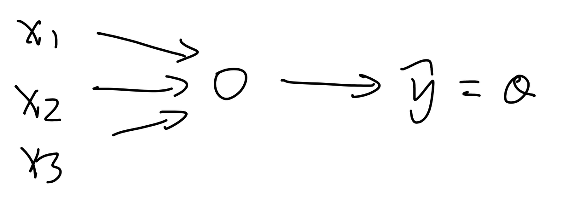
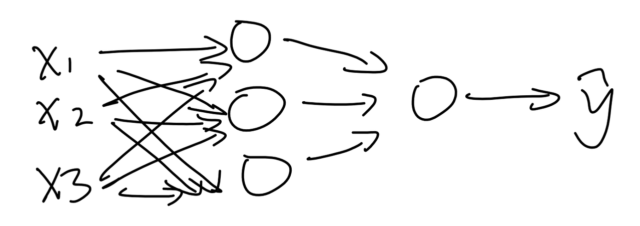
神经网络第三章

2022年7月12日 星期二 08:01

Overview



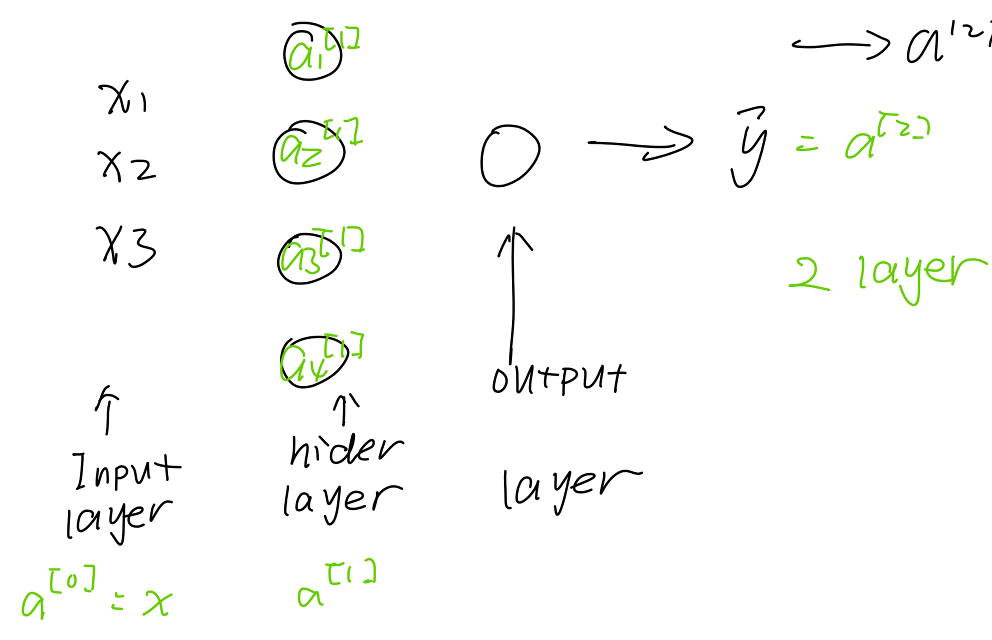
x, w, b → $z = w^Tx + b \longrightarrow a = \sigma(z) \longrightarrow \mathcal{L}(a, y)$



$z^{(1)} = W^{(1)}x + b^{(1)} \longrightarrow a^{(1)} = \sigma(z^{(1)})$

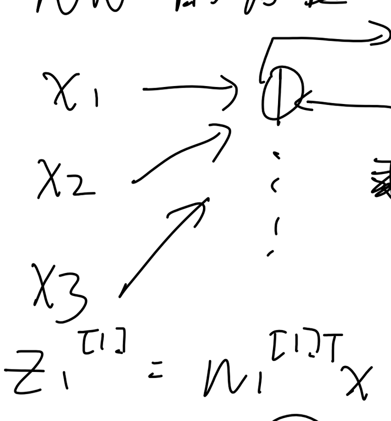
$\longrightarrow z^{(2)} = W^{(2)}a^{(1)} + b^{(2)}$

$\longrightarrow a^{(2)} = \sigma(z^{(2)}) \longrightarrow \mathcal{L}(a^{(2)}, y)$



2 layer Neural Network

NN 的原理：

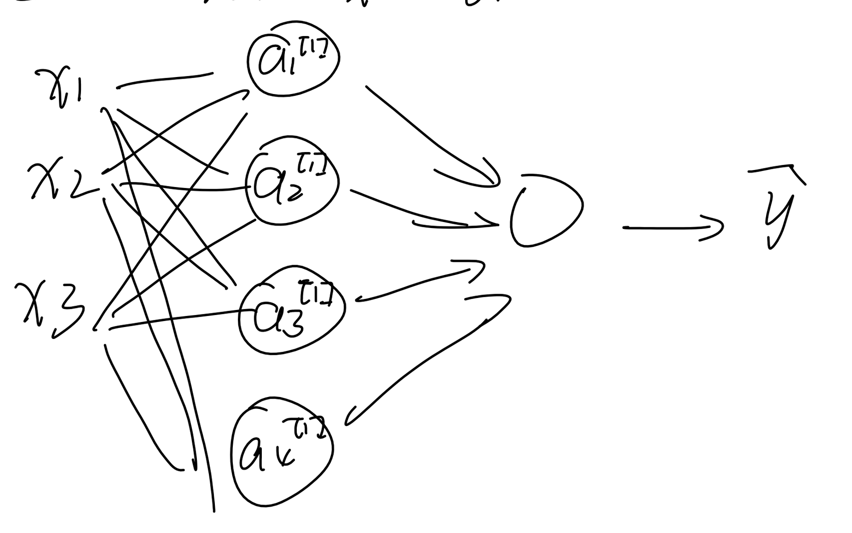


$z^{[1]} = w_1^{[1]T}x + b_1^{[1]}$

$a_1^{[1]} = \sigma(z_1^{[1]})$

剩下的同理

$z_1^{[1]} = w_1^{[1]T}x + b_1^{[1]}, \; a_1^{[1]} = \sigma(z_1^{[1]})$



$z_1^{[1]} = w_1^{[1]T}x + b_1^{[1]}, \; a_1^{[1]} = \sigma(z_1^{[1]})$

$z_2^{[1]} = w_2^{[1]T}x + b_2^{[1]}, \; a_2^{[1]} = \sigma(z_2^{[1]})$

$z_3^{[1]} = w_3^{[1]T}x + b_3^{[1]}, \; a_3^{[1]} = \sigma(z_3^{[1]})$

$z_4^{[1]} = w_4^{[1]T}x + b_4^{[1]}, \; a_4^{[1]} = \sigma(z_4^{[1]})$

$z^{[2]} = W^{[2]} \cdot a^{[1]} + b^{[2]}$

$a^{[2]} = \sigma(z^{[2]})$

将 w 放入一个矩阵中

$$\begin{bmatrix} — & w_1^{[1]T} & — \\ — & w_2^{[1]T} & — \\ & \cdots & \\ — & w_4^{[1]T} & — \end{bmatrix}$$

向量化表示

$x \longrightarrow a^{[2]} \; \hat{y}$

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

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

$\vdots$

$x^{(m)} \longrightarrow a^{[2](m)} \; \hat{y}^{(m)}$

$a^{[2](i)}$ ← 第 i 个训练实例

第二层

for $i = 1$ to m,

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

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

$z^{[2](i)} = W^{[2]}a^{[1](i)} + b^{[2]}$

$a^{[2](i)} = \sigma(z^{[2](i)})$

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

(n_x, m)

$$Z^{[1]} = \begin{bmatrix} | & | & & | \\ z^{[1](1)} & z^{[1](2)} & \cdots & z^{[1](m)} \\ | & | & & | \end{bmatrix}$$

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

sigmoid function ⟶ activate function

- given x:

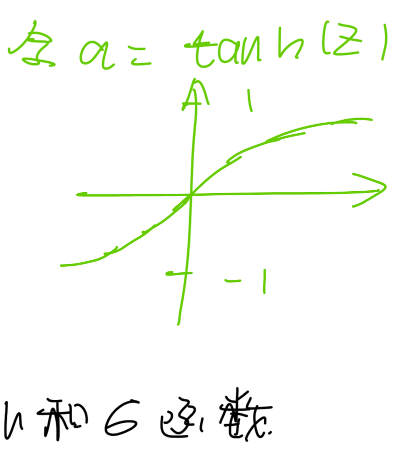
$z^{[1]} = W^{[1]}x + b^{[1]}$

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

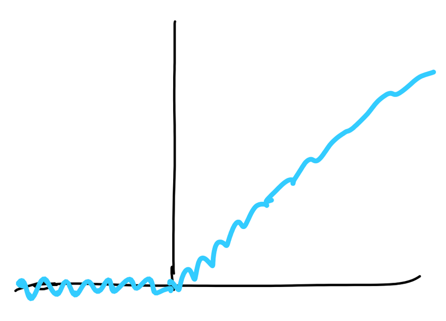
$z^{[2]} = W^{[2]}x + b^{[2]}$

$a^{[2]} = \sigma(z^{[2]}) = g(z^{[2]})$

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



- 不同层可以用不同算法，用 tanh 和 σ 函数

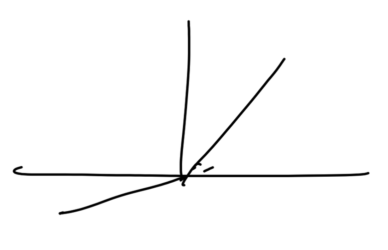


$a = \max(0, z)$

当 z 为负，斜率为0

当 z 为正，斜率为1

(Relu 函数)



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

(leaky Relu 函数)

如果初始函数为线性激活，则后续计算都为线性激活

故此时线性无效，要采用非线性

- 反向传播 slope 计算

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

$\dfrac{d}{dz}g(z) = \dfrac{1}{1+e^{-z}}\left(1 - \dfrac{1}{1+e^{-z}}\right)$

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

$g(z) = \tanh z = \dfrac{e^z - e^{-z}}{e^z + e^{-z}}$

$g'(z) = \dfrac{d}{dz}g(z) = 1 - (\tanh(z))^2$

Gradient Descent:

Repeat {

Compute para... $(\hat{y}^{(i)}, i = 1, \ldots, m)$

$dw^{[1]} = \dfrac{dJ}{dw^{[1]}}, \; db^{[1]} = \dfrac{dJ}{db^{[1]}}$

}

Forward propagation

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

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

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

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

Back propagation

$dZ^{[2]} = A^{[2]} - Y$

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

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

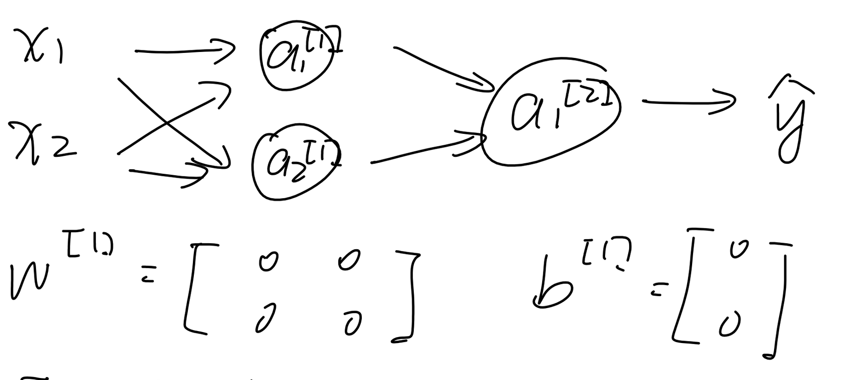
$dZ^{[1]} = W^{[2]T} dZ^{[2]} * g^{[1]\prime}(Z^{[1]})$

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

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

$dZ^{[1]} = W^{[2]T} dZ^{[2]} * g^{[1]\prime}(Z^{[1]})$

Initialize.



$W^{[1]} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \quad b^{[1]} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

Symmitric

得到结论：权重参数矩阵为零，

则迭代一次后值不变

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

→ $b^{[1]}$ = np.zero((2,1))

$W^{[2]} = \ldots$

$b^{[2]} = 0$

set 0.01：为了使步长更短