Deep Learning

ISTD 50.035

Computer Vision

Acknowledgement: Some images are from various sources: UCF, Stanford cs231n, etc.

Linear Classifier

$$s = f(x; W, b) = Wx + b$$

- **Shorthand notation**

$$s = [W\ b][x\ 1]^T$$

$$W \qquad x$$

Input x: (D+1)x1
Weight W: Kx(D+1)
Score s:Kx1

$$s = f(x; W) = Wx$$

Rather insufficient to predict the class of x
- High dimensional input
- Highly nonlinear classification function

Classification

- Classification function for image is **complex, non-linear**

$$y = F(x)$$

x =

y = 1, 2, ... or K (class index)

- Given data points (training examples)

$$y_i = F(x_i)$$

- Able to generalize to unseen example
- Our goal is to learn a good approximation of $F(x)$
- Deep neural network: a class of function with large capacity to provide this approximation
  - With certain parameters learned in training

Stacking linear classifiers

- Stacking linear classifiers to improve representational power (to approximate $F(x)$)

$$s_1 = W_1 x$$

$$s_2 = W_2 s_1$$

Still linear, $W = W_2 W_1$

-

Stacking linear classifiers

- Stacking linear classifiers to improve representational power (to approximate $F(x)$)

$$s_1 = W_1 x$$
$$s_2 = W_2 s_1$$

Still linear, $W = W_2 W_1$

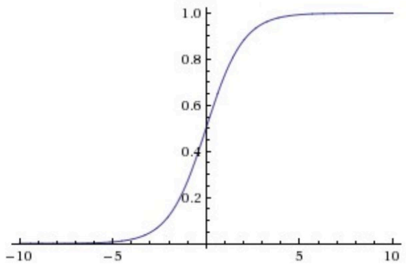
- Add non-linearity between layers (stages)

$$s_1 = W_1 x$$
$$s_2 = W_2 \sigma(s_1)$$

Can approximate any continuous function $F(x)$

- **Activation function** is applied elementwise
- Sigmoid:

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



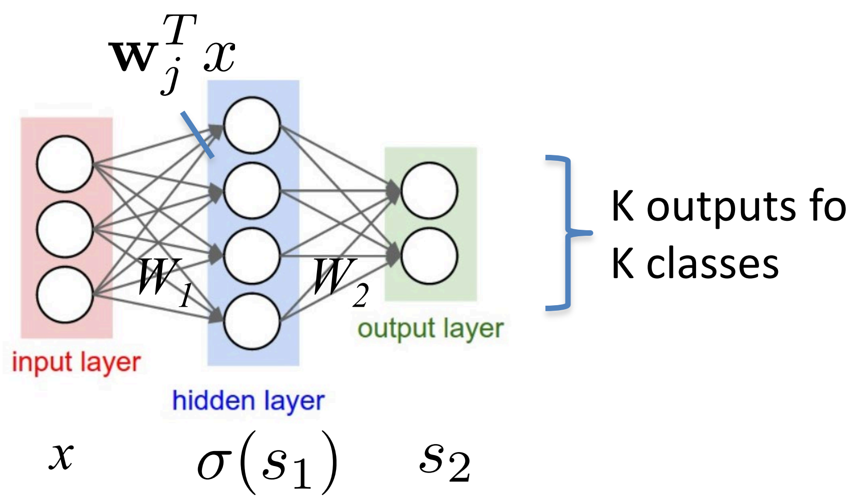
We obtain a neural network

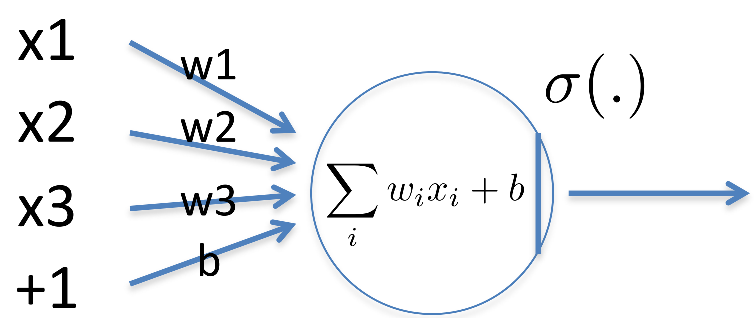
Neural Network

- Neural network: collection of **neurons**
  - Connected in an acyclic graph
  - Output of a neuron can be input of another

$$s_1 = W_1 x$$
$$s_2 = W_2 \sigma(s_1)$$

Linear classifier with activation as input



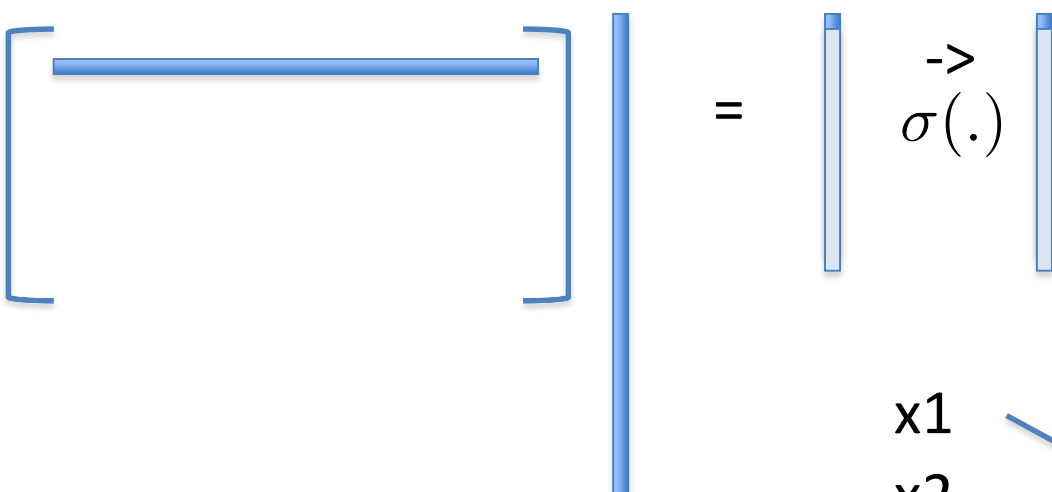


A neuron

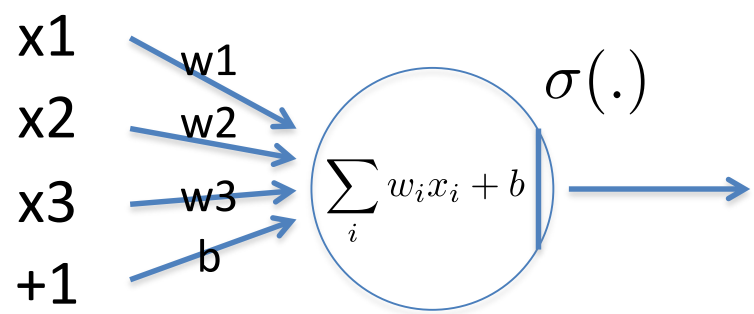
- Neuron: a computational unit, take input x, output:

$$\sigma(\sum_i w_i x_i + b)$$

- Activation (Sigmoid) function is applied elementwise

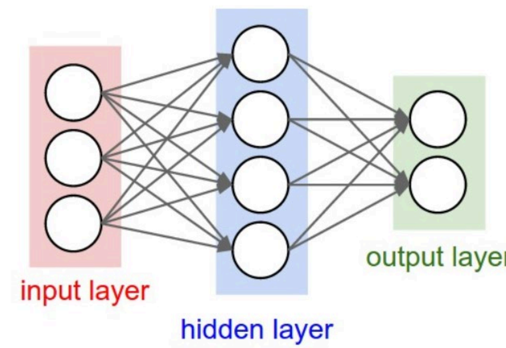


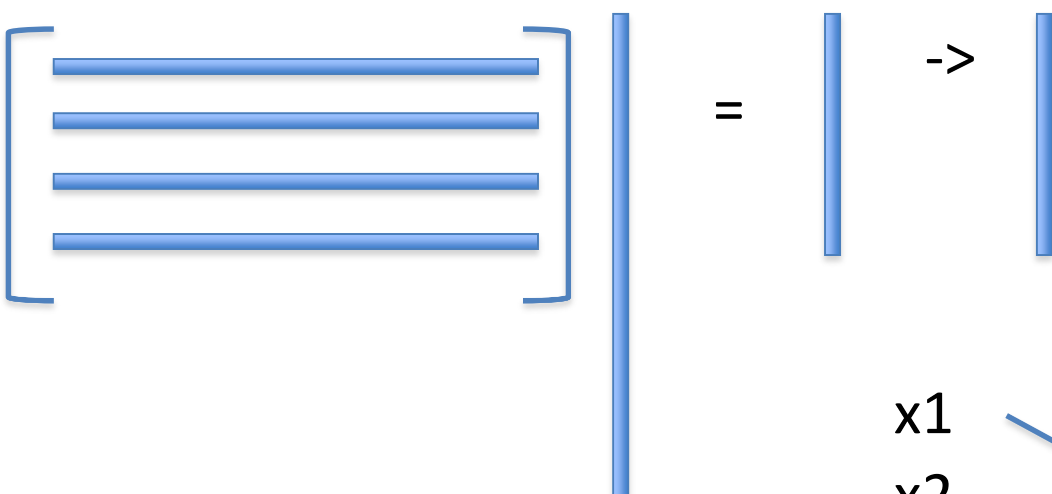
$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

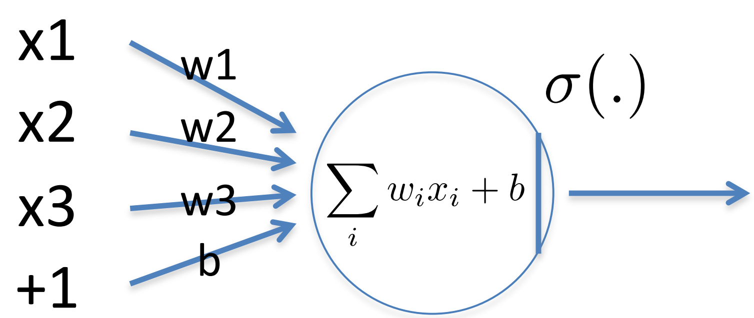


Neural Network

- Neural network: collection of neurons
  - Connected in an acyclic graph
  - Output of a neuron can be input of another



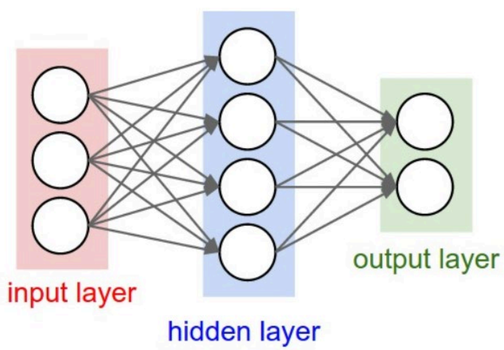




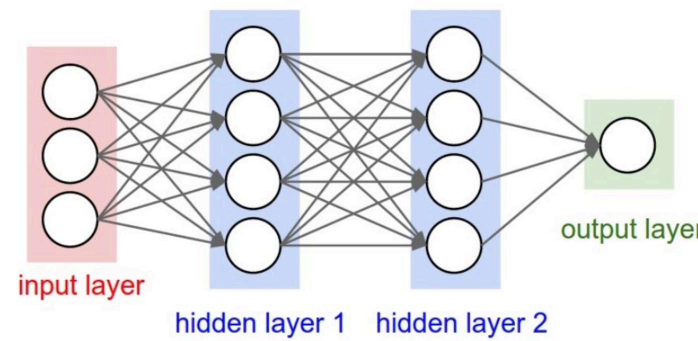
Neural Network

- Hidden layer: values are not observed in the training set
- Output layer: no activation

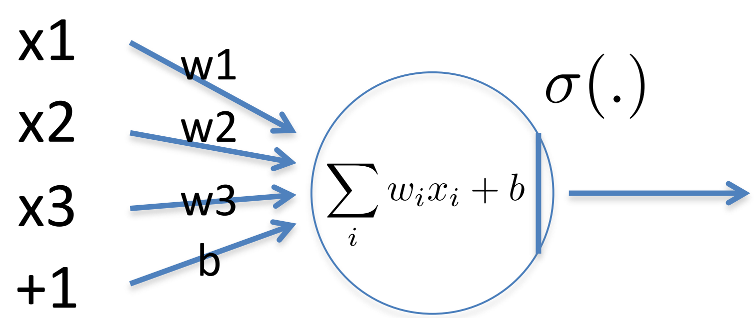
2-layer NN: 1 hidden, 1 output layer



3-layer NN: 2 hidden, 1 output layer



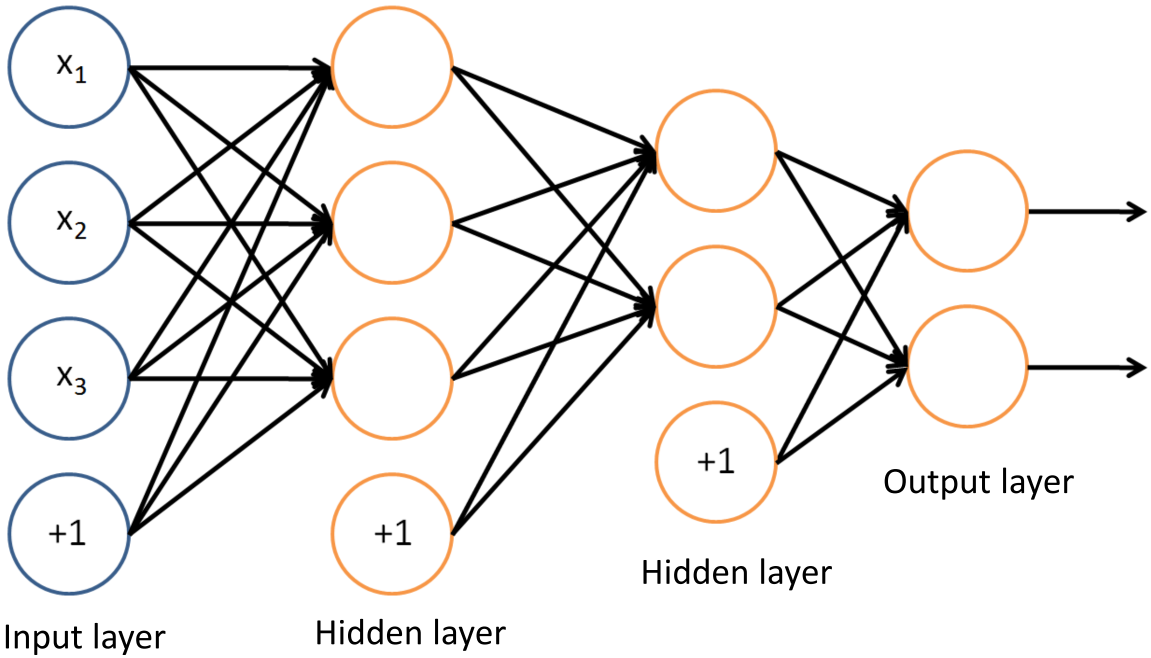
Artificial neural network (ANN)
Multi-layer perceptrons (MLP)



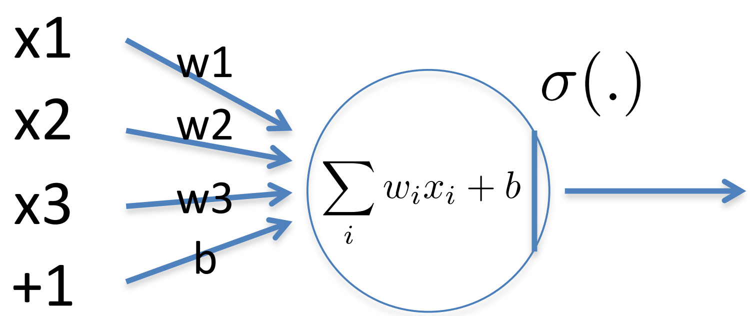
Neural Network

- Hidden layer: values are not observed in the training set
- Output layer: no activation

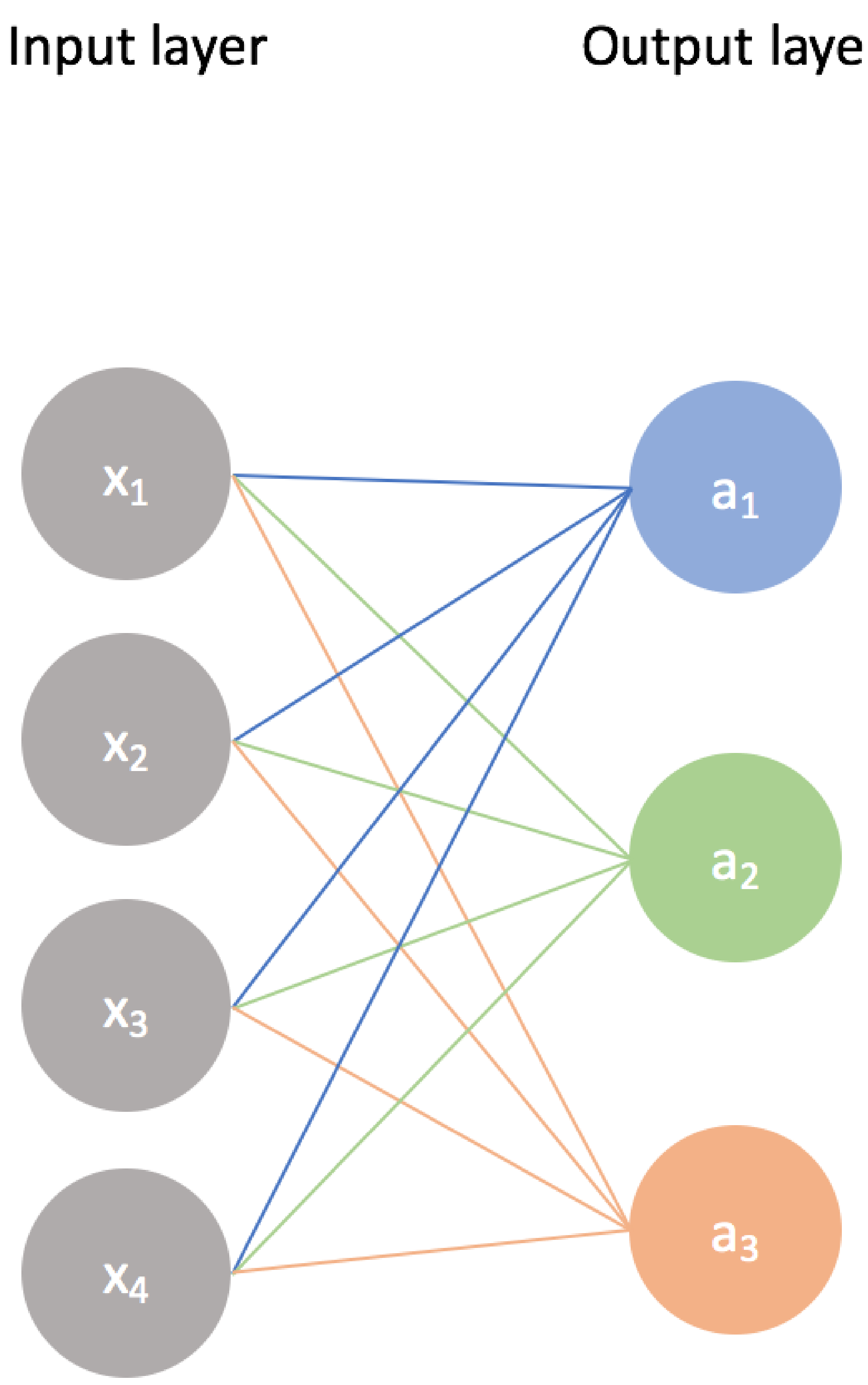
Explicit indication of bias



Artificial neural network (ANN)
Multi-layer perceptrons (MLP)



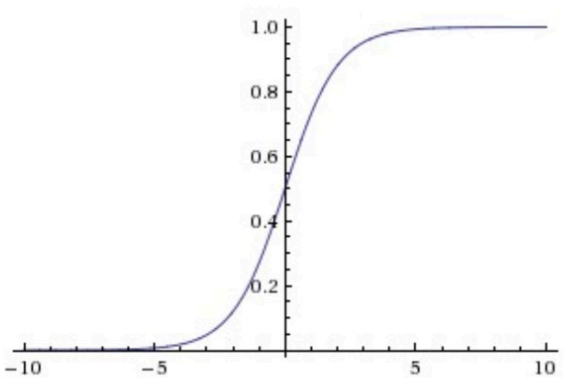
A simple neural network



$$\begin{bmatrix} w_1 & w_2 & w_3 & w_4 \\ w_1 & w_2 & w_3 & w_4 \\ w_1 & w_2 & w_3 & w_4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} b \\ b \\ b \end{bmatrix} = \begin{bmatrix} w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b \\ w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b \\ w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b \end{bmatrix} \xrightarrow[activation]{} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

Activation function

Sigmoid

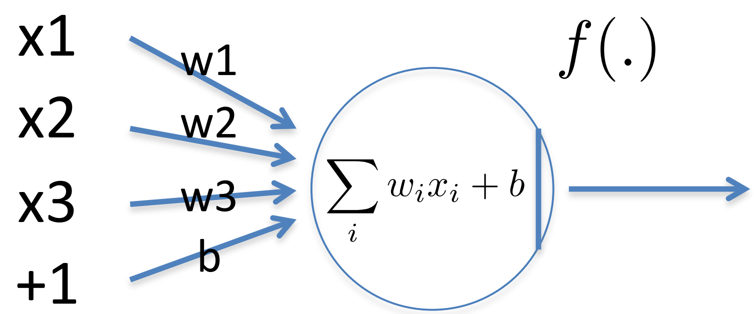


$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Easy to compute gradient:

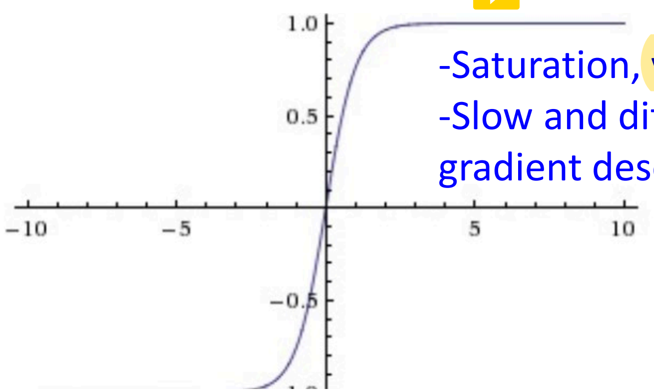
$$\sigma'(z) = \sigma(z)(1 - \sigma(z))$$

-Incorporate non-linear
-Limit the output range (or additional normalization)
-Decision / probabilistic interpretation
 -Detect feature or not
 -Biological neuron: to fire or not



Activation function

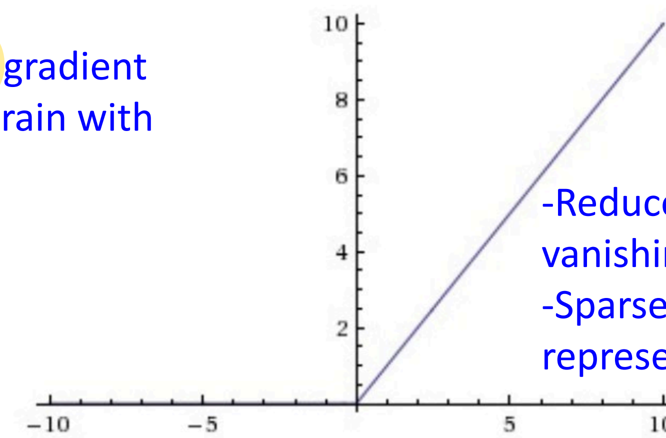
Hyperbolic tangent



-Saturation, vanishing gradient
-Slow and difficult to train with gradient descent

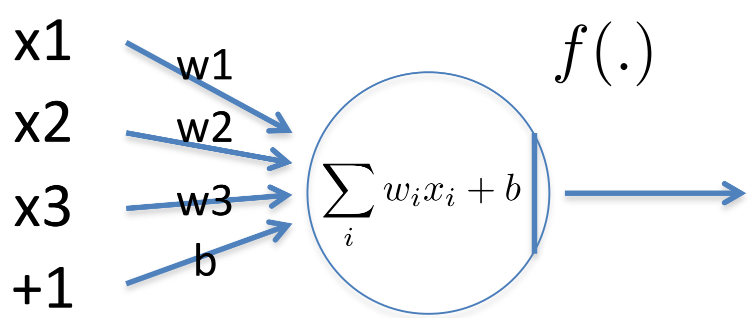
$$f(z) = tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$

Rectified linear unit (ReLU)



-Reduce issue of vanishing gradient
-Sparse representation

$$f(z) = max(0, z)$$



NN as a function approximation

- NN with one hidden layer can approximate any continuous function F(x)
- Classification function in our case
- In practice, NN with multiple hidden layers performs better