الجامعة السورية الخاصة
SYRIAN PRIVATE UNIVERSITY

الذكاء الصنعي العملي

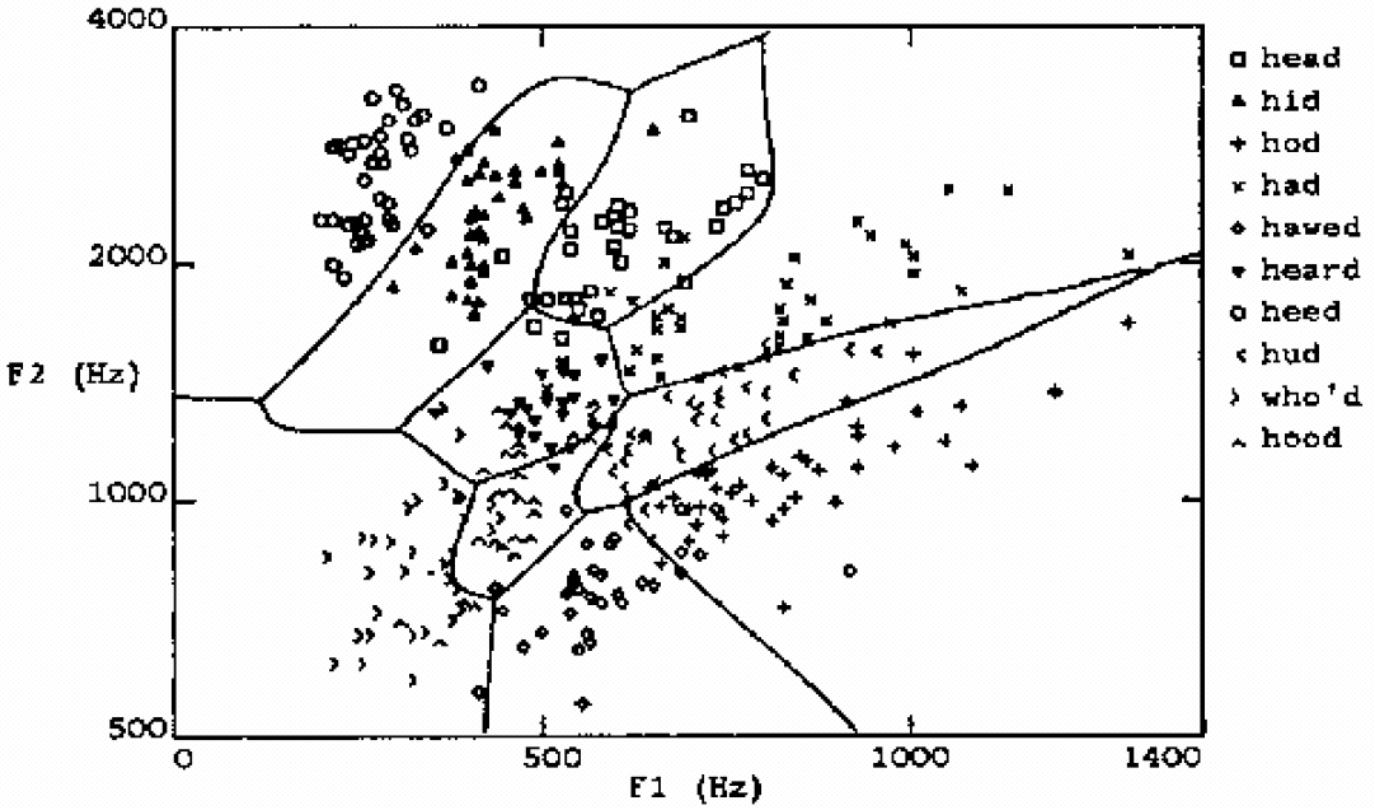
المحاضرة الأولى

مقدمة إلى التعلّم العميق
Introduction to Deep Learning

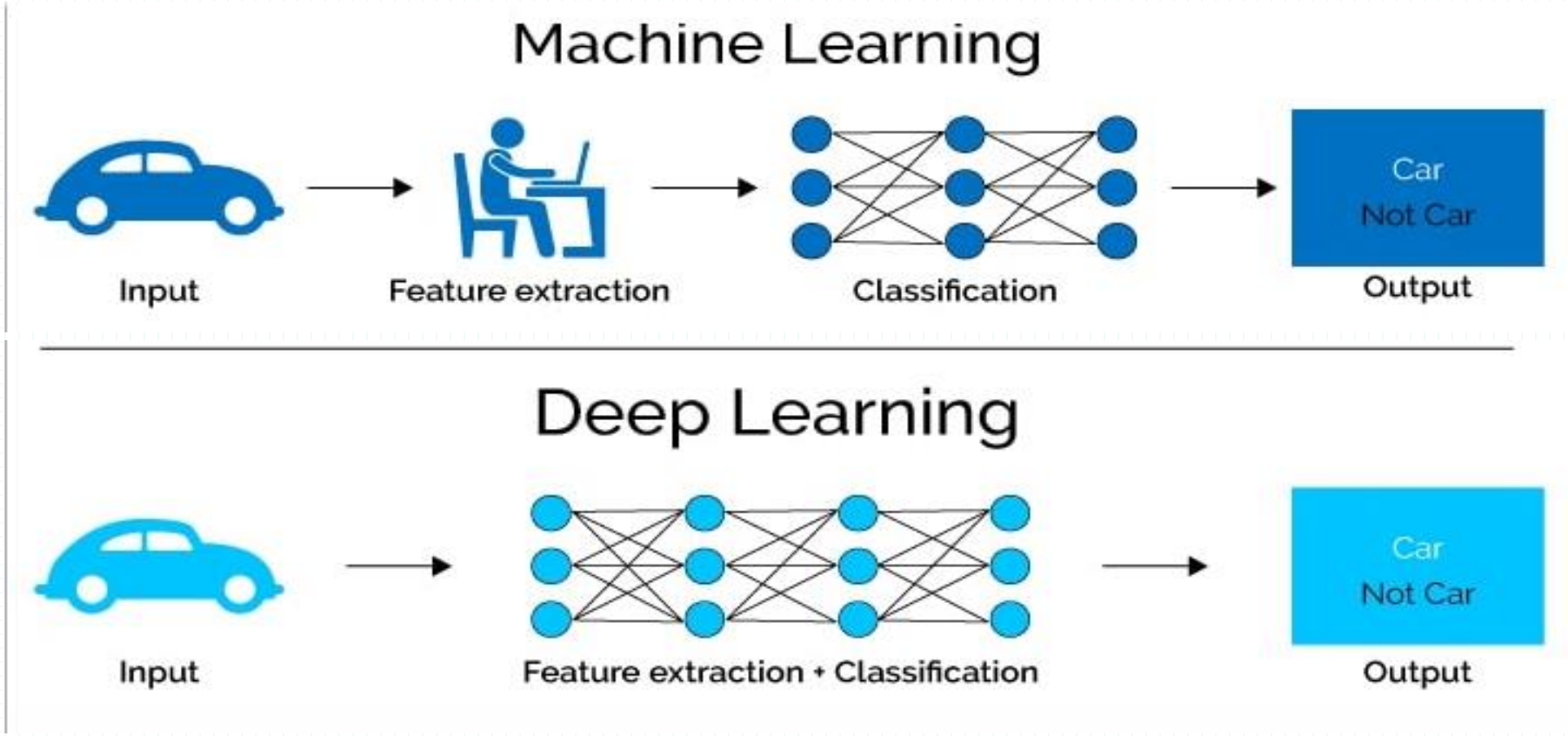
د. رياض سنبل

Why Deep Learning?

- What kind of discussion boundaries can we learn using:
  - Decision Trees,
  - KNN,
  - SVM
- Learning highly non-linear functions



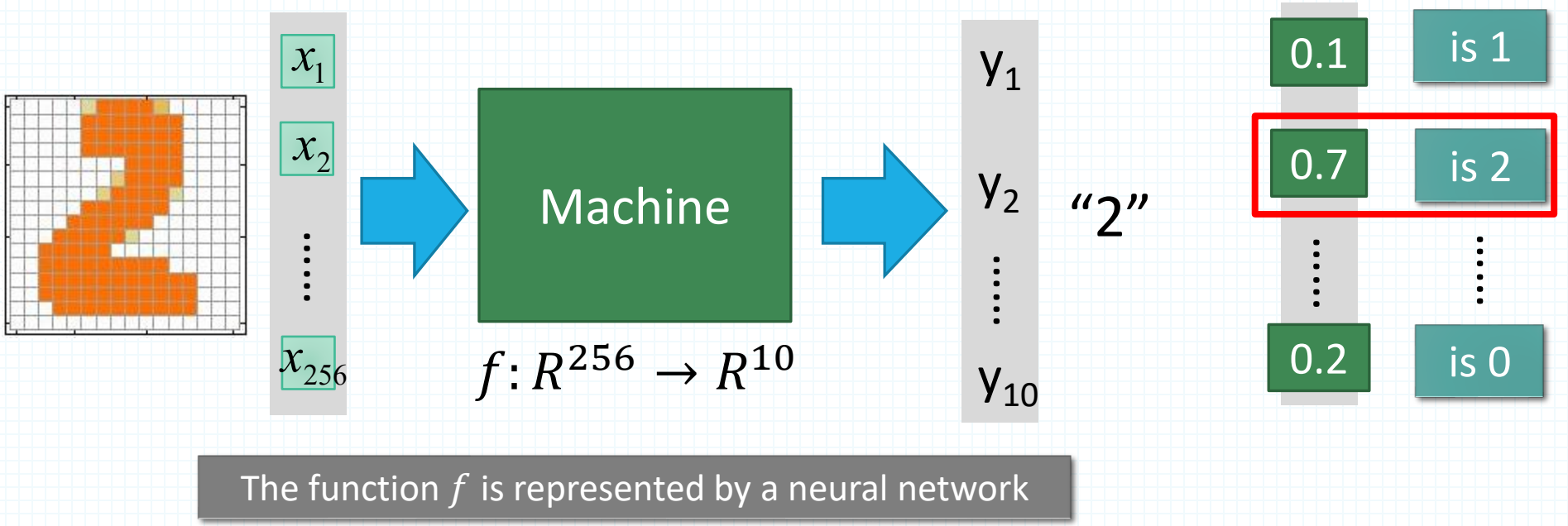
What is Deep Learning?



can we learn underlying features directly from data

Neural Networks (NNs)

- **Input:** Raw data—such as images—is represented as a set of numerical features (e.g., pixel intensities in an image).
- **Output:** a predicted class label representing the category or type of the input data.



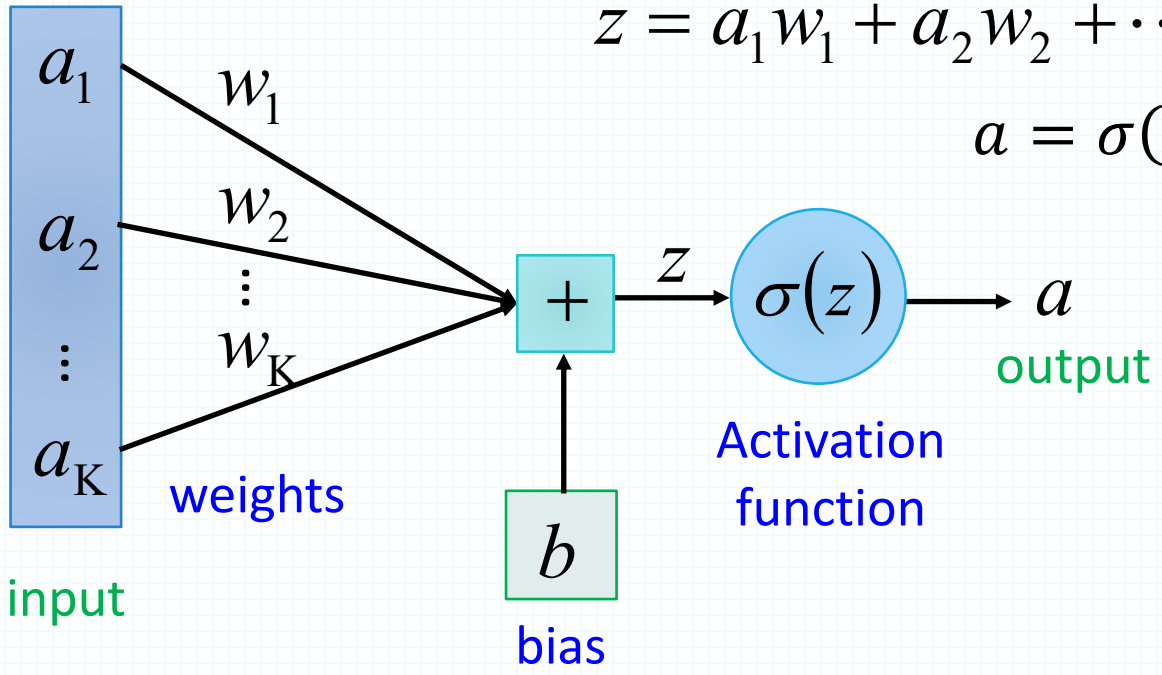
The function f is represented by a neural network

Elements of Neural Networks

- NNs consist of **hidden layers with neurons** (i.e., computational units)
- A single neuron maps a set of inputs into an output number, or $f: R^K \rightarrow R$

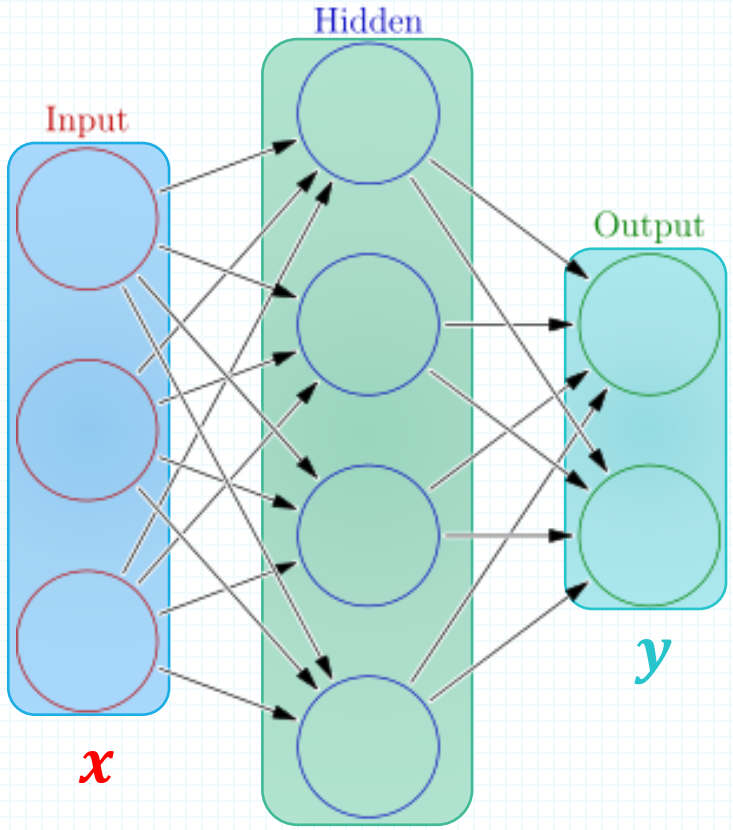
$$z = a_1 w_1 + a_2 w_2 + \cdots + a_K w_K + b$$

$$a = \sigma(z)$$



Elements of Neural Networks

- An NN with one hidden layer and one output layer



Weights

Biases

$$hidden\ layer\ h = \sigma(W_1 x + b_1)$$

$$output\ layer\ y = \sigma(W_2 h + b_2)$$

Activation functions

learnable parameters?

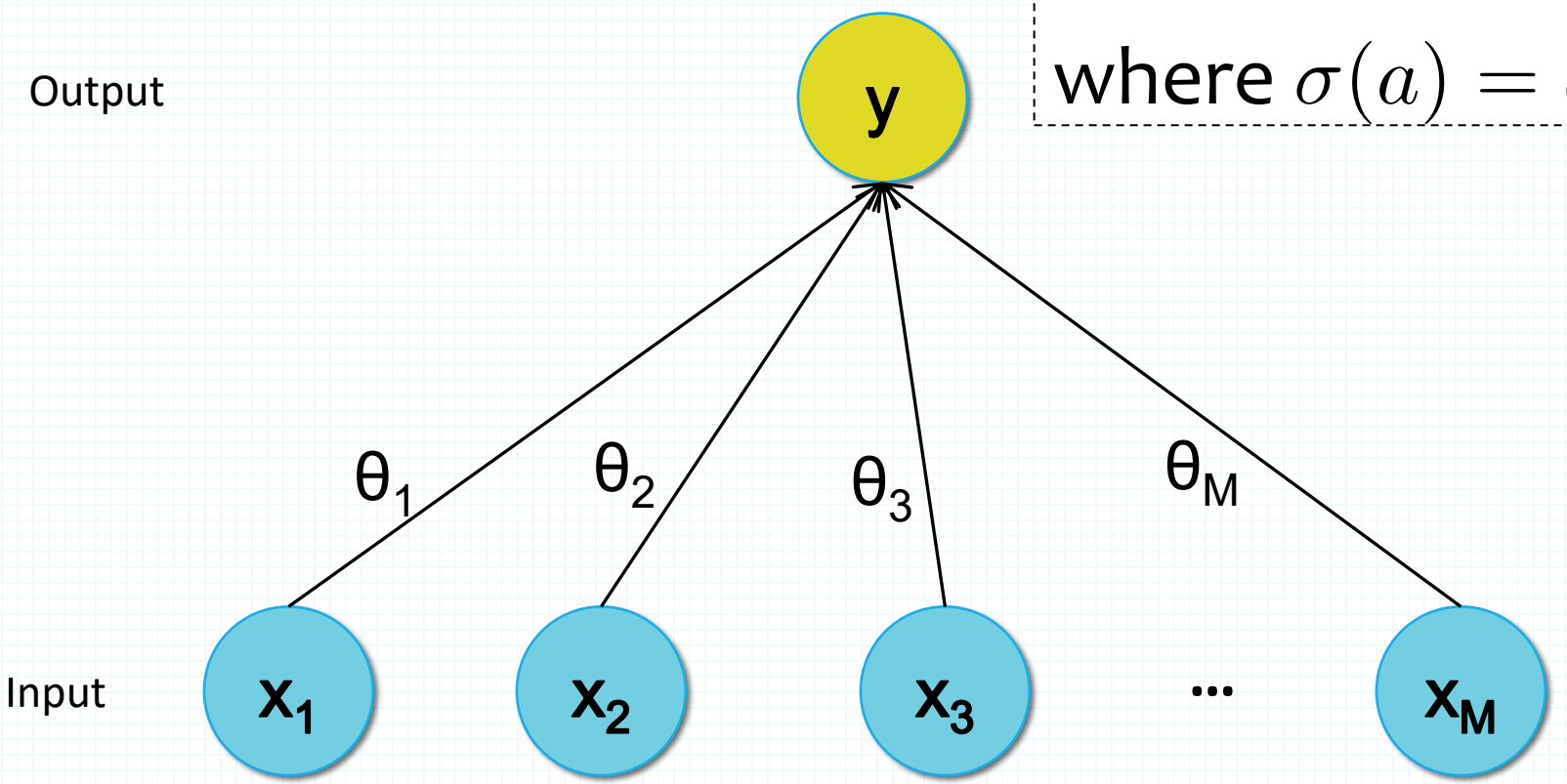
4 + 2 = 6 neurons (not counting inputs)
[3 × 4] + [4 × 2] = 20 weights
4 + 2 = 6 biases

26 learnable parameters

Linear Regression

$$y = h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sigma(\boldsymbol{\theta}^T \boldsymbol{x})$$

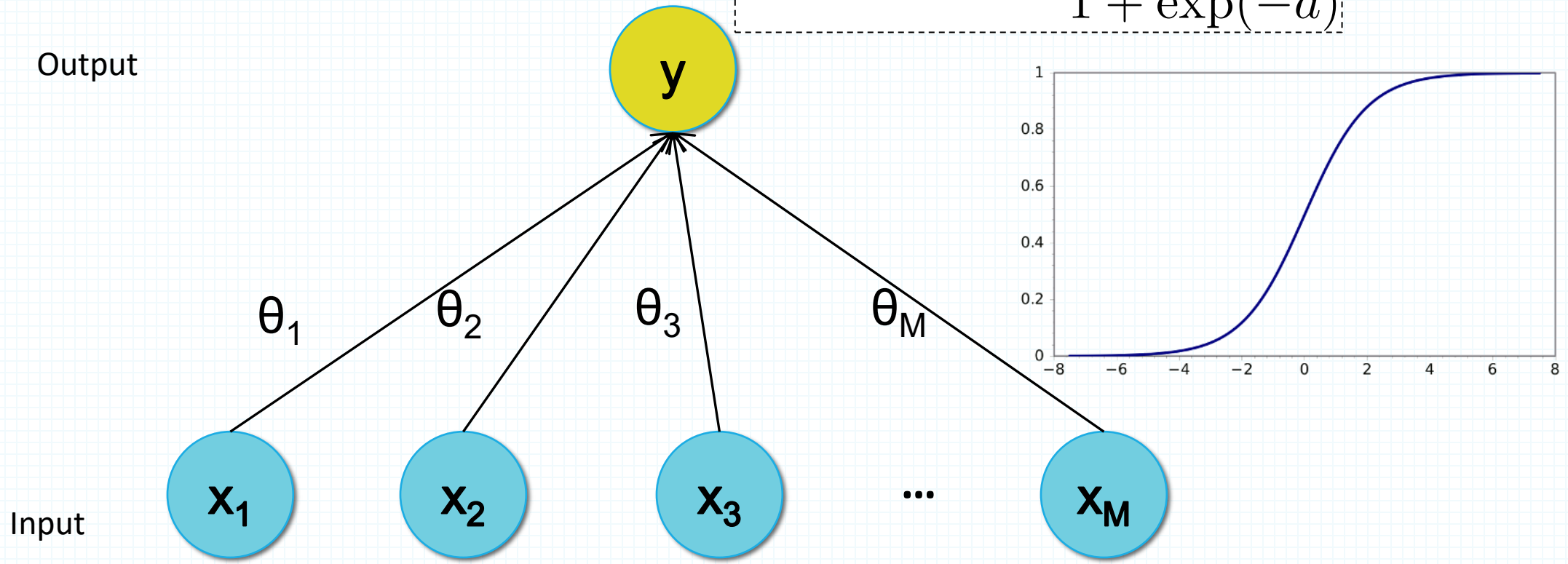
where $\sigma(a) = a$



Logistic Regression

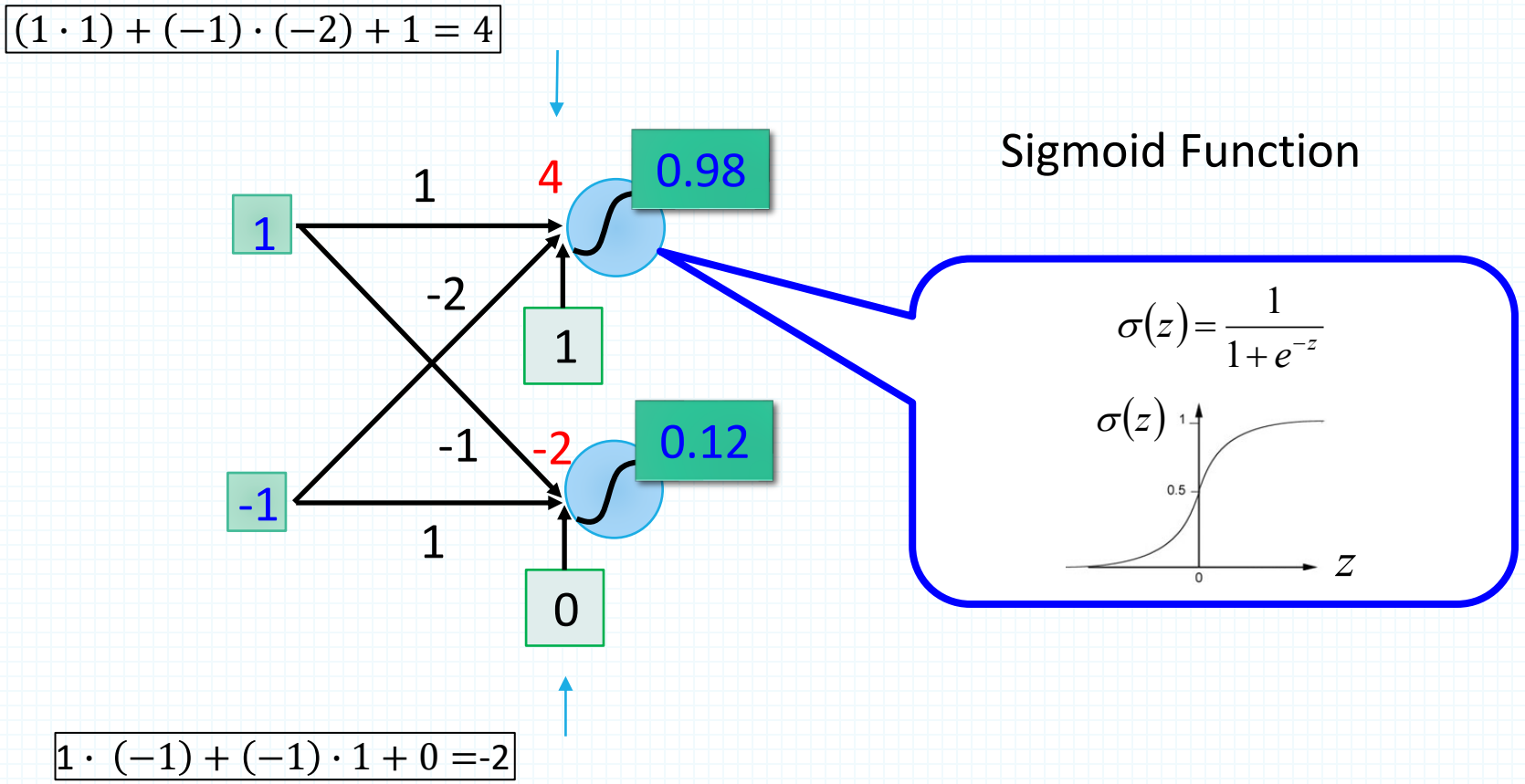
$$y = h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sigma(\boldsymbol{\theta}^T \boldsymbol{x})$$

$$\text{where } \sigma(a) = \frac{1}{1 + \exp(-a)}$$



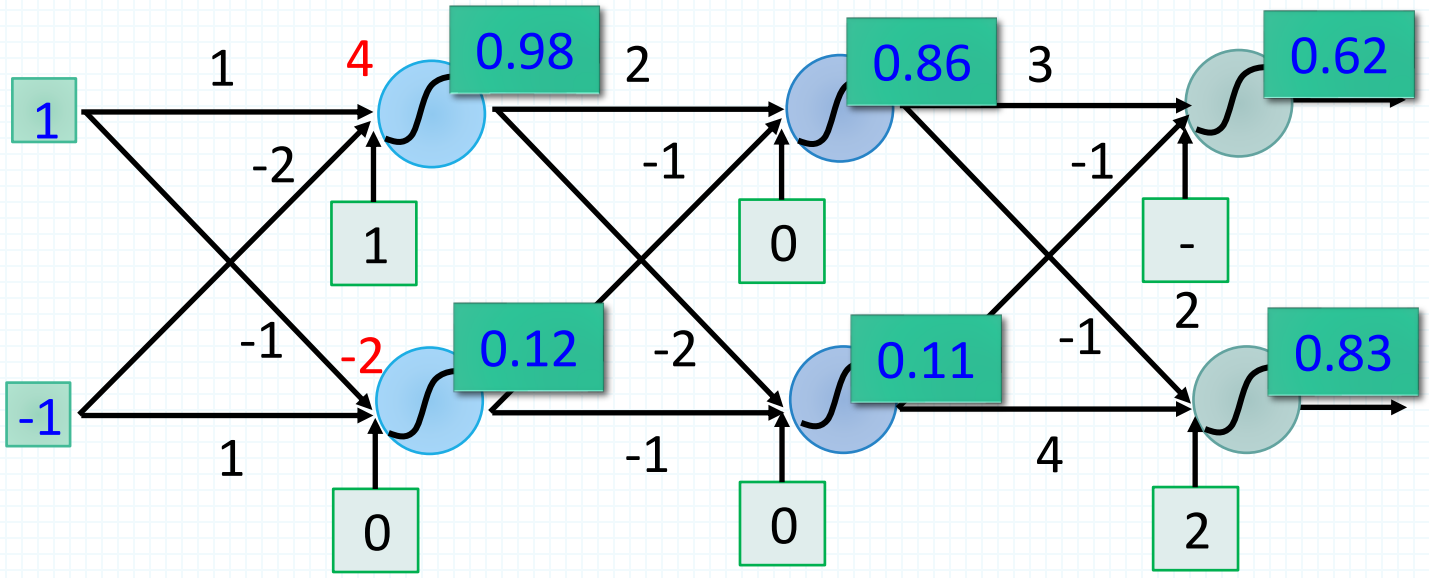
Elements of Neural Networks

- A simple network:



Elements of Neural Networks

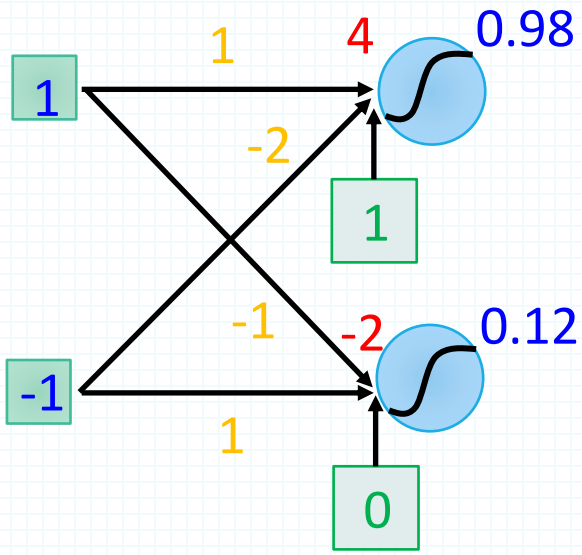
- A simple network, toy example (cont'd)
  - For an input vector $[1 \quad -1]^T$, the output is $[0.62 \quad 0.83]^T$



$f: R^2 \rightarrow R^2$ $\qquad f\left(\begin{bmatrix} 1 \\ -1 \end{bmatrix}\right) = \begin{bmatrix} 0.62 \\ 0.83 \end{bmatrix}$

Matrix Operation

- Matrix operations are helpful when working with multidimensional inputs and outputs

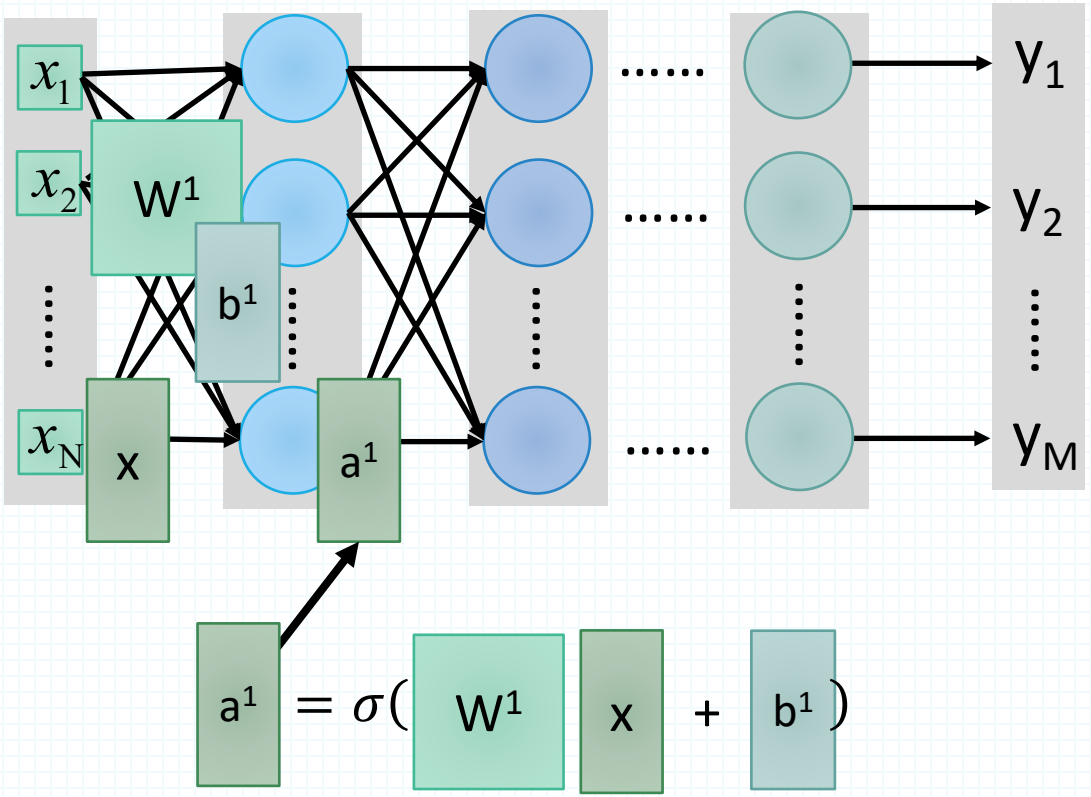


$$\sigma\left(\begin{bmatrix}1 & -2\\ -1 & 1\end{bmatrix}\begin{bmatrix}1\\ -1\end{bmatrix} + \begin{bmatrix}1\\ 0\end{bmatrix}\right) = \begin{bmatrix}0.98\\ 0.12\end{bmatrix}$$

$$\begin{bmatrix}4\\ -2\end{bmatrix}$$

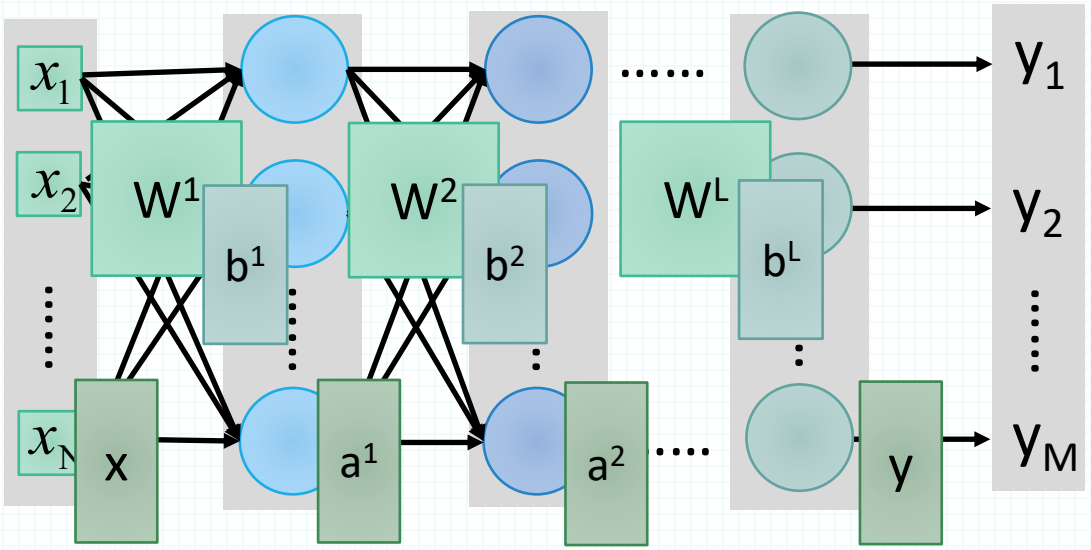
Matrix Operation

- Multilayer NN, matrix calculations for the first layer
- Input vector x, weights matrix W1, bias vector b1, output vector a1



Matrix Operation

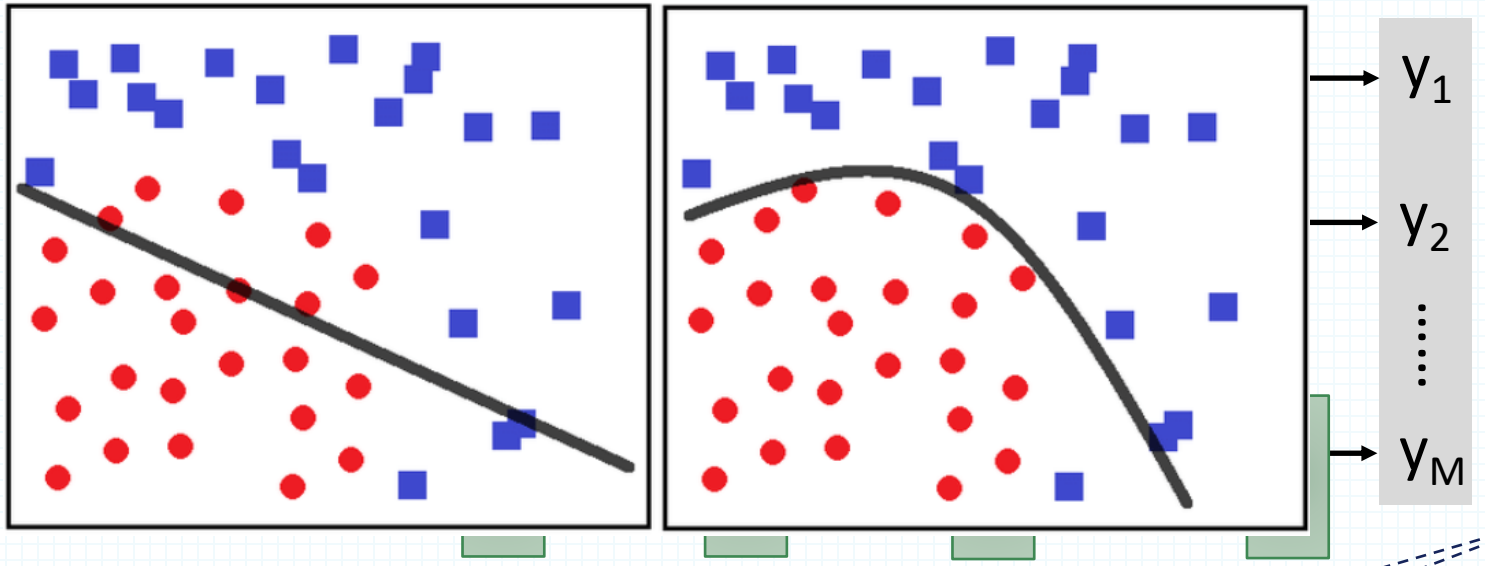
- Multilayer NN, function f maps inputs x to outputs y, i.e., $y = f(x)$



$$\mathbf{y} = f(\mathbf{x}) = \sigma(\mathbf{W}^L \cdots \sigma(\mathbf{W}^2 \sigma(\mathbf{W}^1 \mathbf{x} + \mathbf{b}^1) + \mathbf{b}^2) \cdots + \mathbf{b}^L)$$

Matrix Operation

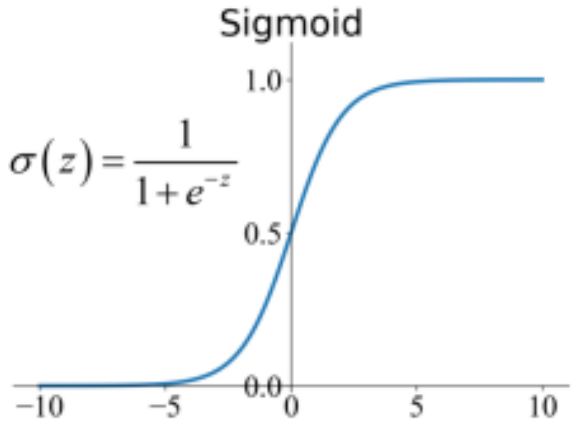
- Multilayer NN, function f maps inputs x to outputs y, i.e., $y = f(x)$



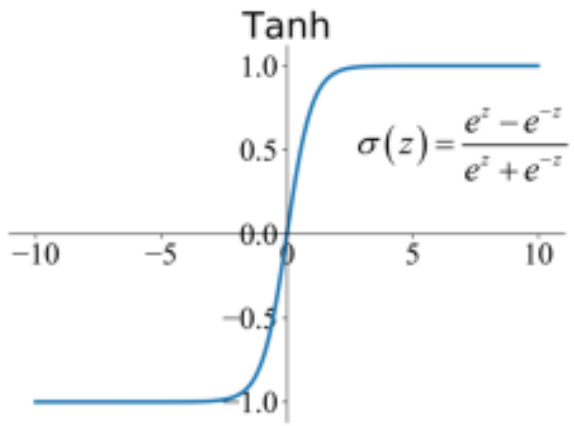
Activation Functions
the role of activation function is to introduce non-linearity into the network

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

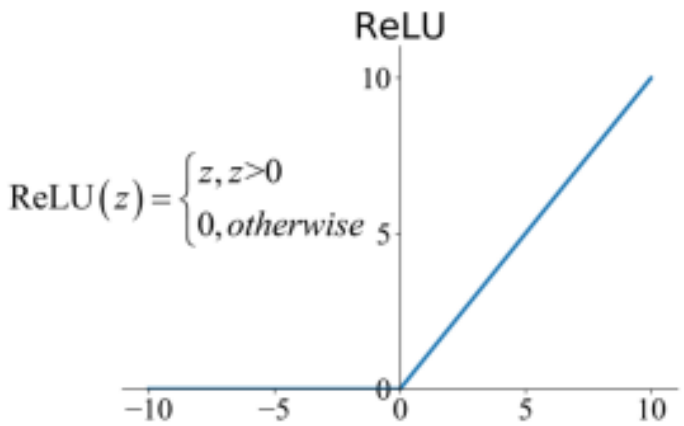
Activation Functions



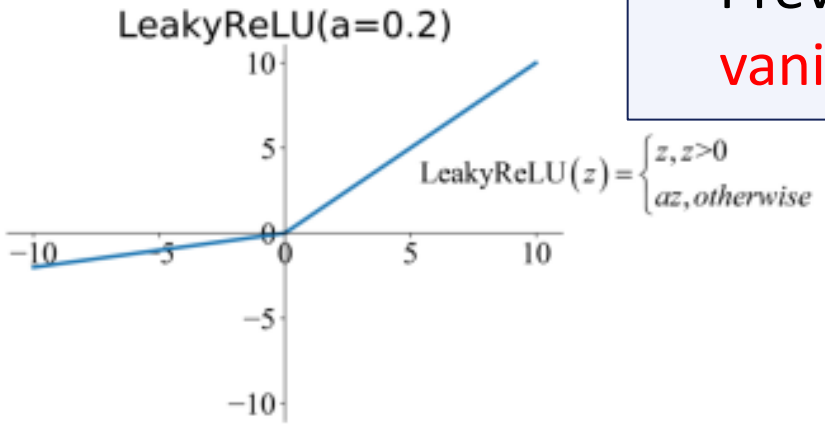
(a)



(b)



(c)

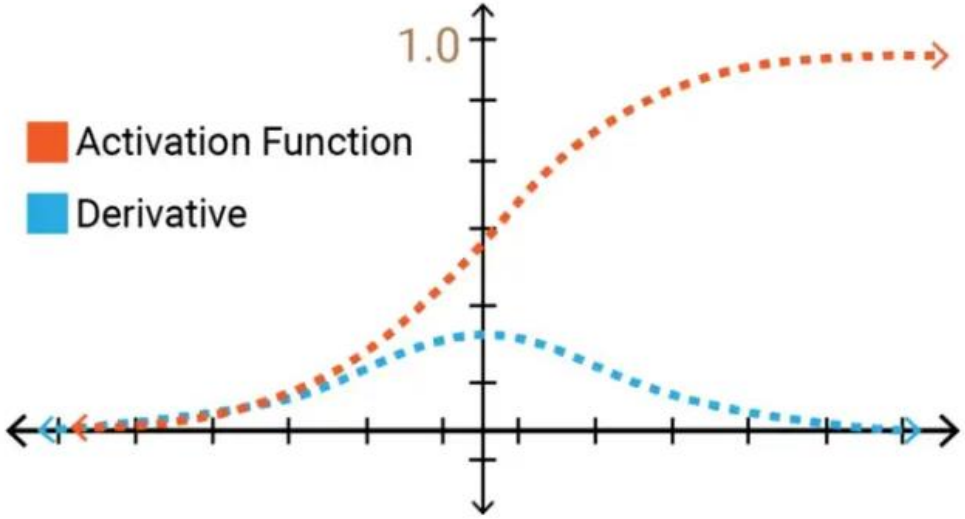


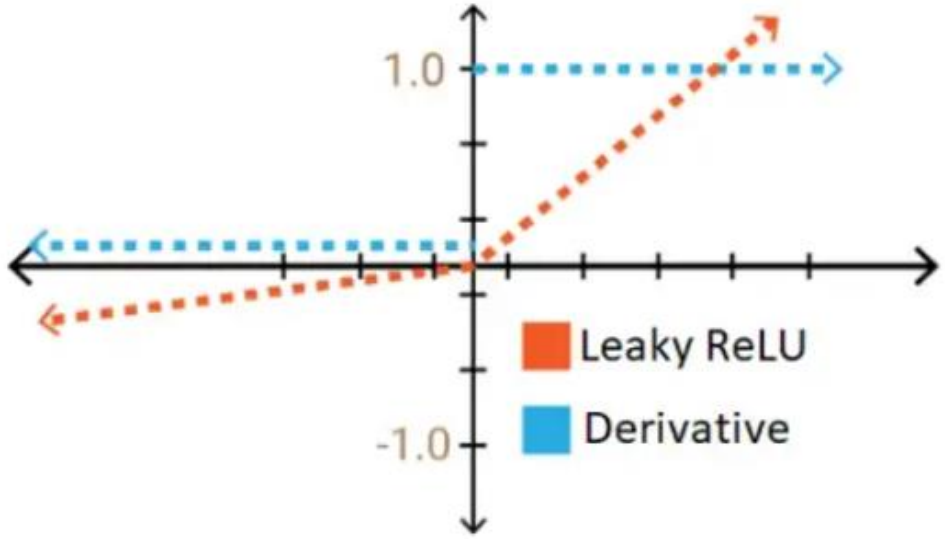
(d)

- Most modern deep NNs use ReLU or Leaky ReLU activations
- fast to compute compared to sigmoid, tanh
- Accelerates the convergence of gradient descent
- Prevents the gradient vanishing problem

Gradient Vanishing Problem

Vanishing gradient problem is a phenomenon that occurs during the training of deep neural networks, where the gradients that are used to update the network become extremely small or "vanish" as they are backpropogated from the output layers to the earlier layers.

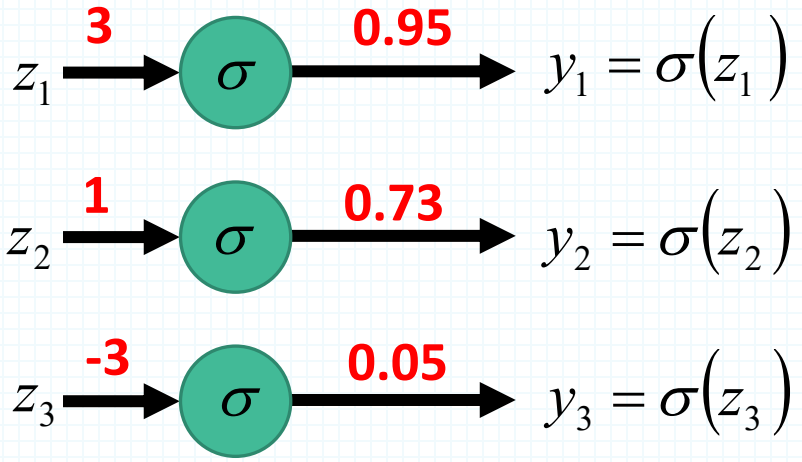




Softmax Layer

- In multi-class classification tasks, the **output layer** is typically a ***softmax layer***
  - I.e., it employs a ***softmax activation function***
  - If a layer with a **sigmoid activation** function is used as the output layer instead, the predictions by the NN may **not be easy to interpret**
    - Note that an output layer with sigmoid activations can still be used for binary classification

A Layer with Sigmoid Activations



Softmax Layer

- The softmax layer applies softmax activations to output **a probability value** in the range [0, 1]
  - The values *z* inputted to the softmax layer are referred to as ***logits***

Probability:
- $0 < y_i < 1$
- $\sum_i y_i = 1$

A Softmax Layer