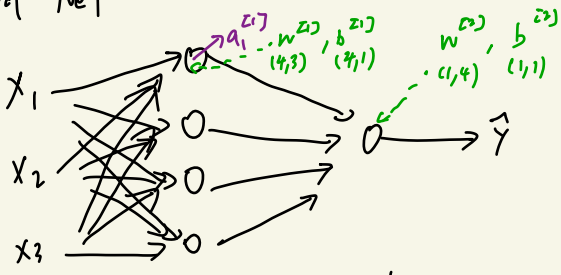
$[1]$ ← layer

X

Neural Net



2 layer NN

Input layer

Hidden layer

output layer

$a^{[2]}$ $a^{[2]}$

$a^{[0]} = X$

$a^{[1]}$

$$= \begin{bmatrix} a_1^{[1]} \\ a_2^{[1]} \\ a_3^{[1]} \\ a_4^{[1]} \end{bmatrix}$$

Hidden layer node

$z_1^{[1]} = W_1^{[1]T} X + b_1^{[1]}$ → $a_1^{[1]} = \sigma(z_1^{[1]})$

$X \longrightarrow a^{[2]} = \hat{y}$

$x^{(1)} \longrightarrow a^{[2](1)} = \hat{y}^{(1)}$

$\vdots$

$a^{[2](i)}$

for i in n

$z^{[1](i)} = W^{[1]} x^{(i)} + b^{[1]}$

$a^{[1](i)} = \sigma(z^{[1](i)})$

$$X = \begin{bmatrix} | & | & | & & | \\ x^{(1)} & x^{(2)} & x^{(3)} & \cdots & x^{(m)} \\ | & | & | & & | \end{bmatrix}$$

$$A^{[1]} = \begin{bmatrix} | & | & & | \\ a^{[1](1)} & a^{[1](2)} & \cdots & a^{[1](m)} \\ | & | & & | \end{bmatrix}$$ ↓ # hidden unit

→ training sample

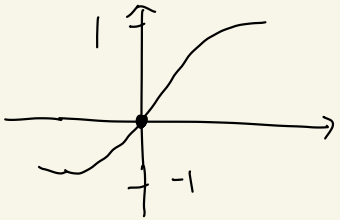
$W^{[1]} = \begin{bmatrix} \text{---} \\ \text{---} \\ \text{---} \end{bmatrix}$ $W^{[1]}X^{(1)} = \begin{bmatrix} \vdots \\ \vdots \\ \vdots \end{bmatrix}$ $W^{[1]}X^{(2)} = \begin{bmatrix} \vdots \\ \vdots \\ \vdots \end{bmatrix}$ $W^{[1]}X^{(3)} = \begin{bmatrix} \vdots \\ \vdots \\ \vdots \end{bmatrix}$

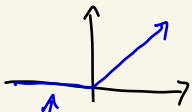
$$W^{[1]} \begin{bmatrix} | & | & | & & | \\ x^{(1)} & x^{(2)} & x^{(3)} & \cdots & x^{(n)} \\ | & | & | & & | \end{bmatrix} = \begin{bmatrix} \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet \end{bmatrix} = \begin{bmatrix} | & | & | \\ z^{[1](1)} & z^{[1](2)} & z^{[1](3)} \\ | & | & | \\ +b^{[1]} & +b^{[1]} & +b^{[1]} \end{bmatrix} = Z^{[1]}$$

X

Activation function

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





disadvantage

Why use activation function?

$a^{[z]}$ just linear output

If $0 \le \hat{y} \le 1$, sigmoid

If $-1 \le \hat{y} \le 1$, $\tanh(z)$ — tanh is always better

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

$$g'(z) = a(1-a)$$

$$g(z) = \tanh(z)$$

$$g'(z) = 1 - (\tanh z)^2$$

ReLu

$$g(z) = \max(0, z)$$

$$g'(z) = \begin{cases} 0 & \text{if } z \le 0 \text{ (or)} \\ 1 & \text{if } z \ge 1 \end{cases}$$

Leaky ReLu

$$g(z) = \max(0.01z, z)$$

$$g'(z) = \begin{cases} 0.01, & \text{if } z < 0 \\ 1, & \text{if } z \ge 1 \end{cases}$$

Gradient Descent for Neural Net

parameter: $W^{[1]}$, $b^{[1]}$, $W^{[2]}$, $b^{[2]}$

$(n^{[1]}, n^{[0]})$ $(n^{[1]}, 1)$ $(n^{[2]}, n^{[1]})$ $(n^{[2]}, 1)$

$n_x = n^{[0]}$, $n^{[1]}$, $n^{[2]} = 1$

cost function $J = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(\hat{y}, y)$

$\hat{y} = a^{[2]}$

Random init

$W^{[1]} = np.random.randn((2,2)) * 0.01$

$b^{[1]} = np.zeros((2,1))$

Gradient Descent:

init random

Repeat {

predict ($\hat{y}^i$, $i = 1 \cdots m$)

$dW = \frac{dJ}{dW}$ $db = \frac{dJ}{db}$

$W^{[1]} = W^{[1]} - \alpha dW^{[1]}$

$b^{[1]} = b^{[1]} - \alpha db^{[1]}$

$W^{[2]} = W^{[2]} - \alpha dW^{[2]}$

Forward

$Z^{[1]} = W^{[1]} X + b^{[1]}$

$A^{[1]} = g^{[1]}(Z^{[1]})$

$Z^{[2]} = W^{[2]} A^{[1]} + b^{[2]}$

$A^{[2]} = g^{[2]}(Z^{[2]})$

Back

$dZ^{[2]} = A^{[2]} - Y$ $\quad Y = [y^{(1)} y^{(2)} \cdots y^{(m)}]$

$dW^{[2]} = \frac{1}{m} dZ^{[2]} A^{[1]T}$

$db^{[2]} = \frac{1}{m} np.sum(dZ^{[2]}, axis=1, keepdim=True)$

$dZ^{[2]} = W^{[2]T} dZ^{[2]} * g^{[1]'}(Z^{[1]})$ stop ch, $\rightarrow (n,1)$

$W^{[2]T} dZ^{[2]}$: $(n^{[1]}, m)$; $g^{[1]'}(Z^{[1]})$: $(n^{[1]}, m)$; $*$: element wise product

$dW = \frac{1}{m} dZ^{[1]} X^T$

$db = \frac{1}{m} np.sum(dZ^{[1]}, axis=1, keepdim=True)$