

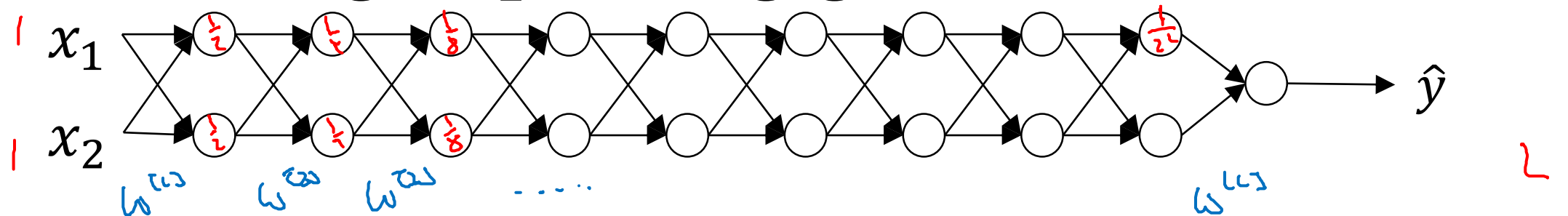


deeplearning.ai

Setting up your
optimization problem

Vanishing/exploding
gradients

Vanishing/exploding gradients



$g(z) = z$ $b^{(1)} = 0$

$\hat{y} = w^{(L,1)} a^{(L-1,1)} + w^{(L,2)} a^{(L-1,2)} + \dots + w^{(L,L)} a^{(L-1,L)}$

$a^{(L-1,1)} = w^{(L-1,1)} x^{(L-2,1)} + w^{(L-1,2)} x^{(L-2,2)} + \dots + w^{(L-1,L)} x^{(L-2,L)}$

$w^{(1,1)} > I$

$w^{(1,2)} < I$ $\begin{bmatrix} 0.9 & 0.9 \end{bmatrix}$

$w^{(1,2)} = \begin{bmatrix} 1.5 & 0 \\ 0 & 1.5 \end{bmatrix}$

0.5 0.5 0.5

$z^{(1)} = w^{(1,1)} x$

$a^{(1)} = g(z^{(1)}) = z^{(1)}$

$a^{(2)} = g(z^{(2)}) = g(w^{(2,1)} a^{(1)})$

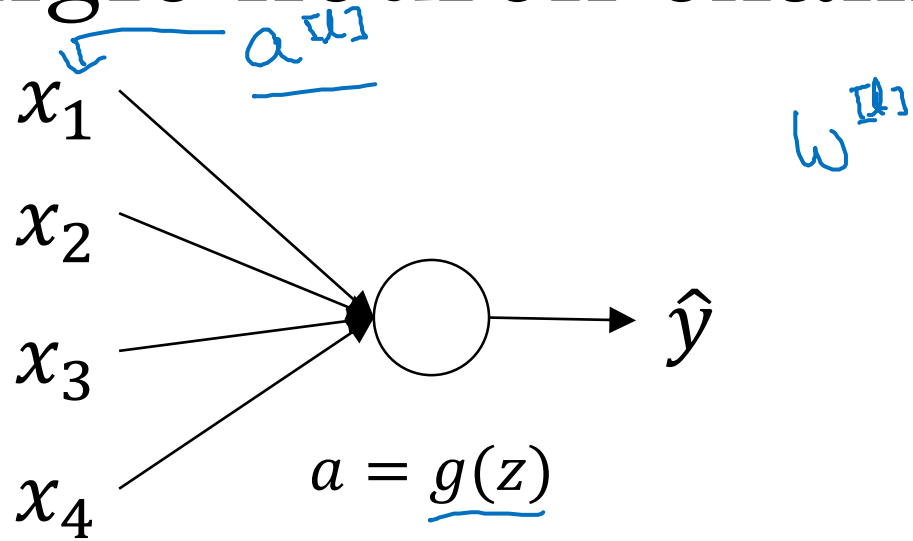
$\hat{y} = w^{(L,1)} \begin{bmatrix} 1.5 & 0 \\ 0 & 1.5 \end{bmatrix}^{L-1} x$

0.5 0.5 0.5

$1.5^{L-1} x$

$0.5^{L-1} x$

Single neuron example



$$z = w_1 x_1 + w_2 x_2 + \dots + w_n x_n$$

large $n \rightarrow$ Smaller w_i

$$\text{Var}(w_i) = \frac{1}{n} \frac{2}{n}$$

$$\underline{w^{[1]}} = \text{np.random.randn}(\text{shape}) * \text{np.sqrt}\left(\frac{2}{n^{[1-1]}}\right)$$

ReLU $g^{[2]}(z) = \text{ReLU}(z)$

Other variants:

tanh

$$\frac{1}{n^{[l-1]}}$$

Xavier initialization ↑

$$\sqrt{\frac{2}{n^{[l-1]} + n^{[1]}}}$$

↑