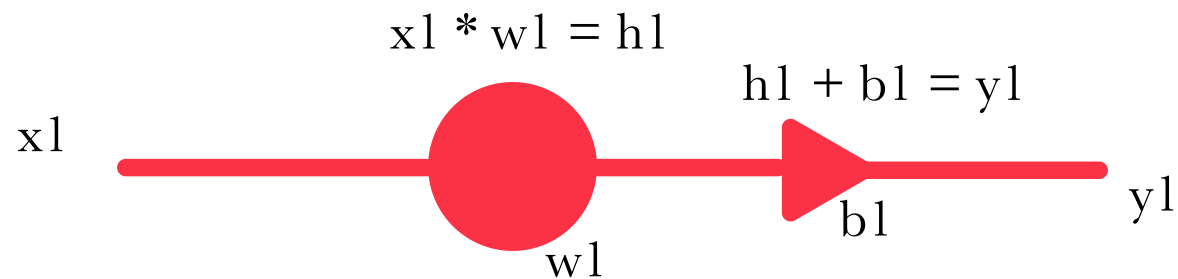


RETI NEURALI

1. rete neurale da 1 nodo

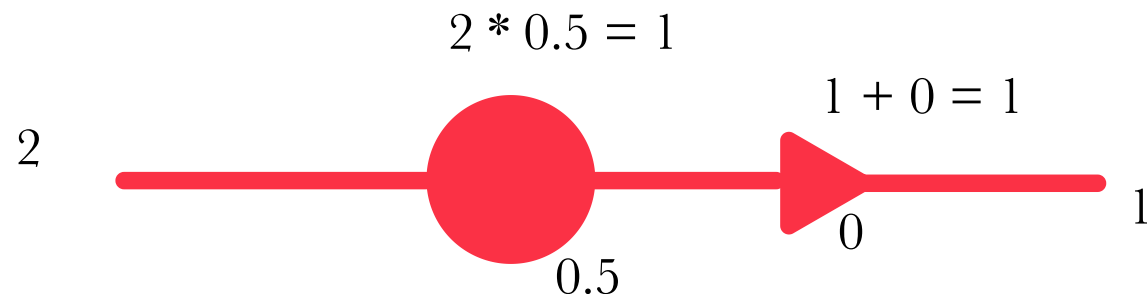


Richiesta:

Vogliamo creare una rete neurale della funzione $y=2x$, essendo una funzione lineare riusciamo a crearla anche attraverso un solo nodo

- $x = 2$, $w1 = 0.5$ (peso casuale iniziale),
 $b1 = 0$ (ipotesi semplificativa), target = 4

Forwarding pass



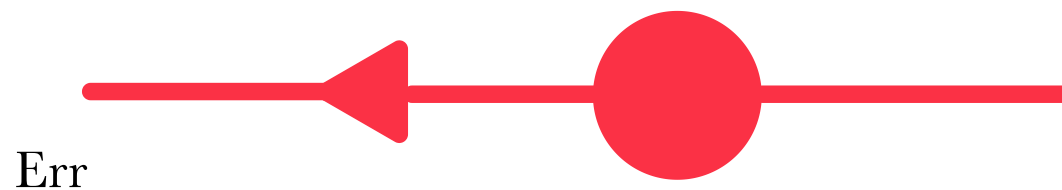
Abbiamo ottenuto un risultato pari a 1

- Errore lineare(per semplicità) = $y - \text{target} = 1 - 4 = -3$

Back propagation

Correggiamo il peso w dell' unico nodo andando a moltiplicare il suo errore in output per la derivata della sua formula di forward rispetto a w ottenendo così il gradiente

$$h' = \text{deriv}(x * w) \\ = x$$



$U(\text{gradiente del nodo}) = \text{Err} * \text{derivata parziale di tutta l'equazione di } y \text{ rispetto al peso del nodo di cui stiamo calcolando il gradiente} = \text{Err} * x = -3 * 2 = -6$

Per correggere il peso w dovremmo sottrarre al peso stesso il gradiente appena calcolato, ma così facendo sarebbe troppo brusco, infatti il gradiente lo andiamo a moltiplicare per una variabile chiamata learning rate, lr pari a 0,1 in questo caso, che imposta la velocità e la precisione con la quale i pesi vengono corretti

$$w_{\text{new}} = w - (lr * \text{grad}(w)) = 0,5 - (0,1 * -6) = 1,1$$

Forwarding pass step 2

$$y_{\text{new}} = 2 * 1,1 + 0 = 2,2$$

$$\text{Err}_{\text{new}} = 2,2 - \text{target} = 2,2 - 4 = -1,8$$

L'errore è diminuito, ma dobbiamo andare avanti così finchè il modello non è abbastanza preciso per la predizione della funzione $y = 2x$

Forwarding pass steps

$$\text{grad} = -1,8 * 2 = -3,6$$

$$w_{\text{new}} = 1,1 - (0,1 * (-3,6)) = 1,46$$

...

$$y = 2 * 1,46 = 2,92$$

$$\text{err}_{\text{new}} = 2,92 - 4 = -1,08$$

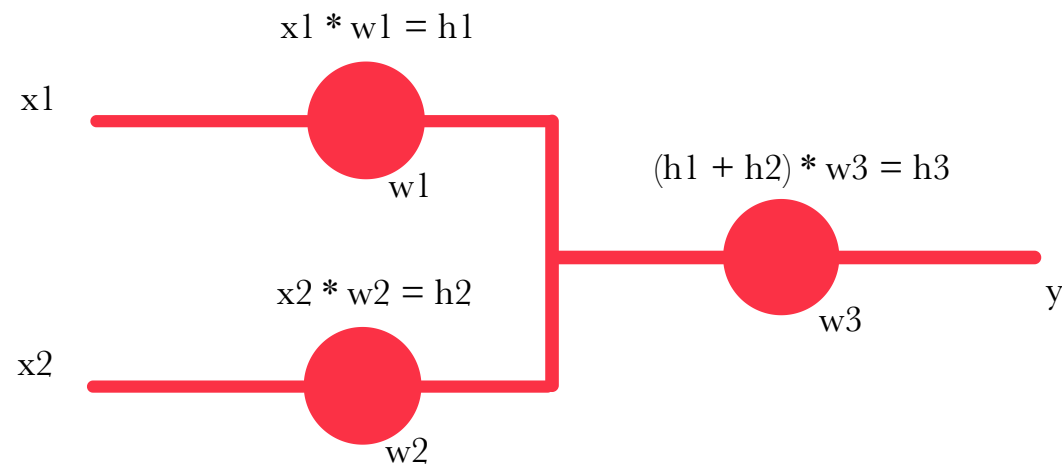
$$\text{grad} = -1,08 * 2 = -2,16$$

$$w_{\text{new}} = 1,46 - (0,1 * (-2,16)) = 1,676$$

...

2. rete neurale multinodo

- ipotizziamo $b = 0$ per semplicità



Richiesta:

modello che determina se un appartamento di $x1$ mq e con prezzo $x2$ e più o meno un'opportunità, y da 0.0 a 1.0.

I mq vanno da 0 a 100 normalizzati da 0.0 a 1.0, stessa cosa con il prezzo da 0 a 100.000

- $w1 = w2 = w3 = 0.5$
- $lr = 0.1$

Forward pass (training entry #1)

- $x1 = 1.0, x2 = 0.3$
- $target = 0.2$

$$h = w * x + b$$

$$h1 = 1.0 * 0.5 = 0.5$$

$$h2 = 0.3 * 0.5 = 0.15$$

$$y = (h1 + h2) * w3 = 0.65 * 0.5 = 0.325$$

$$Err = y - target = 0.325 - 0.2 = 0.125 \text{ (previsione alta)}$$

Back propagation (training entry #1)

$$\begin{aligned}\text{gradiente nodo 3} &= \text{Err} * \text{deriv_w3}(\text{espressione forward pass}) \\ &= \text{Err} * f'((h1 + h2) * w3) = 0.125 * (0.65) = 0.081\end{aligned}$$

$$\begin{aligned}\text{grad 1} &= \text{Err output nodo 1} * \text{deriv_w1}(\text{espressione nodo 1}) \\ &= (\text{Err} * w3) * x1 = 0.125 * 0.5 * 1.0 = 0.062\end{aligned}$$

$$\text{grad 2} = 0.125 * 0.5 * 0.3 = 0.018$$

$$w3_{\text{new}} = 0.5 - (0.1 * 0.081) = 0.492$$

$$w1_{\text{new}} = 0.5 - (0.1 * 0.062) = 0.494$$

$$w2_{\text{new}} = 0.5 - (0.1 * 0.018) = 0.498$$

Forward pass (training entry #2)

- $x1 = 0.8, x2 = 0.8$
- $\text{target} = 0.85$

$$h1 = 0.8 * 0.494 = 0.395$$

$$h2 = 0.8 * 0.498 = 0.398$$

$$y = (h1 + h2) * w3 = 0.793 * 0.492 = 0.390$$

$$\text{Err} = y - \text{target} = 0.390 - 0.85 = -0.46 \text{ (previsione bassa)}$$

Back propagation (training entry #2)

$$\text{grad } 3 = -0.46 * 0.793 = -0.364$$

$$\text{grad } 1 = -0.46 * 0.492 * 0.8 = -0.181$$

$$\text{grad } 2 = -0.46 * 0.492 * 0.8 = -0.181$$

$$w3_new = 0.492 - (0.1 * -0.364) = 0.528$$

$$w1_new = 0.494 - (0.1 * -0.181) = 0.512$$

$$w2_new = 0.498 - (0.1 * -0.181) = 0.516$$

Abbiamo così finito un'epoca cioè il training su tutto il nostro set per un intero step, ora per allenare il modello dovremmo continuare a fare questo processo fin quando non otteniamo un buon margine d'errore sui due esempi senza sforare nell' overfitting

