Soorten Machine Learning

targets: \ type:	Supervised	Unsupervised
Nominaal Categorisch	Classificatie One-R NN	Clustering NN? k-means
Numeriek Continu	Regressie NN lineaire regressie	Dimensie reductie (NN) PCA

Machine learning: modellen/algoritmen die beter presteren naarmate ze aan meer (trainings)data worden blootgesteld ("ervaring")

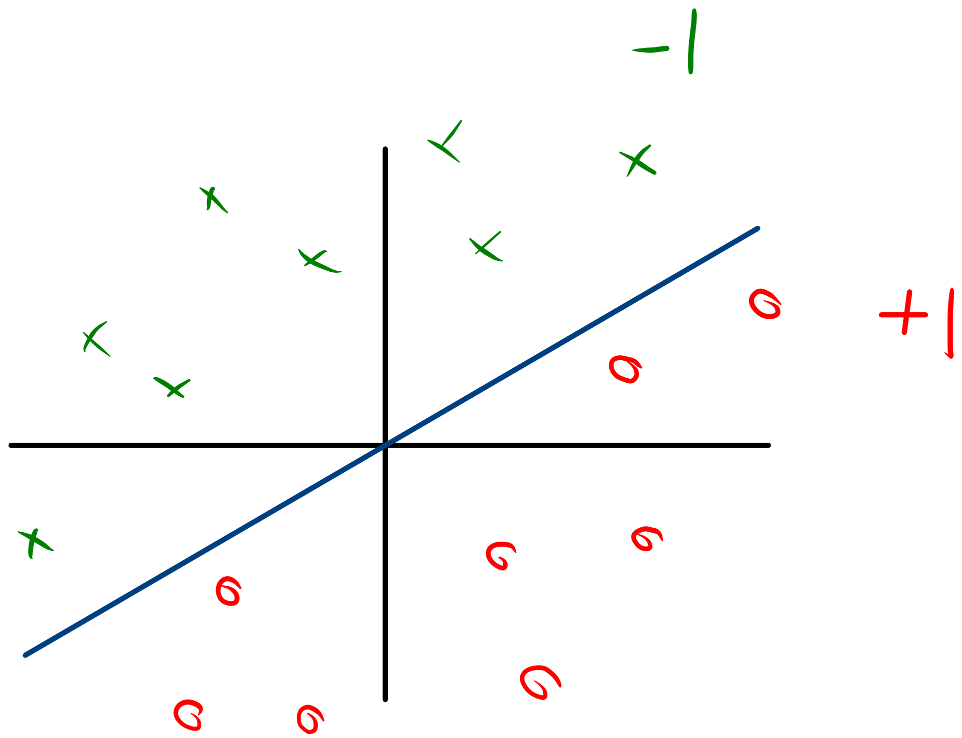
Deep learning: modellen/algoritmen die hierarchisch zijn opgezet ("in lagen")

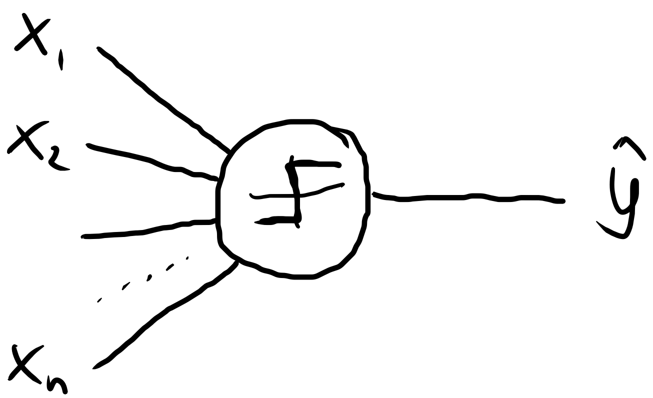
Rosenblatt's perceptron

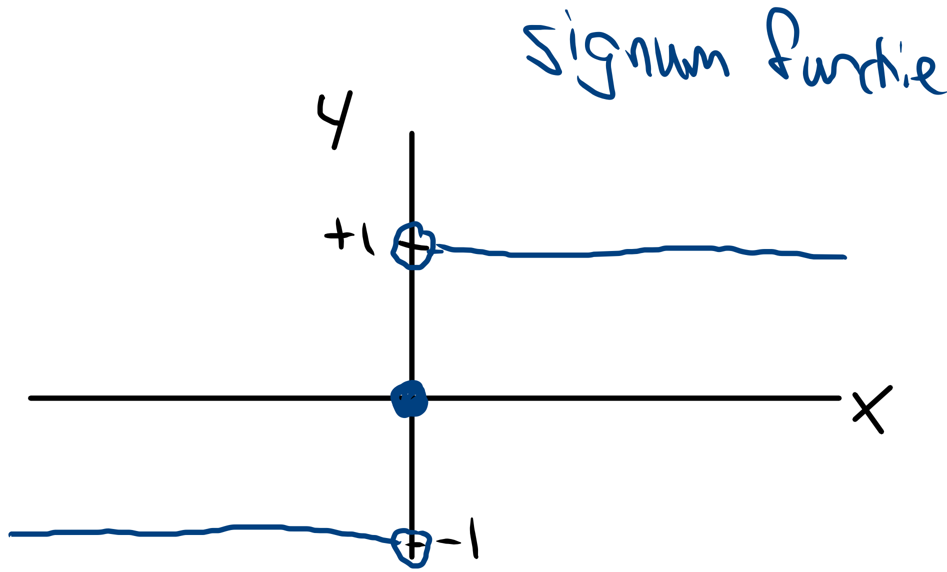
Classificatiemodel die twee klassen perfect kan onderscheiden op basis van numeriek attributen als ze lineair separabel zijn

data:

x_1	x_2	y
0,3	-1,6	-1
2,5	8,7	+1
0,0	3,14	+1



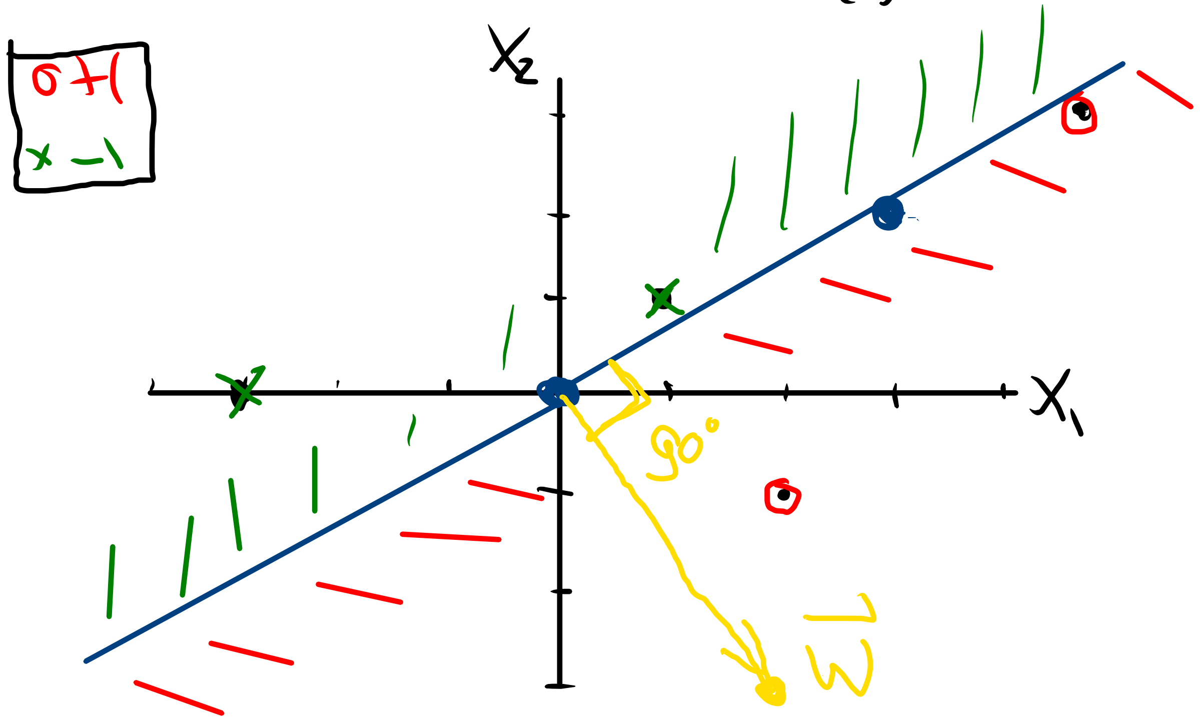




$$\hat{y} = \mathrm{sgn}\left(\overbrace{w_1 \cdot x_1 + w_2 \cdot x_2 + w_3 \cdot x_3 + \ldots + w_n \cdot x_n}^{\text{lineaire combinatie}} \right)$$

$\hat{y} = \mathrm{sgn}(w_1 x_1 + w_2 x_2)$

Stel $w_1 = 2$ $w_2 = -3$ $\boxed{\vec{w}} = \begin{bmatrix} 2 \\ -3 \end{bmatrix}$



Wat voor label kent het perceptron toe aan:

$x_1 = 1$, $x_2 = 1$ $\bar{x} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ $\hat{y} = -1$

$\bar{x} = \begin{bmatrix} -3 \\ 0 \end{bmatrix}$ $\hat{y} = -1$

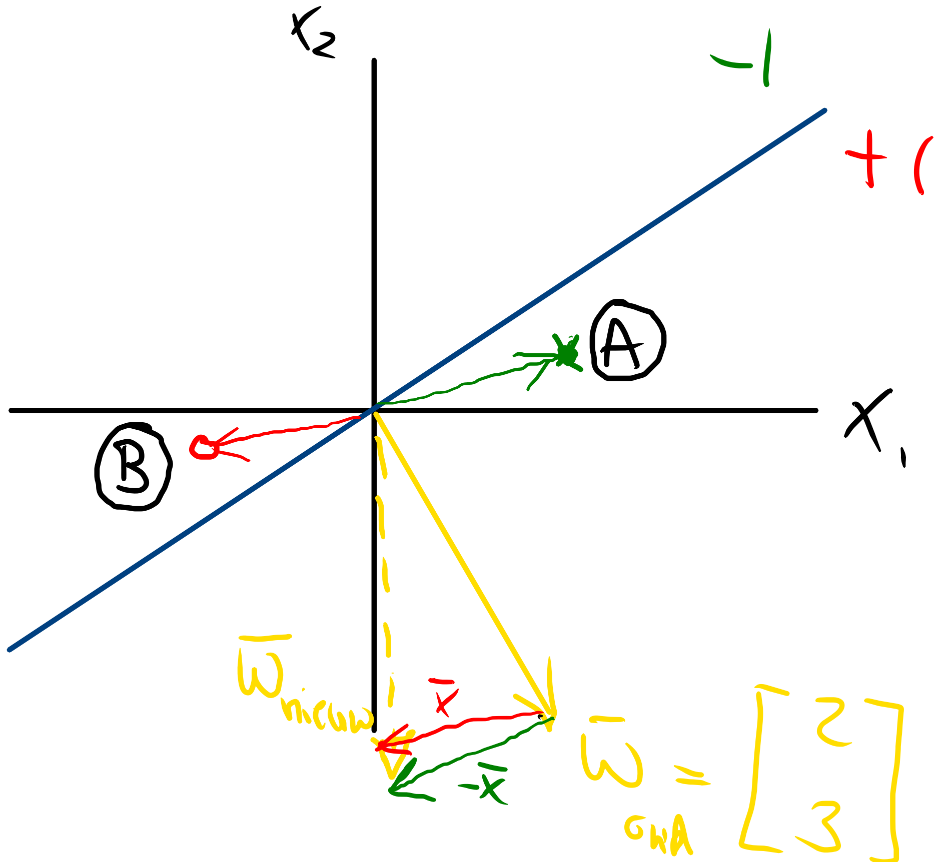
$\bar{x} = \begin{bmatrix} 2 \\ -1 \end{bmatrix}$ $\hat{y} = +1$

$\bar{x} = \begin{bmatrix} 5 \\ 3 \end{bmatrix}$ $\hat{y} = +1$

$\bar{x} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ $\hat{y} = 0$

$\bar{x} = \begin{bmatrix} 3 \\ 2 \end{bmatrix}$ $\hat{y} = 0$

De gewichten w beschrijven een vector die precies loodrecht op de classificatiegrenslijn staat, en wijst naar de richting van de +1 klasse.



A: $y = -1 \qquad \hat{y} = +1$

$$\bar{w}_{nieuw} = \bar{w}_{oud} - 2 \cdot \bar{x}$$

$w_1 \leftarrow w_1 - 2x_1$

$w_2 \leftarrow w_2 - 2x_2$

B: $y = +1 \qquad \hat{y} = -1$

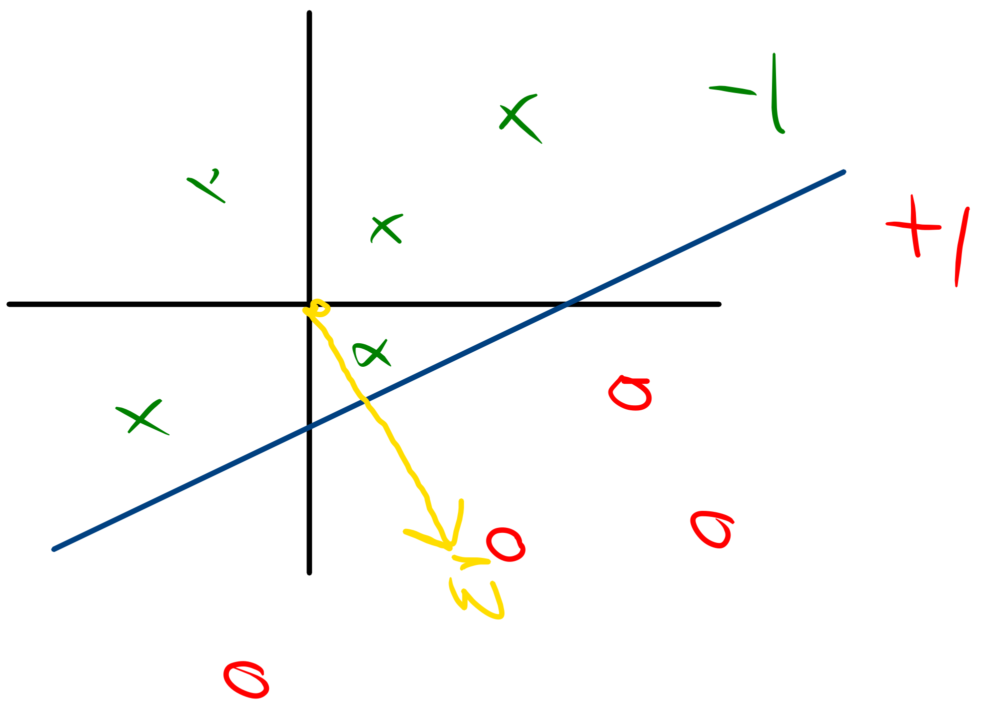
$$\bar{w}_{nieuw} = \bar{w}_{oud} + 2\bar{x}$$

$w_1 \leftarrow w_1 + 2x_1$

$w_2 \leftarrow w_2 + 2x_2$

Samen:

$$\boxed{\bar{w}_{nieuw} = \bar{w}_{oud} - (\hat{y} - y) \cdot \bar{x}}$$



$$\hat{y} = \mathrm{sgn}(b + w_1 x_1 + w_2 x_2 + \ldots + w_d x_d)$$

↑ bias ↑ gewichten $\bar{w}$

Trucje:

Doe net alsof er een extra attribuut is met waarde 1 voor elke instance

x_0	x_1	x_2	$\hat{y}$
1	12	3,5	+1
1	-8	+7,2	-1

"oude" perceptron:

$$\hat{y} = \mathrm{sgn}(\underbrace{w_0}_{=b} \cdot \underbrace{x_0}_{=1} + w_1 \cdot x_1 + \ldots + w_d \cdot x_d)$$

Update-regel voor het perceptron mét bias:

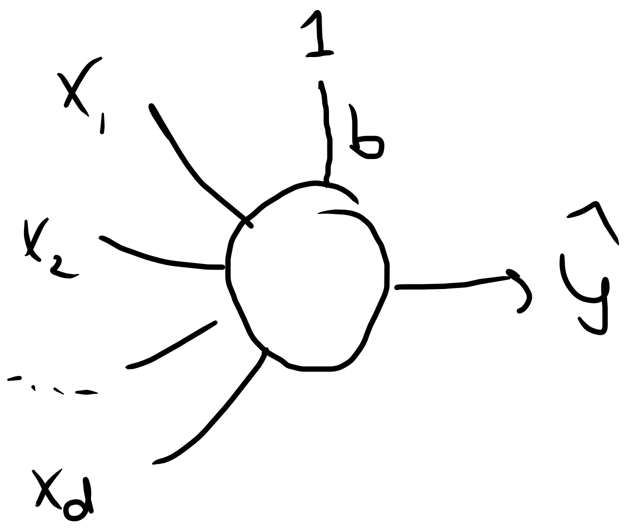
"oud": $\bar{w}_{nieuw} = \bar{w}_{oud} - (\hat{y} - y) \cdot \bar{x}$

"nieuw": $w_{0\ nieuw} = w_{0,\ oud} - (\hat{y} - y) \cdot x_0 \Rightarrow b_{nieuw} = b_{oud} - (\hat{y} - y) \cdot 1$

$w_{1\ nieuw} = w_{1\ oud} - (\hat{y} - y) \cdot x_1$

$\vdots$

$w_{d\ nieuw} = w_{d\ oud} - (\hat{y} - y)\, x_d$

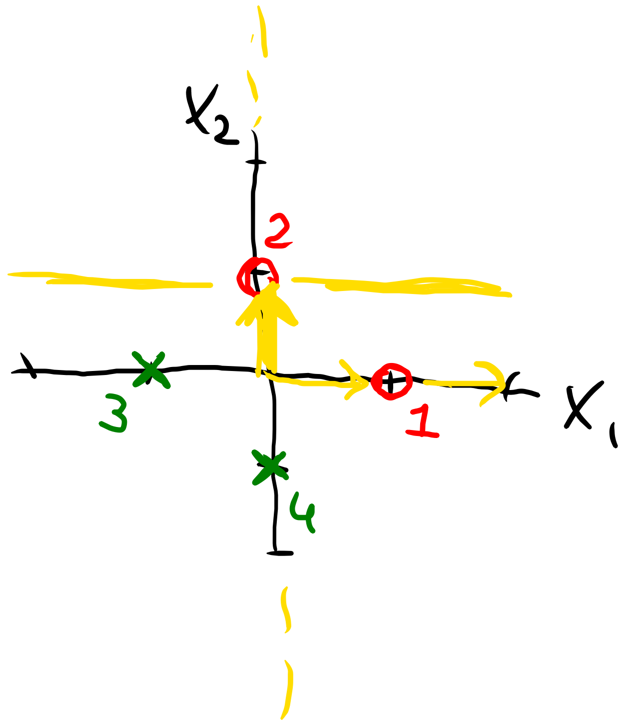


Opgave 14

$$\hat{y} = \mathrm{sgn}(b + w_1 x_1 + w_2 x_2)$$

$$w_i \leftarrow w_i - (\hat{y}-y)x_i$$
$$b \leftarrow b - (\hat{y}-y)$$

	x_1	x_2	y
1	1	0	+1
2	0	1	+1
3	−1	0	−1
4	0	−1	−1



x_1	x_2	y	w_1	w_2	b	$\hat{y}$	$\hat{y}-y$
1	0	+1	0	0	0	0	−1
0	1	+1	1	0	1	+1	0
−1	0	−1	1	0	1	0	+1
0	−1	−1	2	0	0	0	+1
1	0	+1	0	+1	−1	−1	−2

Neuron

$$\hat{y} = \varphi(b + w_1 x_1 + \ldots + w_d x_d)$$

$$\begin{cases} b \leftarrow b - \alpha(\hat{y} - y) \\ w_i \leftarrow w_i - \alpha(\hat{y} - y)x_i \end{cases}$$

φ: activatiefunctie

Perceptron

$$\hat{y} = \text{sgn}(b + w_1 x_1 + \ldots + w_d x_d)$$

$$\begin{cases} b \leftarrow b - (\hat{y} - y) \\ w_i \leftarrow w_i - (\hat{y} - y)x_i \end{cases}$$

φ = signum

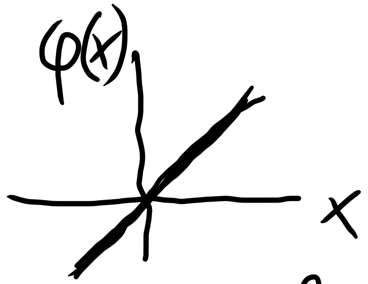
Lineaire regressie

$$\hat{y} = b + w_1 x_1 + \ldots + w_d x_d$$

$$\begin{cases} b \leftarrow b - \alpha(\hat{y} - y) \\ w_i \leftarrow w_i - \alpha(\hat{y} - y)x_i \end{cases}$$

α: learning rate

φ = lambda x: x



identiteitsfunctie

Loss-functie: functie die voor gegeven y en ŷ retourneert "hoe fout" de voorspelling is:
een perfecte voorspelling geeft loss nul, een onjuiste voorspelling geeft positieve loss

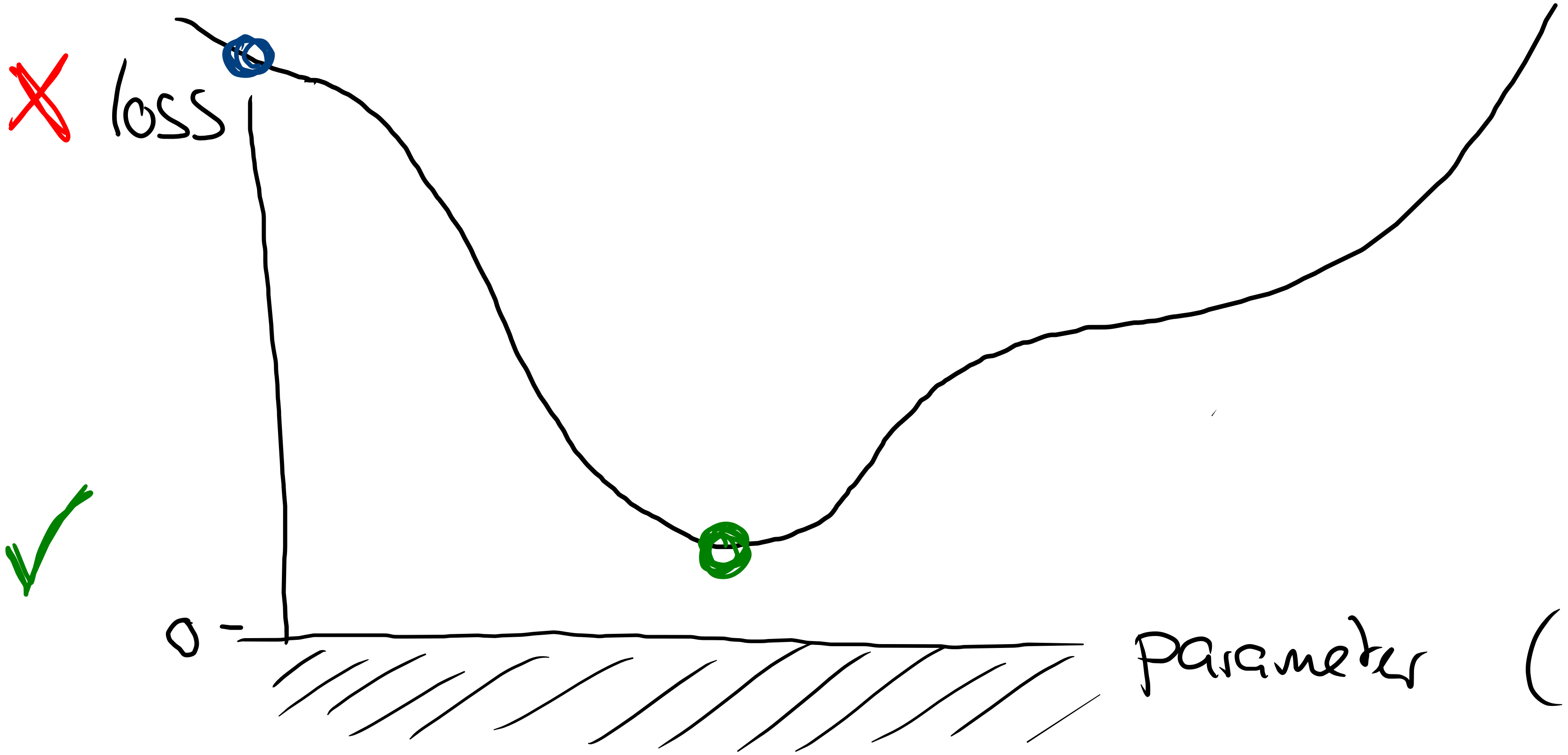
$$\mathcal{L}(\hat{y}\,;\,y)$$

"gradient descent"

bv. voor regressie: $(\hat{y}-y)^2$

$$b \leftarrow b - \alpha \frac{\Delta \text{loss}}{\Delta b}$$

$$w_i \leftarrow w_i - \alpha \frac{\Delta \text{loss}}{\Delta w_i}$$



Het neuron doet achtereenvolgens drie dingen:

1 - lineaire combinatie: pre-activatie $\quad a = b + w_1 \cdot x_1 + \dots + w_d \cdot x_d$

2 - activatiefunctie: post-activatie $\quad \hat{y} = \varphi(a)$

bv: $\varphi(x) = x$ identiteitsfunctie
regressie

$\varphi(x) = \operatorname{sign}(x)$ signum
classificatie

3 - lossfunctie: loss $\quad \ell = \mathcal{L}(\hat{y}; y)$

bv: $\mathcal{L}(\hat{y}; y) = (\hat{y} - y)^2$ kwadratisch

En optimaliseert met gradient descent:

$$w_i \leftarrow w_i - \alpha \cdot \frac{\partial \ell}{\partial w_i} = w_i - \alpha \cdot \frac{\partial \ell}{\cancel{\partial \hat{y}}} \cdot \frac{\cancel{\partial \hat{y}}}{\cancel{\partial a}} \cdot \frac{\cancel{\partial a}}{\partial w_i}$$

"learning rate"

bv. voor regressie: $\varphi(x) = x$

$\mathcal{L}(\hat{y}; y) = (\hat{y} - y)^2$

① $\frac{\partial a}{\partial w_i} = \frac{\partial}{\partial w_i} b + w_i \cdot x_i + \ldots = x_i$

② $\frac{\partial \hat{y}}{\partial a} = \frac{\partial}{\partial a} \varphi(a) = 1$

③ $\frac{\partial \ell}{\partial \hat{y}} = \frac{\partial}{\partial \hat{y}} (\hat{y} - y)^2 = 2(\hat{y} - y)$

$w_i \leftarrow w_i - \alpha \cdot 2(\hat{y} - y) \cdot 1 \cdot x_i = w_i - 2\alpha(\hat{y} - y) x_i$

ditto: $b \leftarrow b - 2\alpha(\hat{y} - y)$

Dit is de update-regel voor regressie (behalve een factor 2)
Dus een neuron met identiteits-activatiefunctie en kwadratische lossfunctie optimaliseren met gradient descent is hetzelfde als lineaire regressie!

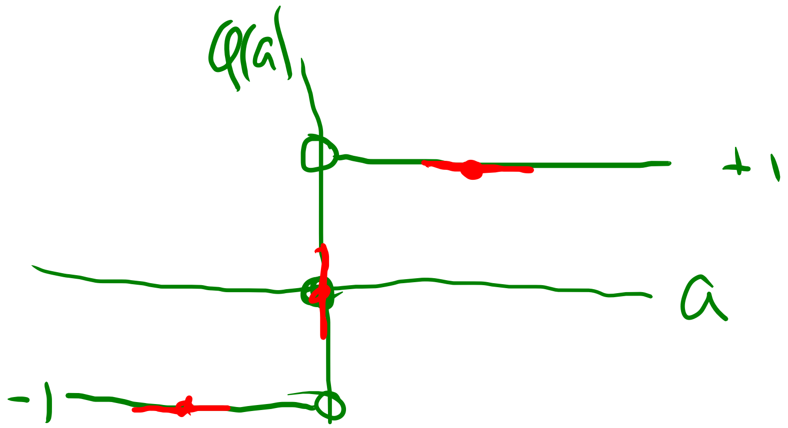
bv: perceptron: $\varphi(x) = \text{sign}(x)$

$$\mathcal{L}(\hat{y}; y) = (\hat{y} - y)^2$$

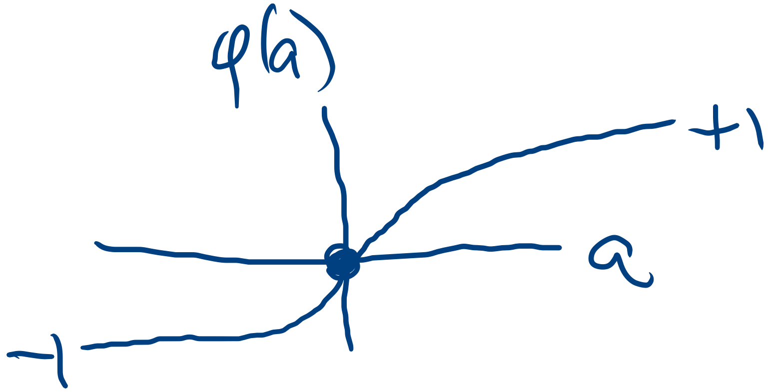
① $\frac{\partial a}{\partial w_i} = x_i$ idem

② $\frac{\partial \hat{y}}{\partial a} =$ nul of oneindig

probleem!



oplossing: afvlakken



$\varphi(a) = \tanh(a)$

$\varphi'(a) = 1 - \tanh^2(a)$

bv. logistische regressie (sort of...)

$$\varphi(a) = \tanh(a)$$

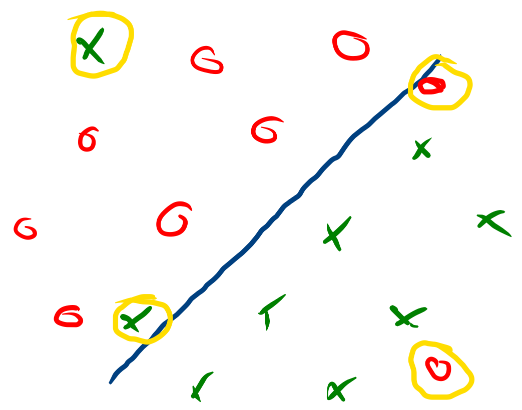
$$\mathcal{L}(\hat{y}, y) = (\hat{y} - y)^2$$

① $\dfrac{\partial a}{\partial w_i} = x_i$

② $\dfrac{\partial \hat{y}}{\partial a} = 1 - \tanh^2(a) = 1 - \hat{y}^2$

③ $\dfrac{\partial \ell}{\partial \hat{y}} = 2(\hat{y} - y)$

$$w_i \leftarrow w_i - 2\alpha(1 - \hat{y}^2)(\hat{y} - y)\, x_i$$



Dit is de updateregel van het perceptron (behalve factor 2α)

Alleen een extra weging met $1 - \hat{y}^2$:

als een fout wordt gemaakt voor een instance dicht op de grenslijn tussen klassen, wordt het model bijgewerkt; als een fout wordt gemaakt voor een instance ver van de lijn, wordt het model niet bijgewerkt ("want dat heeft toch weinig zin")