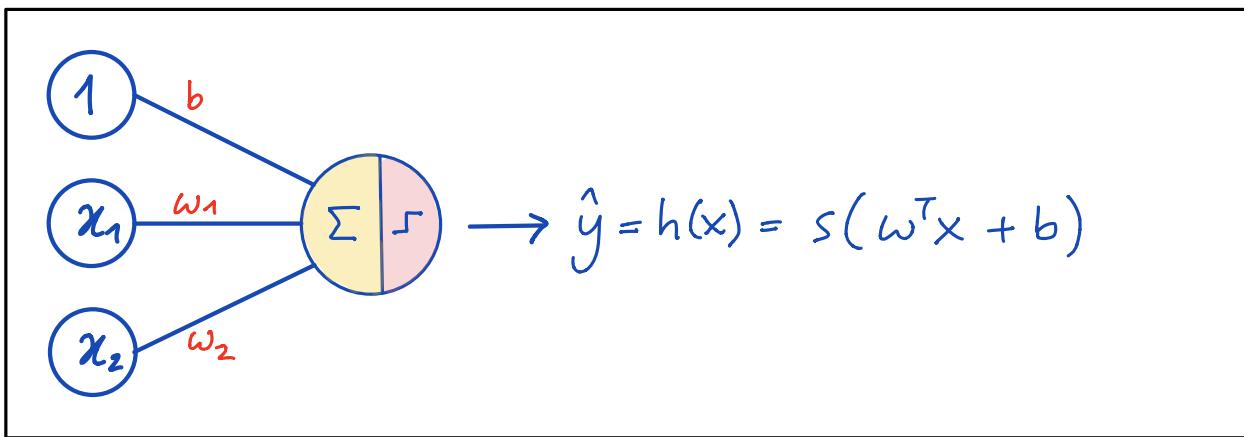


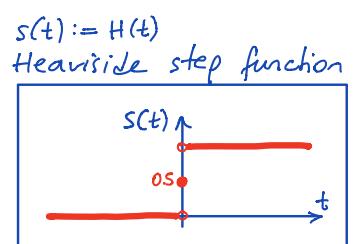
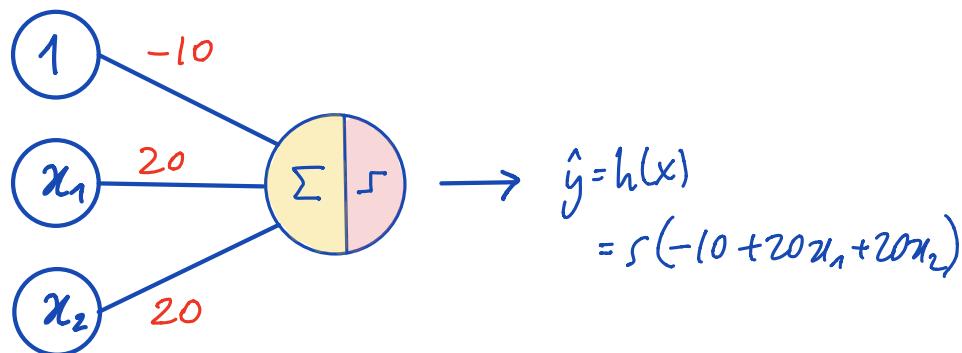
• Multilayer Perzeptron (MLP)

- Das Perzepron = Einschichtiges Neuronales Netzwerk
(auch : Linear Threshold Unit (LTU))



- Grundbaustein von mehrschichtigen Netzwerken.
- Kann nur lineare Beziehungen lernen
(z.B. einfache Logik Funktionen)

BSP.



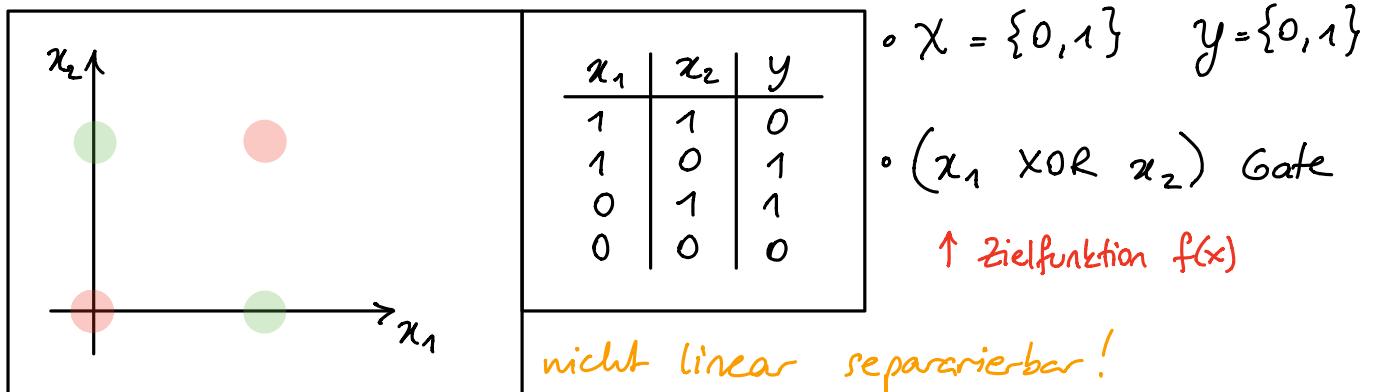
x_1	x_2	y
1	1	
0	1	
1	0	
0	0	

- $x_1 \text{ AND } x_2$
- $(\text{NOT } x_1) \text{ OR } (\text{NOT } x_2)$
- $x_1 \text{ OR } x_2$
- $(\text{NOT } x_1) \text{ AND } (\text{NOT } x_2)$

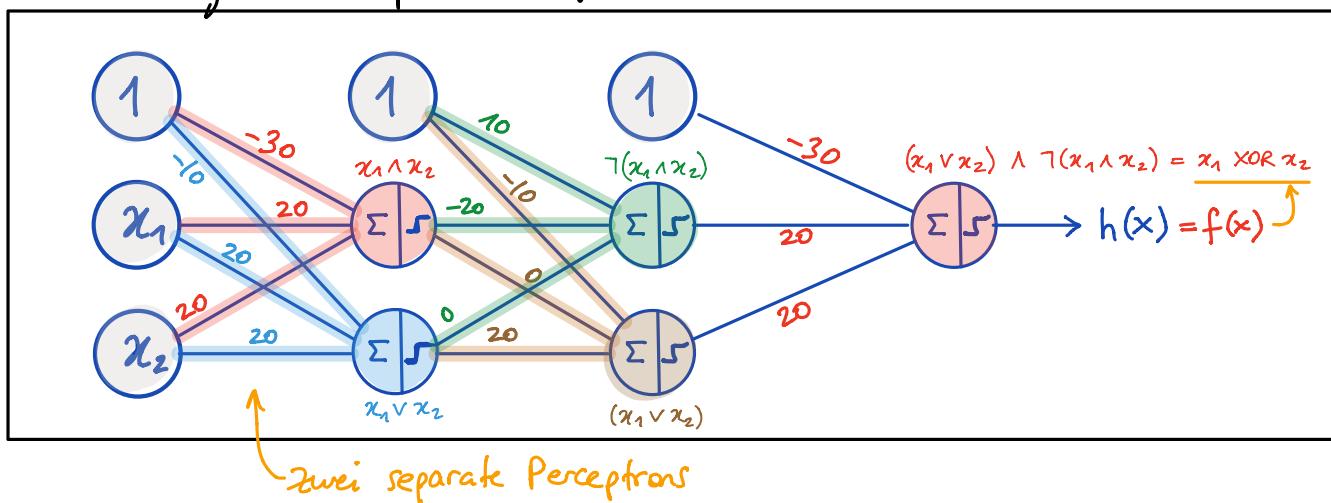
- Das Aneinanderreihen mehrerer Perzeptoren kann diese Einschränkung aufheben

BSP.

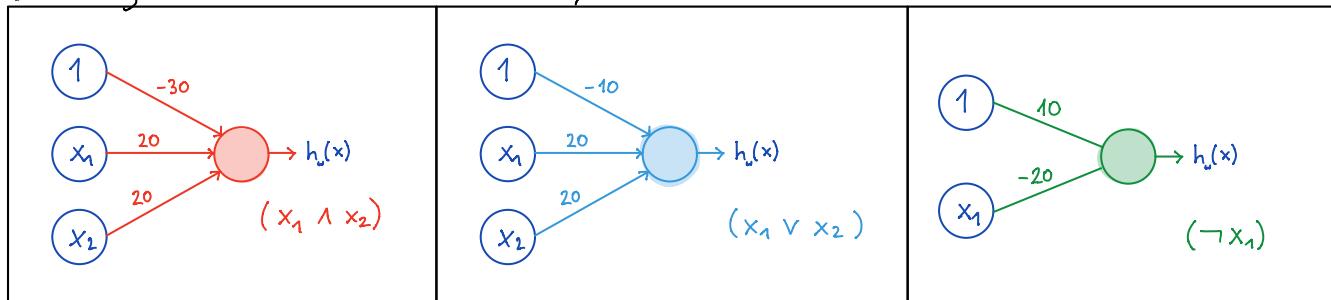
$$y = x_1 \text{ XOR } x_2 = (x_1 \text{ OR } x_2) \text{ AND } \neg(x_1 \text{ AND } x_2)$$



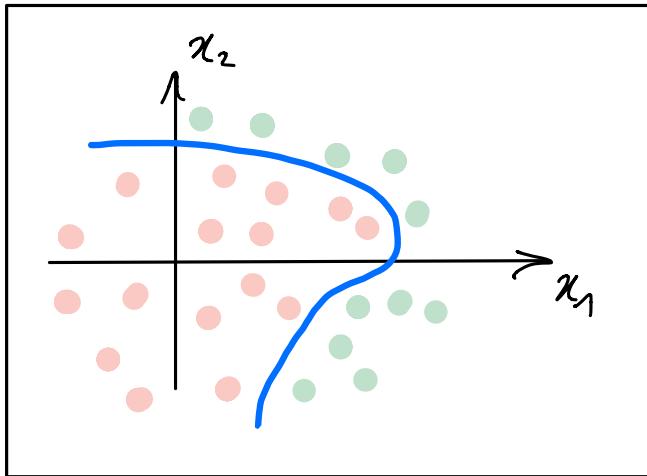
Multi Layer Perzeptron für XOR Gate



Building Blocks (Grundbaustein)



• Warum Neuronale Netzwerke?



- Nichtlineare Grenzen mit Logistischer Regression durch das Einführen neuer Merkmale möglich:

$$x_1^2, x_2^2, x_1 \cdot x_2, \dots$$

$$x_1^3, x_2^3, x_1^2 x_2, x_1 x_2^2, \dots$$

- Warum brauchen wir tiefe Netzwerke?

BSP. x_1, x_2, \dots, x_{100} (ursprüngliche Merkmale)

$x_1 x_2, x_1 x_3, \dots, x_1 x_{100}, x_2 x_3, x_2 x_4, \dots, x_2 x_{100}, \dots$ (quadr. Terme)

$x_1 x_2 x_3, x_1^2 x_3, x_1 x_5 x_{17}, \dots$

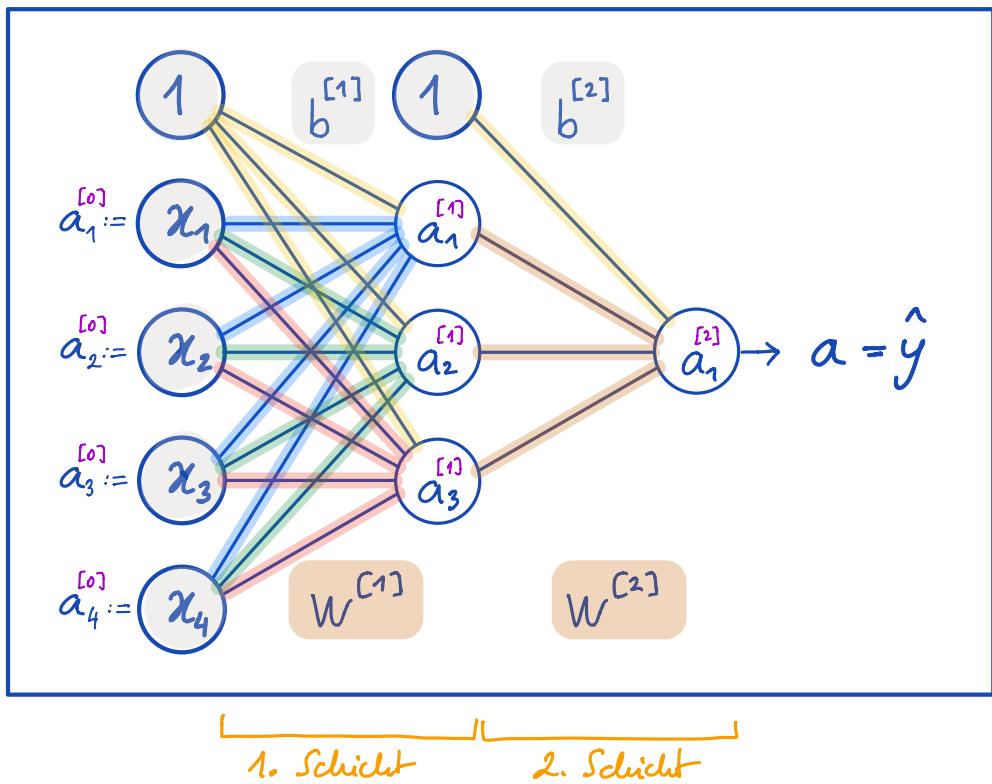
→ ≈ 180000

BSP. 50×50 px Graustufenbilder → $n = 2500$

Graustufenwert 0-255

$$x = \begin{bmatrix} x_1 \\ \vdots \\ x_{2500} \end{bmatrix} \in \mathbb{R}^{2500} \rightarrow \text{mit quadratischen Termen...?}$$

Fully Connected Feed Forward Deep Neural Network



$a^{[1]}$: Aktivierungen
der 1. Schicht
(Vektor!)

$a^{[0]} := x$ (Eingabe)

$$b^{[1]} = \begin{bmatrix} b_1^{[1]} \\ b_2^{[1]} \\ b_3^{[1]} \end{bmatrix} : \text{Bias Termen der 1. Schicht.}$$

1. Schicht 2. Schicht

- Die "Aktivierungen" der ersten Schicht:

$$\begin{aligned} a_1^{[1]} &= \text{relu}\left(w_{11}^{[1]} \cdot x_1 + w_{12}^{[1]} \cdot x_2 + w_{13}^{[1]} \cdot x_3 + w_{14}^{[1]} \cdot x_4 + b_1^{[1]}\right) && \text{ehem. } w_{10} \\ a_2^{[1]} &= \text{relu}\left(w_{21}^{[1]} \cdot x_1 + w_{22}^{[1]} \cdot x_2 + w_{23}^{[1]} \cdot x_3 + w_{24}^{[1]} \cdot x_4 + b_2^{[1]}\right) && \text{ehem. } w_{20} \\ a_3^{[1]} &= \text{relu}\left(w_{31}^{[1]} \cdot x_1 + w_{32}^{[1]} \cdot x_2 + w_{33}^{[1]} \cdot x_3 + w_{34}^{[1]} \cdot x_4 + b_3^{[1]}\right) && \text{ehem. } w_{30} \end{aligned}$$

- Matrix Notation für Vektor $a^{[1]}$:

$$\begin{bmatrix} a_1^{[1]} \\ a_2^{[1]} \\ a_3^{[1]} \end{bmatrix} = \text{relu} \left[\begin{bmatrix} w_{11}^{[1]} & w_{12}^{[1]} & w_{13}^{[1]} & w_{14}^{[1]} \\ w_{21}^{[1]} & w_{22}^{[1]} & w_{23}^{[1]} & w_{24}^{[1]} \\ w_{31}^{[1]} & w_{32}^{[1]} & w_{33}^{[1]} & w_{34}^{[1]} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} b_1^{[1]} \\ b_2^{[1]} \\ b_3^{[1]} \end{bmatrix} \right]$$

$$a^{[1]} = \text{relu}(\mathbf{W}^{[1]} \cdot \mathbf{x} + \mathbf{b}^{[1]})$$

$$z^{[1]} = W^{[1]} \cdot a^{[0]} + b^{[1]}$$

(3x1) (3x4) · (4x1) + (3x1)

Dimensionen?

$$a^{[1]} = \text{relu}(z^{[1]})$$

(3x1) (3x1)

$$\sigma \begin{pmatrix} \vdots \\ \vdots \\ \vdots \end{pmatrix} = \begin{pmatrix} \sigma(\cdot) \\ \sigma(\cdot) \\ \sigma(\cdot) \end{pmatrix}$$

- Matrix Notation für Vektor $a^{[2]} = \hat{y}$:

$$a_1^{[2]} = \sigma(w_{11}^{[2]} \cdot a_1^{[1]} + w_{12}^{[2]} \cdot a_2^{[1]} + w_{13}^{[2]} \cdot a_3^{[1]} + b_1^{[2]})$$

$$z^{[2]} = W^{[2]} \cdot a^{[1]} + b^{[2]} = [w_{11}^{[2]} \ w_{12}^{[2]} \ w_{13}^{[2]}] \cdot \begin{bmatrix} a_1^{[1]} \\ a_2^{[1]} \\ a_3^{[1]} \end{bmatrix} + [b_1^{[2]}]$$

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

(1x1)

- Forward Propagation (Forwärtslauf)

- $a^{[0]} := x = \text{Eingabe}$

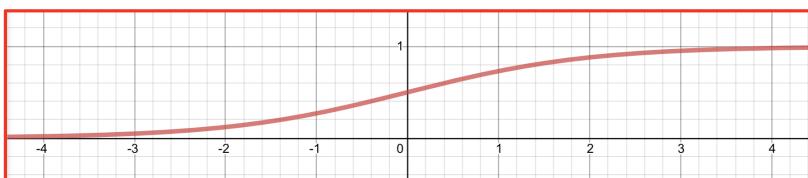
- $z^{[1]} = W^{[1]} \cdot x + b^{[1]}$

- $a^{[1]} = \text{ReLU}(z^{[1]})$

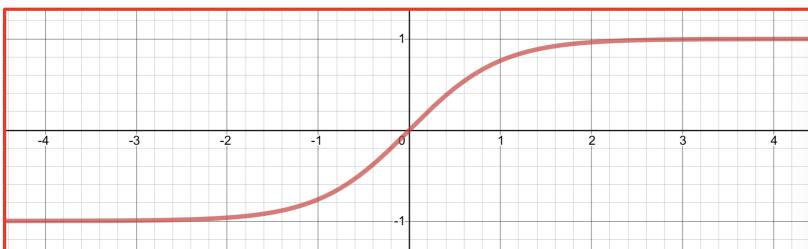
- $z^{[2]} = W^{[2]} \cdot a^{[1]} + b^{[2]}$

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

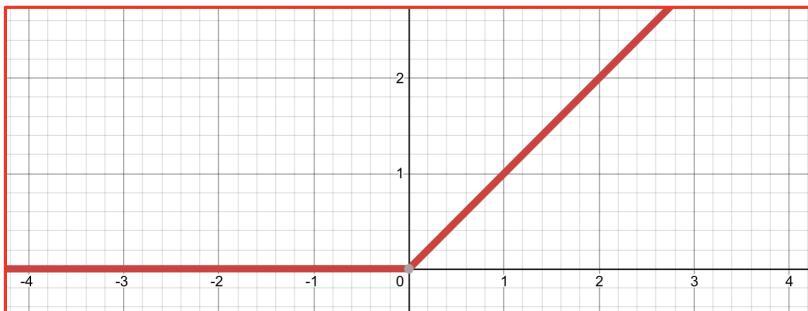
• Andere Aktivierungsfunktionen



- $\text{Sigmoid}(x) = \sigma(x)$



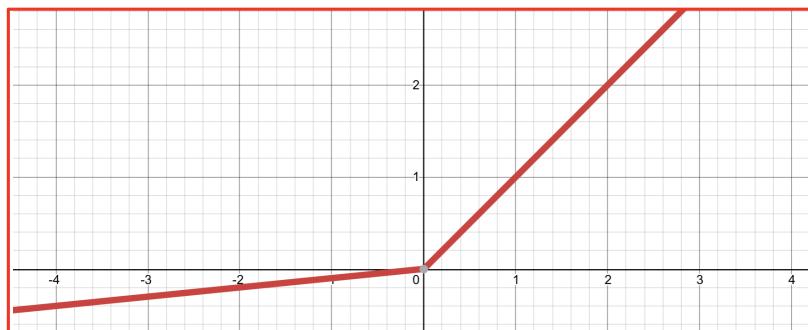
- $\tanh(x)$



- $\text{ReLU} = \max(0, x)$

$$= \begin{cases} 0, & x \leq 0 \\ x, & x > 0 \end{cases}$$

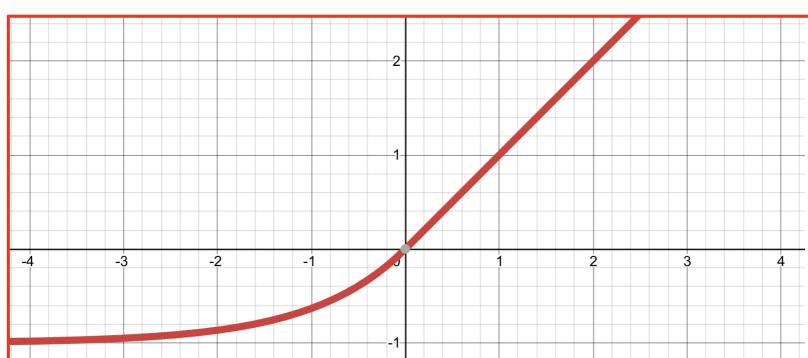
Good default choice (for hidden layers)



- Leaky ReLU

$$= \max(0.1x, x)$$

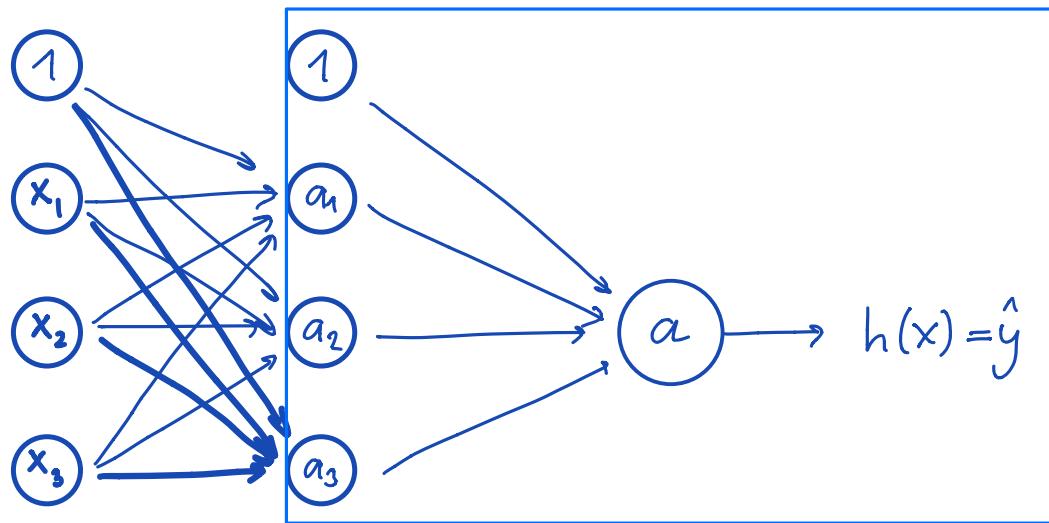
$$= \begin{cases} 0.1x, & x < 0 \\ x, & x \geq 0 \end{cases}$$



- ELU

$$= \begin{cases} \alpha(e^x - 1), & x < 0 \\ x, & x \geq 0 \end{cases}$$

- Intuition : Das Lernen von Merkmalen für Logistische Regression

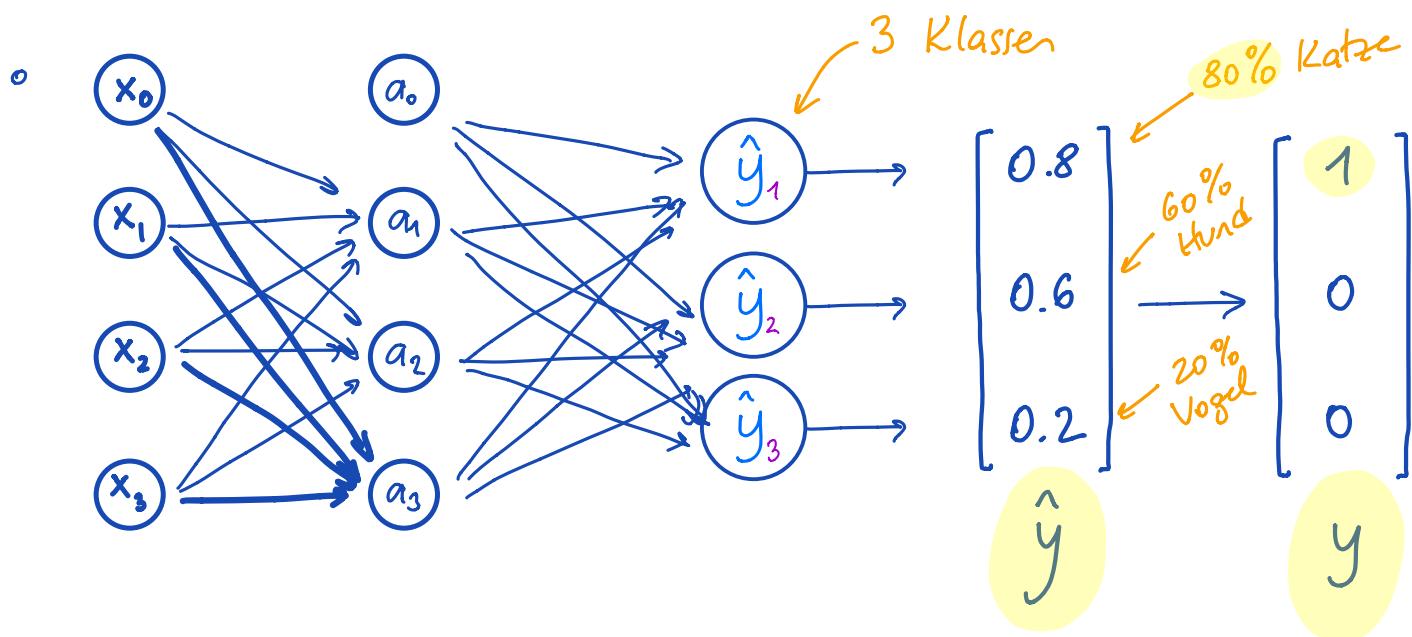
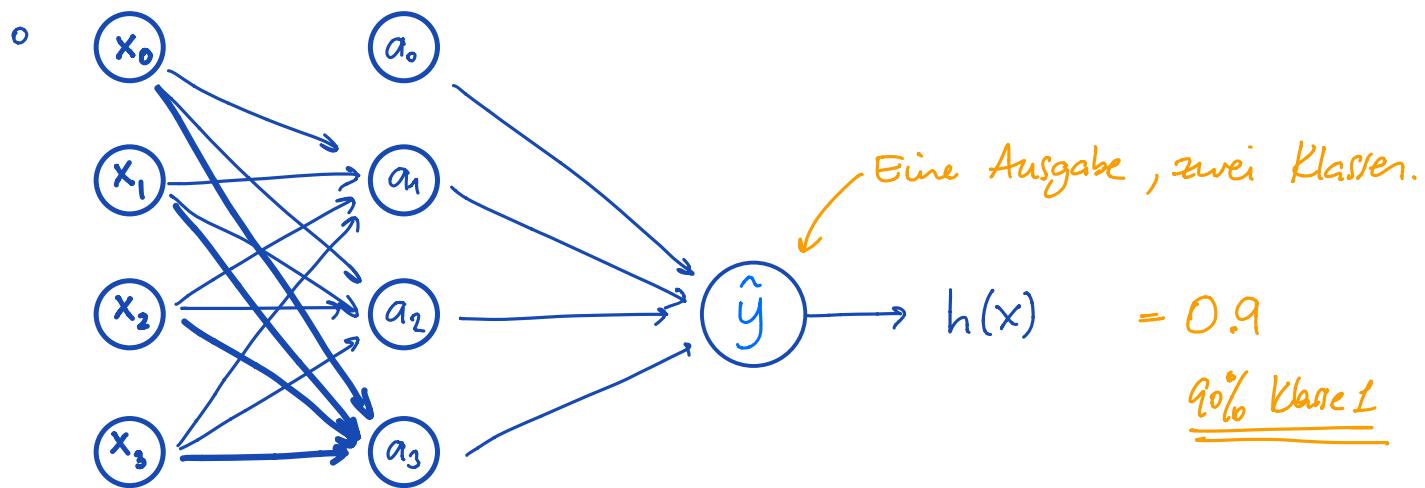


Logistische Regression
mit Merkmalen a_1, a_2, a_3

$$\hat{y} = \sigma(\omega_{11}a_1 + \omega_{12}a_2 + \omega_{13}a_3 + b_1)$$

→ Das Netzwerk lernt eigene Merkmale ($a^{[1]}$) für logistische Regression

• Mehrklassige Klassifizierung (one-vs-all)



- Kostenfunktion (L Schichten, m Beispiele, K Klassen)

$$J(\omega) = \frac{-1}{m} \left[\sum_{i=1}^m \left[\sum_{k=1}^K y_k^{(i)} \cdot \log(\hat{y}_k^{(i)}) + (1 - y_k^{(i)}) \cdot \log(1 - \hat{y}_k^{(i)}) \right] \right]$$

$$+ \frac{\lambda}{2m} \sum_{\ell=1}^{L-1} \sum_{i=1}^{s_\ell} \sum_{j=1}^{s_{\ell+1}} (\omega_{ji}^{(\ell)})^2$$

All weights in all layers