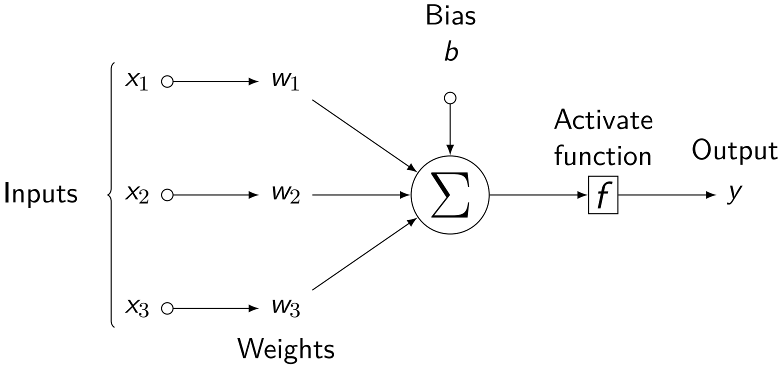
Data Mining Lectures - Neural Networks

Piotr Wasiewicz

Institute of Computer Science

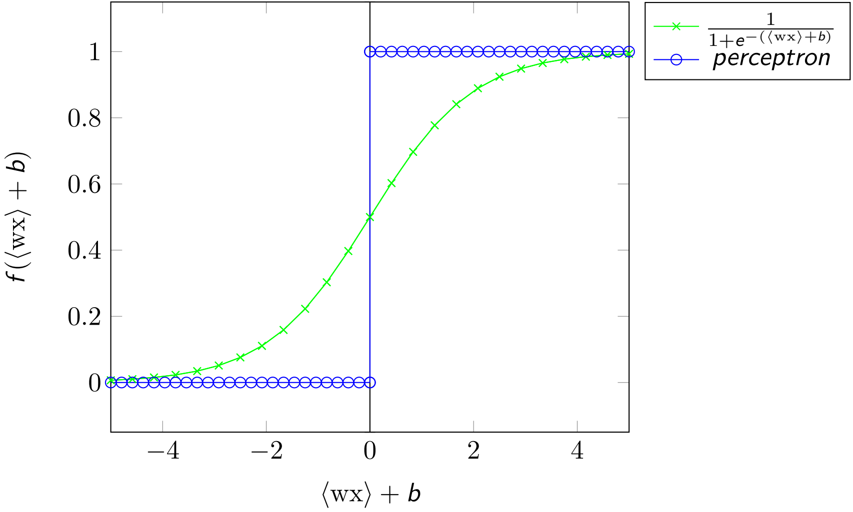
pwasiewi@elka.pw.edu.pl

11 czerwca 2017



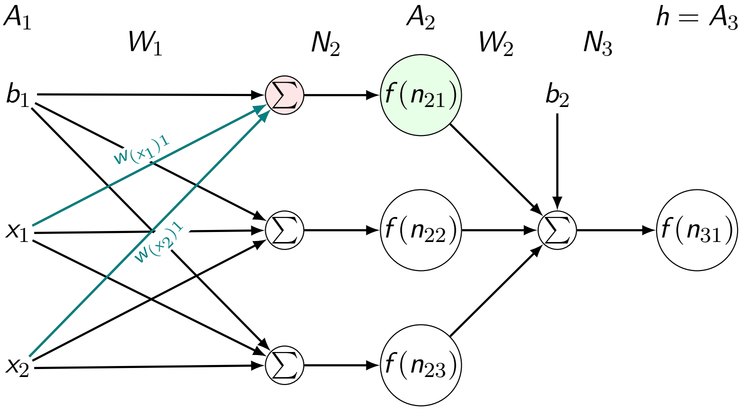
$$y = f(\langle \mathrm{wx} \rangle + b) = f\Big(\sum_{i \in \{1,2,3\}} w_i x_i + b\Big)$$

Neural Networks: Sigmoid function



The sigmoid function $f(a)$ and its derivative $f'(a) = f(a) * (1 - f(a))$ has the main role in error backpropagation.

Neural Networks: neuron layers



$$a_{21} = f(n_{21}) = f(w_{(x_1)1}x_1 + w_{(x_2)1}x_2 + b_{11})$$

Approximation Error

Mean Squared Error

$$E_{\text{total}} = \sum_{p \in T} E^{p}_{MSE} \quad \text{where } p \text{ is an element in a training set } T$$

$$E^{h}_{MSE} = \frac{1}{2} \sum_{j \in M} \left(\hat{y}_h - h\right)^2$$

Backpropagation of error to weights

$$\triangle\, w_{mh} \sim -\nabla_w \cdot\ E^{h}_{MSE}$$

where w_{mh} is the connection weight from the neuron m to the neuron h.

Error backpropagation to the output neuron weights

The gradient of MSE influences the weights of connections to the output neuron

$$\triangle\ w_{mh} = -\eta \frac{\partial E\left(w_{mh}\right)}{\partial w_{mh}}$$

where η is the training coefficient.

The backpropagation algorithm search for the minimum of the error function in weight

$$\frac{\partial E\left(w_{mh}\right)}{\partial w_{mh}} = \frac{\partial E}{\partial N_3} \cdot \frac{\partial N_3}{\partial w_{mh}} = -\delta_h \cdot \frac{\partial N_3}{\partial w_{mh}}$$

The derivative chain rule

$$\frac{\partial E\left(w_{mh}\right)}{\partial w_{mh}} = \frac{\partial E}{\partial N_3} \cdot \frac{\partial N_3}{\partial w_{mh}} = \frac{\partial E}{\partial h} \cdot \frac{\partial h}{\partial N_3} \cdot \frac{\partial N_3}{\partial w_{mh}}$$

Error backpropagation to the output neuron weights

The partial derivative of E^h_{MSE} with respect to the net output h

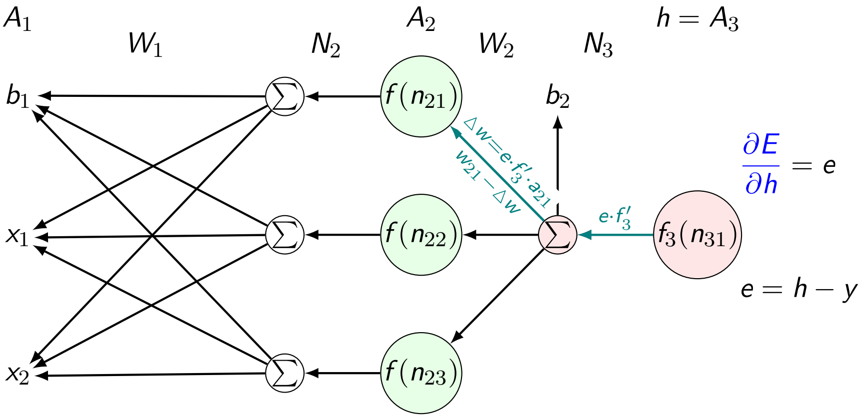
$$\frac{\partial E}{\partial h} = \frac{\partial}{\partial h} \cdot \frac{1}{2} \sum_{j \in M} \left(\hat{y}_h - h\right)^2 = -\left(\hat{y}_h - h\right) = \left(h - \hat{y}_h\right)$$

The partial derivative of the sigmoid function

$$\frac{\partial h}{\partial N_3} = \frac{\partial f(N_3)}{\partial N_3} = h(1-h)$$

The partial derivative of the dot product $N_3 = \langle A_2 W_2 \rangle$

$$\frac{\partial N_3}{\partial w_{mh}} = a_{2m}$$



$$\triangle\, w_{mh} = \eta \cdot \frac{\partial E\,(w_{mh})}{\partial w_{mh}} = \eta \cdot \frac{\partial E}{\partial h} \cdot \frac{\partial h}{\partial N_3} \cdot \frac{\partial N_3}{\partial w_{mh}}$$

$$\triangle\, W_2 = \eta \cdot (h - \hat{y}_h) \cdot h(1-h) \cdot A_2$$

Error backpropagation to the output neuron weights

The partial derivative of E^h_{MSE} with respect to the output neuron weights

$$\frac{\partial E\left(w_{mh}\right)}{\partial w_{mh}} = -\delta_h \cdot \frac{\partial N_3}{\partial w_{mh}} = -\delta_h \cdot a_{2m} = (h - \hat{y}_h) \cdot h(1-h) \cdot a_{2m}$$

Widrow-Hoff rule – δ-rule

$$\boxed{\triangle\ w_{mh} = \eta \cdot \delta_h \cdot a_{2m}} \qquad \boxed{\triangle\ W_{mh} = \eta \cdot \delta_h \cdot A_2}$$

R code

```
delta3 = (h - y)*h*(1 - h)
W2 <- W2 - alfa*delta3 %*% t(A2)
```

Error backpropagation to the hidden neuron weights

$$\frac{\partial E\left(w_{nm}\right)}{\partial w_{nm}} = \frac{\partial E}{\partial N_2} \cdot \frac{\partial N_2}{\partial w_{nm}} = \frac{\partial E}{\partial A_2} \cdot \frac{\partial A_2}{\partial N_2} \cdot \frac{\partial N_2}{\partial w_{nm}}$$

The partial derivative of E_{MSE}^m with respect to the hidden layer outputs A_2 ($h_1 = h$ - it is only one output neuron)

$$\frac{\partial E}{\partial A_2} = \frac{\partial(E_{h_1} + E_{h_2})}{\partial A_2} = \frac{\partial E^h}{\partial N_3} \cdot \frac{\partial N_3}{\partial A_2} = -\delta_h \cdot W_2$$

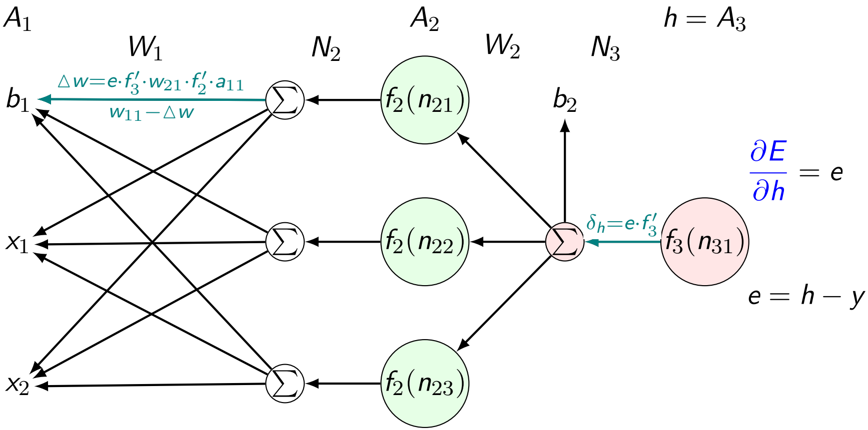
The partial derivative of the sigmoid function

$$\frac{\partial A_2}{\partial N_2} = \frac{\partial f(N_2)}{\partial N_2} = A_2(1 - A_2)$$

The partial derivative of the dot product $N_2 = \langle A_1 W_1 \rangle$

$$\frac{\partial N_2}{\partial w_{nm}} = a_{1n}$$

Neural Networks: error backpropagation



$$\triangle\, w_{nm} = \eta \cdot \frac{\partial E\,(w_{nm})}{\partial w_{nm}} = \eta \cdot \frac{\partial E}{\partial N_2} \cdot \frac{\partial N_2}{\partial w_{nm}} = \eta \cdot \frac{\partial E}{\partial A_2} \cdot \frac{\partial A_2}{\partial N_2} \cdot \frac{\partial N_2}{\partial w_{nm}}$$

$$\triangle\, W_1 = \eta \cdot \delta_m \cdot A_1 = \delta_h \cdot W_2 \cdot A_2(1 - A_2) \cdot A_1$$

Error backpropagation to the hidden neuron weights

The partial derivative of E_{MSE}^{m} with respect to the hidden neuron weights

$$\frac{\partial E\left(w_{nm}\right)}{\partial w_{nm}} = -\delta_m \cdot \frac{\partial N_2}{\partial w_{nm}} = -\delta_m \cdot a_{1n} = -\delta_h \cdot W_2 \cdot A_2(1 - A_2) \cdot a_{1n}$$

Widrow-Hoff rule – δ-rule for the hidden layer

$$\boxed{\triangle\, w_{nm} = \eta \cdot \delta_m \cdot a_{1n} = \eta \cdot \delta_h \cdot W_2 \cdot A_2(1 - A_2) \cdot a_{1n}}$$

$$\delta_m = \delta_h \cdot W_2 \cdot A_2(1 - A_2) \qquad \boxed{\triangle\, W_1 = \eta \cdot \delta_m \cdot A_1}$$

R code

```
delta3 = (h - y)*h*(1 - h)
delta2<-(t(W2) %*% delta3 * A2 * (1 - A2))[-1]
W1 <- W1 - alfa*delta2 %*% t(A1)
```