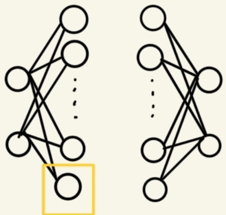
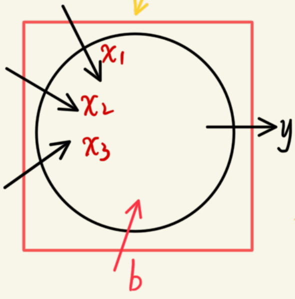
One Neuron

(mainly regression in this page, classication in in next page)





activation function.

$$y = f\left(b + \sum_{i=1}^{n} w_i x_i\right)$$

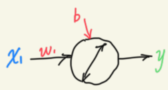
The purpose of the bias is to allow this sum be non-zero

a neuron computes

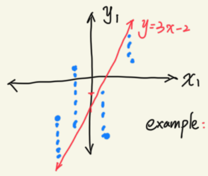
- weighted sum of inputs
- activation function

linear → Regression

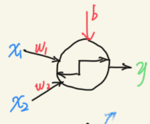
step function → Classification



$x = x_1 w_1 + b$
$y = x$

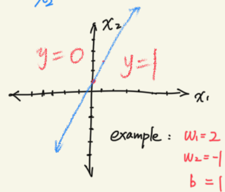


example: $w_1 = 3$, $b = 2$



$x = x_1 w_1 + x_2 w_2 + b$

$$y = \begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases}$$



example: $w_1 = 2$, $w_2 = -1$, $b = 1$

$x = 2x_1 - x_2 + 1$

$x > 0$

$2x_1 - x_2 > -1$

$2x_1 + 1 > x_2$

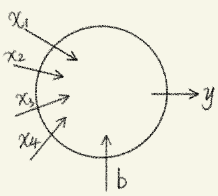
$x_2 < 2x_1 + 1$

loss function measures how wrong the model was on data set.

$$L(w_1, w_2, \dots, w_d, b) = \sum_{j=1}^{N} \left(y_j - f(\vec{x}_j)\right)^2$$

$$x = b + \sum_{i=1}^{d} w_i x_i$$

$$x_j = b + \sum_{i=1}^{d} w_i x_{ij}$$



x_{jk} is the k-th input for j-th training example.

To get a better (smaller) loss result. we shall calculate the gradient. In the case of one-dimension. we to "derivative"

$$\frac{\partial L}{\partial w_i} = \sum_{j=1}^{N} 2\left(y_j - f(\vec{x}_j)\right) \cdot \left(-\frac{\partial f(\vec{x}_j)}{\partial w_i}\right)$$

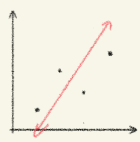
$$= -2 \sum_{j=1}^{N} \left(y_j - f(\vec{x}_j)\right) \cdot \frac{\partial f}{\partial w_i}(\vec{x}_j)$$

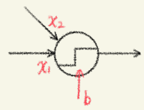
$$\frac{\partial L}{\partial w_i} = -2 \sum_{j=1}^{N} \left(y_j - f(\vec{x}_j)\right) \frac{\partial f}{\partial w_i}(\vec{x}_j)$$

$$\frac{\partial x_j}{\partial w_i} = x_{ji}$$

let's name the error of j th data point is "e_j"

x_1	y_1	f
1	1	0
2	3	2
3	2	4
4	4	6





$L(w_1 = 2,\ b = -2) = (1-0)^2 + (3-2)^2 + (2-4)^2 + (4-6)^2 = 10$

Which means: calculating our partial derivative. just means sum up the errors, times the derivative of activation.

$$\frac{\partial L}{\partial w_1} = -2\sum_{j=1}^{N} e_j \frac{\partial f}{\partial w_1} = -2\sum_{j=1}^{N} e_j x_{j1}$$

$$\frac{\partial L}{\partial w_1} = -2\sum_{j=1}^{N} e_j x_{j1} = -2[1 \cdot 1 + 1 \cdot 2 - 2 \cdot 3 - 2 \cdot 4] = 22$$

$$\frac{\partial L}{\partial b} = -2[1 + 1 - 2 - 2] = 4$$

Now, we have both partial derivatives, so we can put them together into a vector, give us the gradient.

$$\nabla L = \begin{bmatrix} \frac{\partial L}{\partial w_1} \\ \frac{\partial L}{\partial b} \end{bmatrix} = \begin{bmatrix} 22 \\ 4 \end{bmatrix}$$

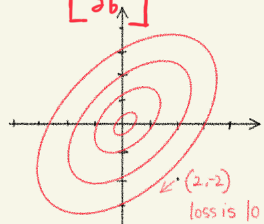
$\eta = 0.01$ ⇒ $w = 2 - 0.22$, $b = -2 - 0.04$

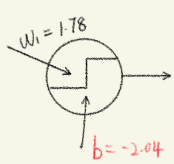
(it calls "eta")

We try to find the point within this space, that minimizes loss function.

When we calculate the gradient at a particular point in the space, that tells us, from this point, in which direction is the loss most steeply increasing.

So if we take a step in the direction opposite the gradient, then we should be able to decrease the loss.





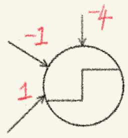
Classification

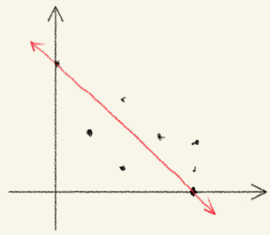
$$y = \begin{cases} 0 & x \le 0 \\ 1 & x > 0 \end{cases}$$

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

When we take the derivative of original activation function, the result always going to be ZERO, so we can't get any information.

We solve this by replacing the original function with a smooth approximation, known as "sigmoid".





Data:

x_1	x_2	y	f
1	2	0	0.27
2	1	0	0.27
2	3	1	0.73
3	2	1	.73
4	1	0	.73
4	2	1	.88
← i →			

$$\frac{d\sigma}{dx} = \frac{-(-e^{-x})}{(1+e^{-x})^2} = \sigma(x)(1-\sigma(x))^2$$

$$\frac{\partial L}{\partial w_2} = -2\sum_{j=1}^{N} e_j \sigma(x_j)(1-\sigma(x_j))x_{j2}$$

Contact: pengyuc@email.sc.edu

Geography Department,
University of South Carolina