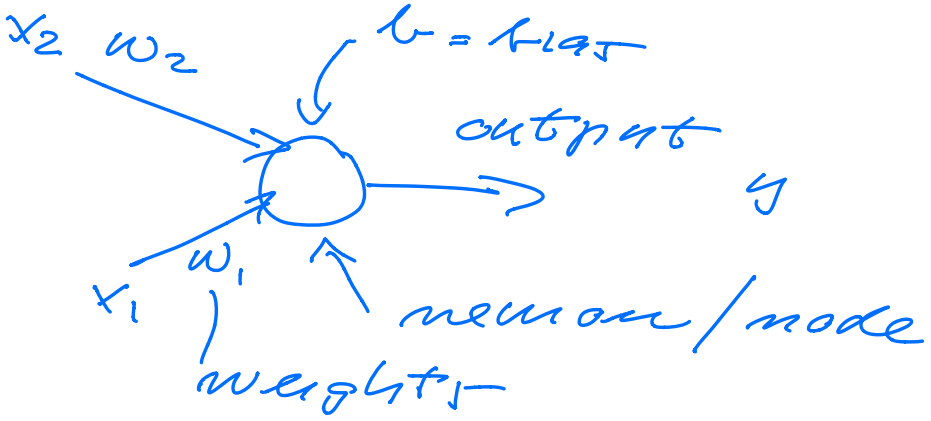
Lecture January 25

neural networks

- single perceptron model



$$y = f(u) \qquad u = \sum_{i=1}^{2} w_i x_i$$

activation function: typical example is the sigmoid, tanh

Linear regression:

$$f(x; \beta) = X \cdot \beta$$

w = our parameter

b = bias

$$f(x_1 w_1 + x_2 w_2 + b) = y$$

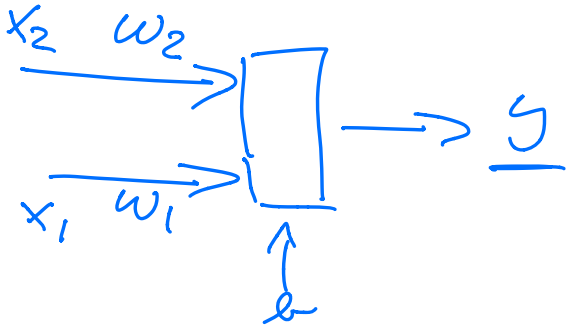
$$y \in [0, 1]$$

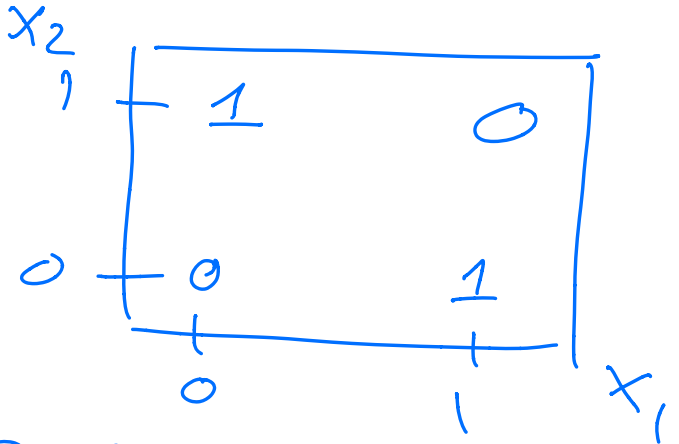
XOR - model (Exclusive ok)

Linear fit

$$x_1 w_1 + x_2 w_2 + b = y$$

$$X = \{ [0,0]^T, [0,1]^T, [1,0]^T, [1,1]^T \}$$





Define a cost function

$$C(w, b) = \frac{1}{4} \sum_{x=1}^{4} (y_i - \tilde{y}_i)^2$$

$$= \frac{1}{4} \sum_{i=1}^{4} (t_i - y_i)^2$$

t_i: target

y_i: output from model

$$\beta = [b, w_1, w_2] \quad (\theta)$$

Design Matrix

$$X = \begin{bmatrix} b & w_1 & w_2 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

$$(X^T X) = \begin{bmatrix} 4 & 2 & 2 \\ 2 & 2 & 1 \\ 2 & 1 & 2 \end{bmatrix}$$

$$(X^T X)^{-1} X^T t = \begin{bmatrix} 1/2 \\ 0 \\ 0 \end{bmatrix}$$

$$b = 1/2 \qquad w_1 = w_2 = 0$$

$$y = X\beta = 1/2$$

all predictions give 1/2

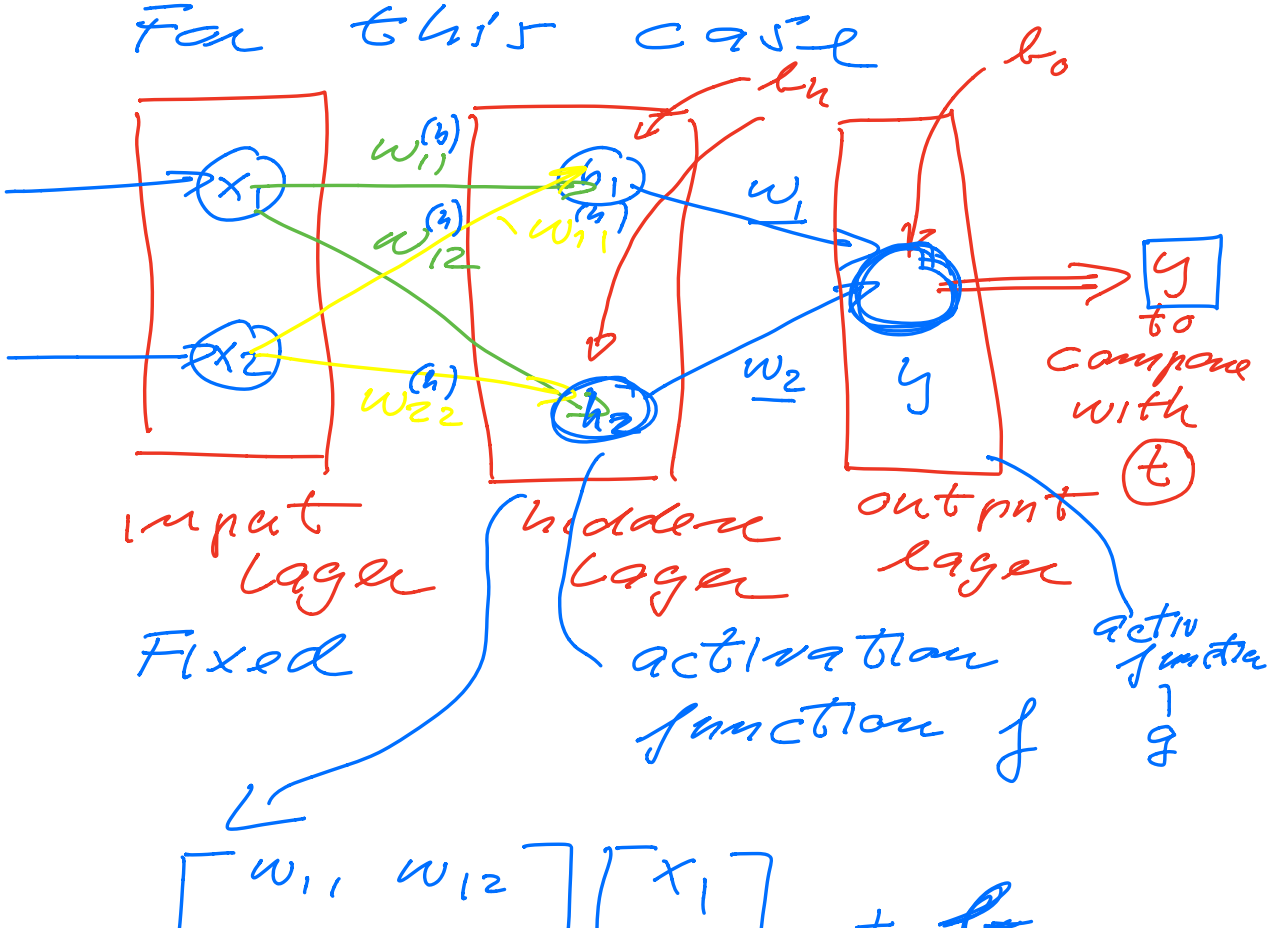
We want $\boxed{0\ 1\ 1}$

Linear regression fails.

We can make a network with a hidden layer and two neurons/nodes.

⇒ This gives correct results

For this case



$$\begin{bmatrix} w_{11} & w_{12} \end{bmatrix} \begin{bmatrix} x_1 \end{bmatrix} + b_h$$

$$\begin{bmatrix} w_{21} & w_{22} \end{bmatrix} \begin{bmatrix} x_2 \end{bmatrix}$$

$W \qquad x$

(i) Feed Forward stage with randomly defined values for W and b.

$$h_1 = w_{11}x_1 + w_{12}x_2$$

$$h_2 = w_{21}x_1 + w_{22}x_2$$

$$Wx + b = h \rightarrow$$

$$f(h) = f(Wx + b)$$

→ output stage

$$g(f(Wx+b))$$

$$= g(w_1 h_1 + w_2 h_2)$$

$$= y$$

to be compared with target t

How do we fit the targets t with our model outputs y?

We need an algorithm to find weights and biases.

(ii) Back Propagation algo, application of chain rule, New values of weights and biases.

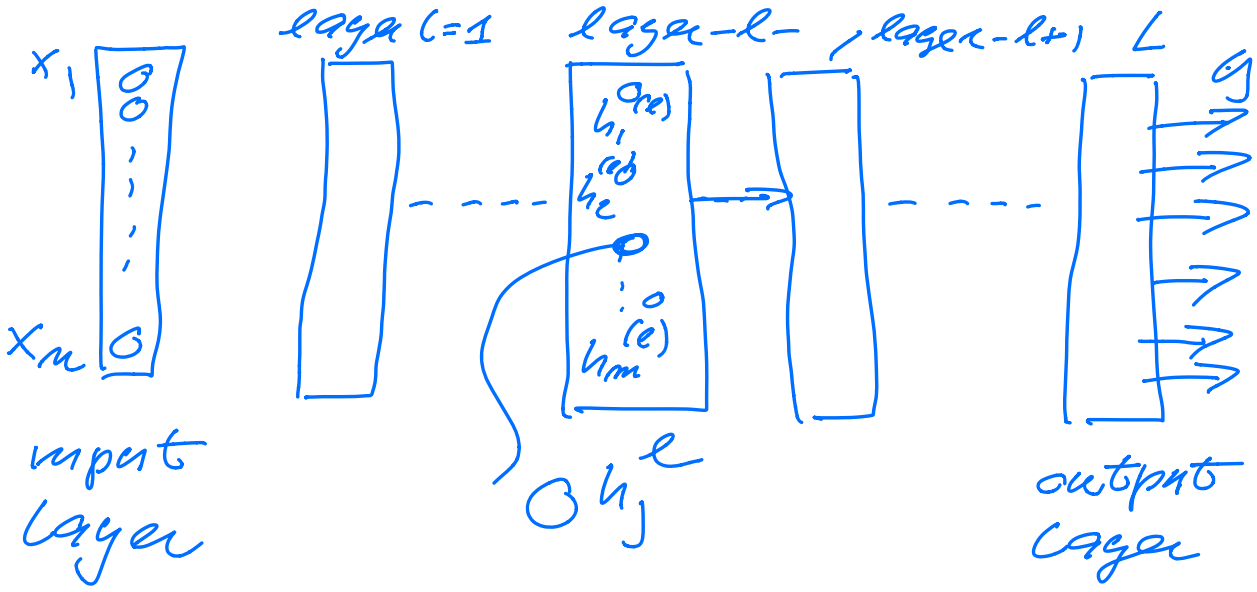
(iii) New FF pass, New Back prop.

(iv) Repeat (i) - (iii) till achieved accuracy.

In Essence:

- Back propagation
- in each layer we need Gradient descent

MLP / FFNN: Basic defs:



Regression:

$$C(w,b) = \frac{1}{n}\sum_{i=0}^{n-1}\left(t_i - y_i\right)^2$$

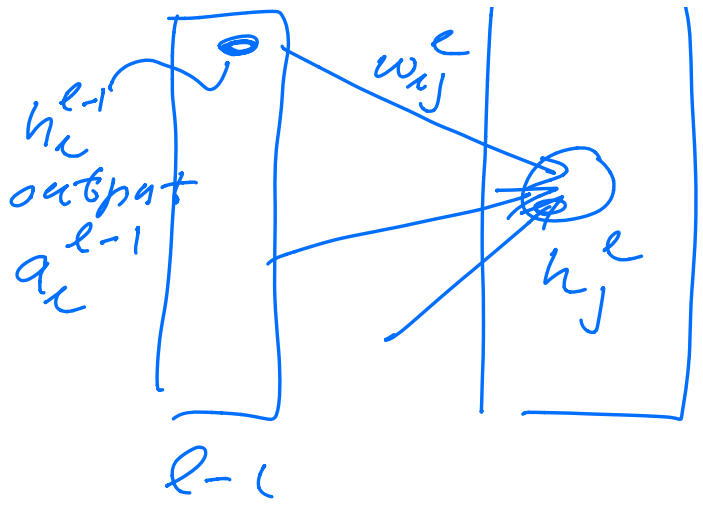
weights in a layer w_{ij}^l

inputs from layer $l-1$ to node in l : a_i^{l-1}

bias in layer l for node j : b_j^l

input to node j in layer l

$$z_j^l = \sum_{i=1}^{M_{l-1}} w_{ij}^l a_i^{l-1} + b_j^l$$



- new parameters:
  - # hidden layers
  - # nodes in hidden layers
  - # Different activation functions
    (normal approach is to have same activation for hidden nodes and different activation function for output layer)

→ output from node h_j^{ℓ}

$$a_j^{\ell} = f(z_j^{\ell}) = \text{sigmoid}$$

$$\frac{1}{1+\exp(-z_j^\ell)}$$

Derivatives & Back prop.

$$\frac{\partial C(w,b)}{\partial w} \quad \& \quad \frac{\partial C}{\partial b}$$

$$\frac{\partial z_j^\ell}{\partial w_{ij}^\ell} = a_i^{\ell-1}$$

$$\frac{\partial z_j^\ell}{\partial a_i^{\ell-1}} = w_{ji}^\ell$$

$$\frac{\partial a_j^\ell}{\partial z_j^\ell} = a_j^\ell(1-a_j^\ell)$$

$$= f(z_j^\ell)(1-f(z_j^\ell))$$

$\ell = L =$ output layer.

$$C(w^L, b^L) = \frac{1}{2}\sum (y_i - t_i)^2$$

$$= \frac{1}{2} \sum_i (a_i^L - t_i)^2$$

$$\frac{\partial C}{\partial w_{jk}^L} = (a_j^L - t_j) \frac{\partial a_j^L}{\partial w_{jk}^L}$$

$$\frac{\partial a_j^L}{\partial w_{jk}^L} = \frac{\partial a_j^L}{\partial z_j^L} \frac{\partial z_j^L}{\partial w_{jk}^L} =$$

$$a_j^L (1 - a_j^L) a_k^{L-1}$$

$$\frac{\partial C(w, b)}{\partial w_{jk}^L} = (a_j^L - t_j) \times a_j^L (1 - a_j^L) a_k^{L-1}$$

$$\delta_j^L = a_j^L (1 - a_j^L)(a_j^L - t_j)$$

$$= f'(z_j^L) \frac{\partial C}{\partial a_j^L}$$

$$\frac{\partial C}{\partial w_{jk}^L} = \delta_j^L a_k^{L-1}$$

$$\frac{\partial C}{\partial w_{jk}^{l}} = \delta_j^{l} a_k$$

$$\delta_j^{l} = \frac{\partial C}{\partial z_j^{l}} = \frac{\partial C}{\partial a_j^{l}} \frac{\partial a_j^{l}}{\partial z_j^{l}}$$

$$\delta_j^{l} = \frac{\partial C}{\partial b_j^{l}} \frac{\partial b_j^{l}}{\partial z_j^{l}} = \frac{\partial C}{\partial b_j^{l}}$$

$\frac{\partial b_j^{l}}{\partial z_j^{l}} = 1$

Parameters

$$C(w, b) = \frac{1}{2} \sum (y_i - t_i)^2$$

- w, b
- # hidden layers
- # nodes hidden layers
- activation function for hidden layers
- Learning rates γ

- for gradient method
- activation func for output layer
- type of gradient methods
- hyperparameter
  λ $\lambda \|W\|_2^2$