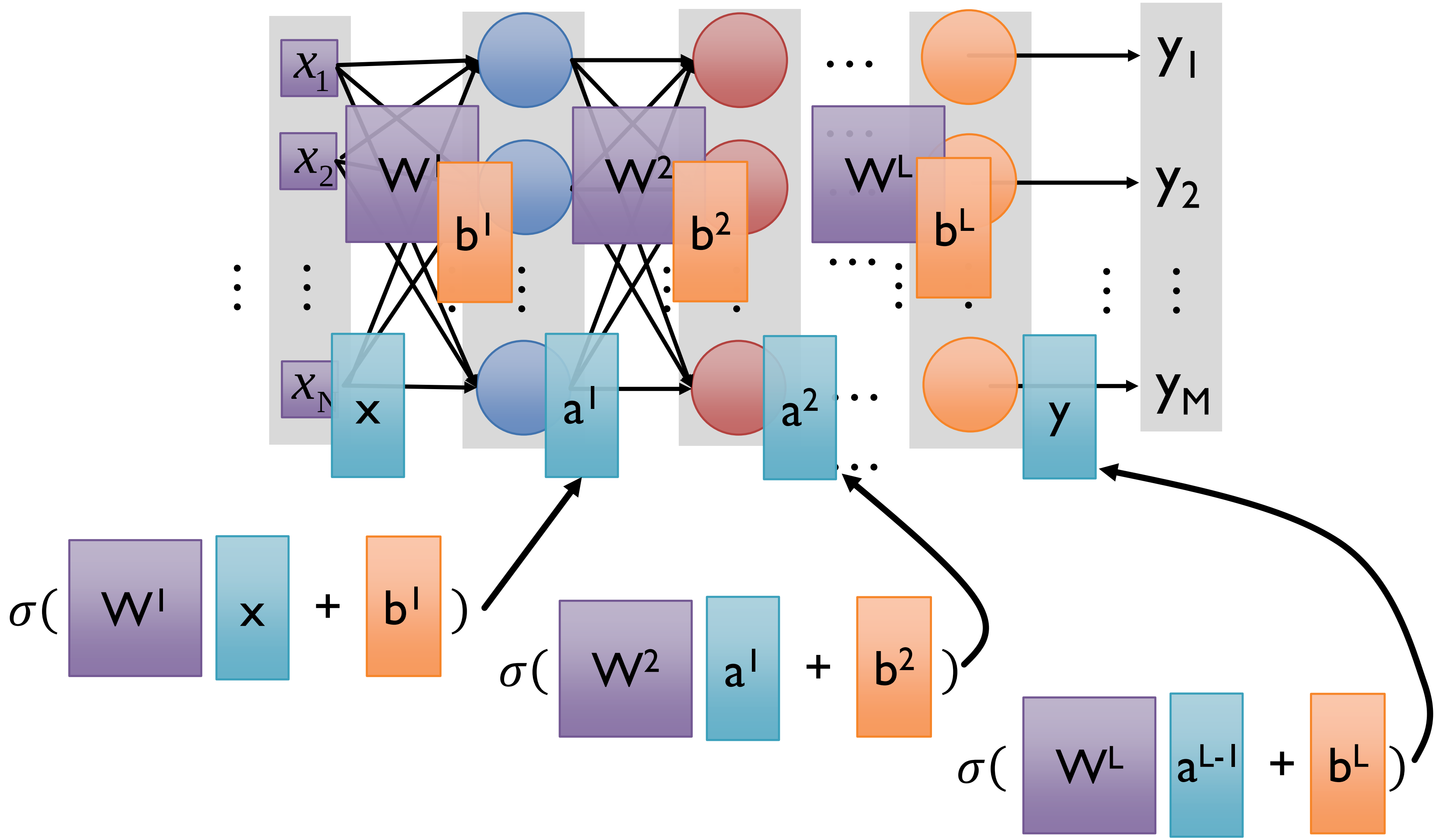
Deep Network – Representation

- Matrix Notation (A NumPy Array)

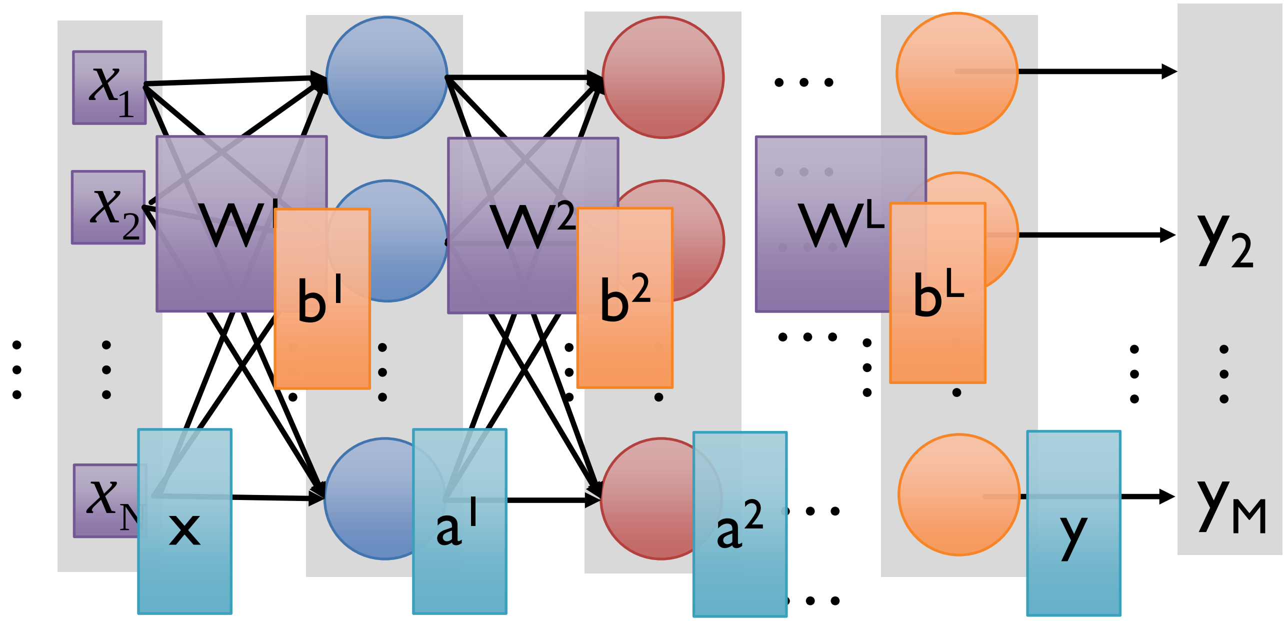


$$\sigma\Bigg(\underbrace{\begin{bmatrix} 1 & -2 \\ -1 & 1 \end{bmatrix}\begin{bmatrix} 1 \\ -1 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix}}_{\begin{bmatrix} 4 \\ -2 \end{bmatrix}}\Bigg) = \begin{bmatrix} 0.98 \\ 0.12 \end{bmatrix}$$

Deep Network – Representation



Deep Network – Representation



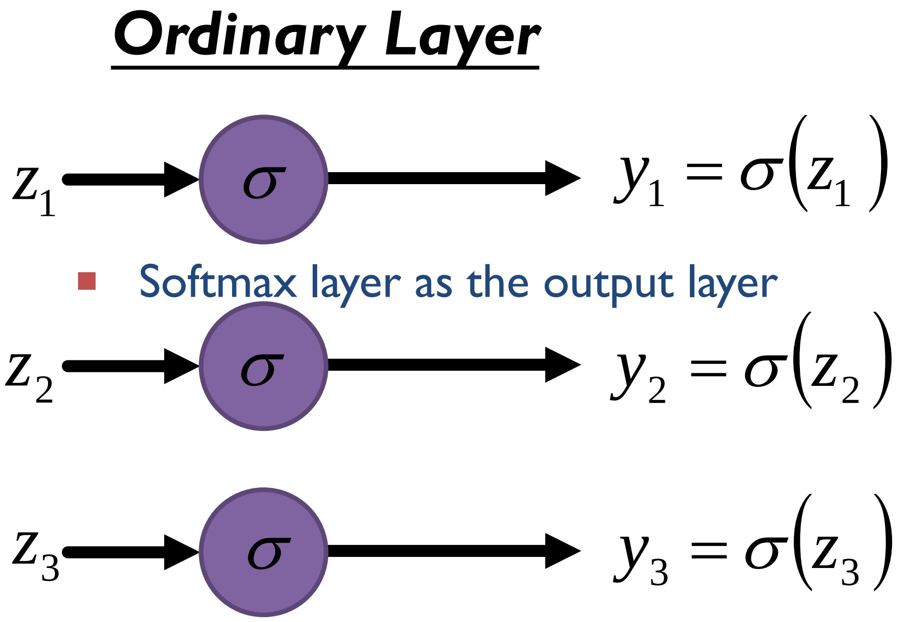
$$y = f(x)$$

Using parallel computing techniques to speed up matrix operation

$$= \sigma(W^L \cdots \sigma(W^2 \, \sigma(W^1 \, x + b^1) + b^2) \cdots + b^L)$$

Deep Network – Representation – Output Layer

Ordinary Layer as the output layer



- Softmax layer as the output layer

In general, the output of network can be any value.

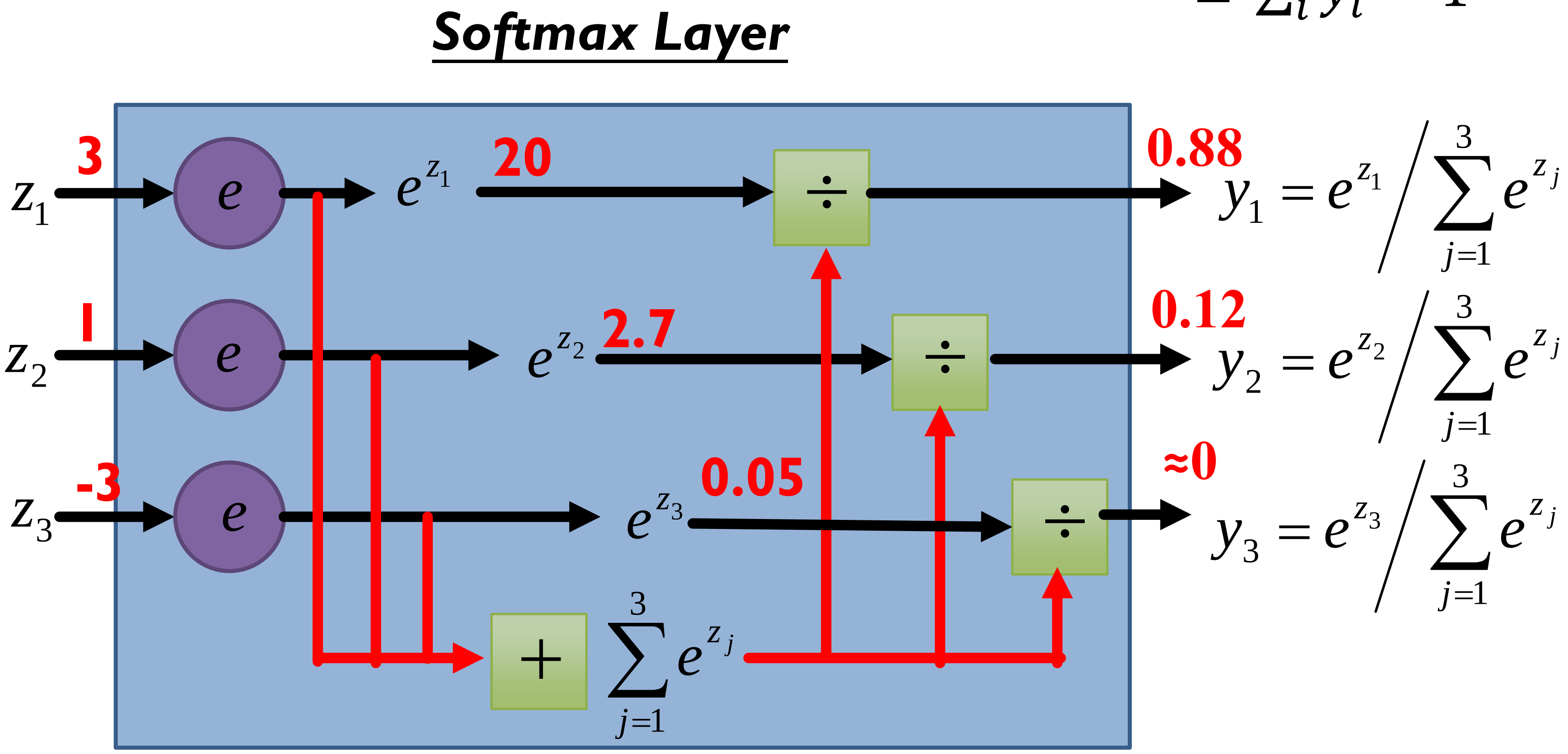
May not be easy to interpret

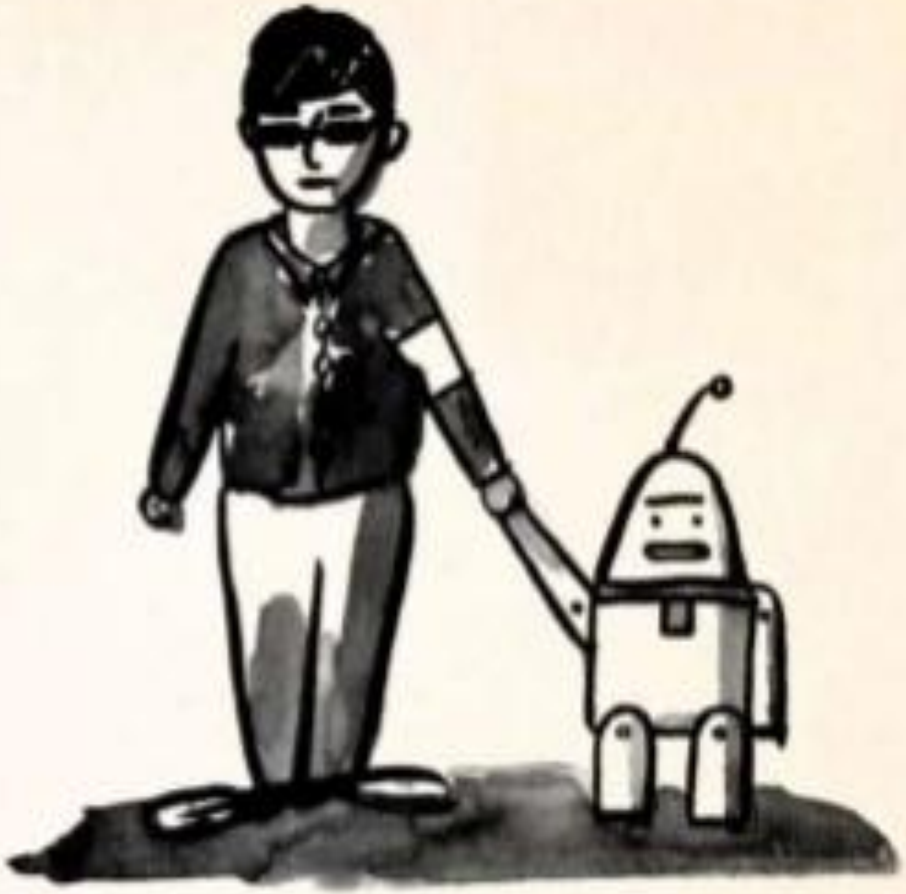
Deep Network – Representation – Output Layer

Softmax Layer (As the output Layer)

Probability:

- $1 > y_i > 0$
- $\sum_i y_i = 1$





THANK YOU!

QUESTIONS?