$$\text{Błąd}(z) = \sigma^2 + \text{Bias}(\hat{f}(z))^2 + \text{Var}(\hat{f}(z))$$

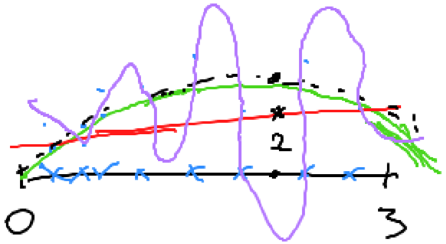
Dane:

$x_i \sim U(0,3)$, $i = 1, 2, \dots, 15$

$y_i = f(x_i) + e_i$, $e_i \sim N(0, 0.25^2)$, $i = 1, \dots, 15$

$f(x) = \sin x$

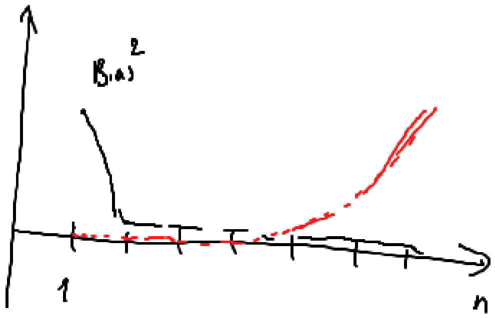
wielomian



Modele:

liniowy $\hat{f}_\theta(x) = \vartheta_0 + \vartheta_1 \cdot x$

wielomianowy $\hat{f}_\vartheta(x) = \vartheta_0 + \vartheta_1 \cdot x + \dots + \vartheta_n \cdot x^n$



Regresja liniowa, gdy zmienne są ze skończonego zbioru

np. $x_1 \in \{kot, pies, papuga\}$

$$\vartheta_0 + \underbrace{\vartheta_1 x_1} + \dots$$

naiwny sposób:

$$\tilde{x}_1 = \begin{cases} 0, & x_1 = kot \\ 1, & x_1 = pies \\ 2, & x_1 = papuga \end{cases} \qquad \begin{matrix} 0 \\ \vartheta_1 \\ 2\vartheta_1 \end{matrix}$$

$$\vartheta_0 + \vartheta_1 \tilde{x}_1 + \dots$$

Lepiej:

$$\tilde{x}_1^{(1)} = \begin{cases} 0, & \text{nie-kot} \\ 1 & \text{kot} \end{cases}$$

$$\tilde{x}_1^{(2)} = \begin{cases} 0 & \text{nie pies} \\ 1 & \text{pies} \end{cases}$$

$$\cancel{\tilde{x}_1^{(3)} = \begin{cases} 0 & \text{nie papuga} \\ 1 & \text{papuga} \end{cases}}$$ zbędne,

kot $\vartheta_1 + c$ $\vartheta_0 + \vartheta_1$

pies $\vartheta_2 + c$ $\vartheta_0 + \vartheta_2$

papuga $\vartheta_3 + c$ ϑ_0

$$\overset{\cdot 1}{(\vartheta_0 - c)} + \underbrace{(\vartheta_1 + c)\tilde{x}_1^{(1)} + (\vartheta_2 + c)\tilde{x}_1^{(2)} + (\vartheta_3 + c)\tilde{x}_1^{(3)}} + \dots$$

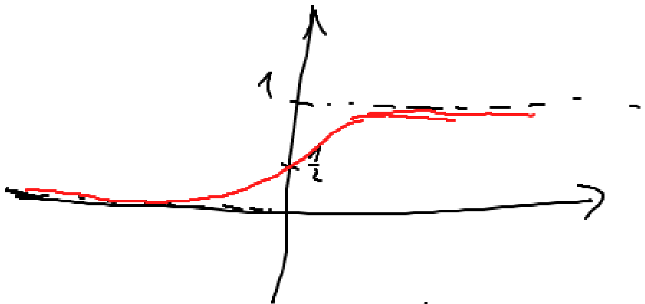
wada: zmienne $(1, \tilde{x}_1^{(1)}, \tilde{x}_1^{(2)}, \tilde{x}_1^{(3)})$ są liniowo zależne

Sieci neuronowe

neuron jest funkcją postaci

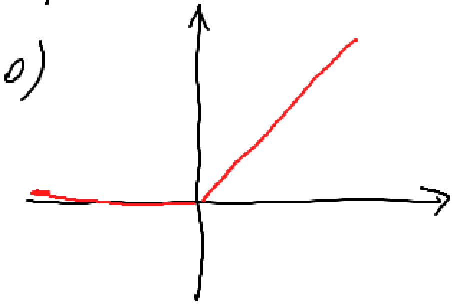
$$\mathbb{R}^n \ni (x_0, x_1, \ldots, x_{n-1}) \longmapsto \sigma(\alpha_0 x_0 + \alpha_1 x_1 + \ldots + \alpha_{n-1} x_{n-1} - \alpha_n) \in \mathbb{R}$$

σ – funkcja aktywacji, nieliniowa np. $\sigma(x) = \frac{1}{1+e^{-x}}$ sigmoid (logistyczna)



$\mathrm{ReLU}(x) = x_+ = \max(x, 0)$

rectified linear unit

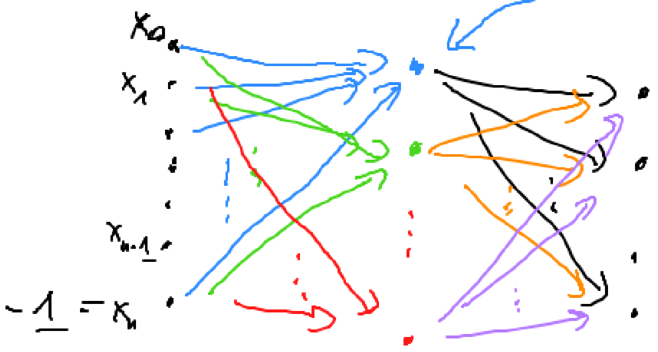


Konwencja: dla wygody połóżmy $x_n := -1$

$$(x_0, x_1, \dots, x_{n-1}) \longrightarrow \sigma\left(\sum_{k=0}^{n} \alpha_k x_k\right)$$

Warstwa (gęsta)

$$\mathbb{R}^n \ni (x_0, x_1, \dots, x_{n-1}) \longrightarrow \left(\sigma\left(\sum_{k=0}^{n} \alpha_{1k} x_k\right), \sigma\left(\sum_{k=0}^{n} \alpha_{2k} x_k\right), \dots, \sigma\left(\sum_{k=0}^{n} \alpha_{mk} x_k\right)\right) \in \mathbb{R}^m$$

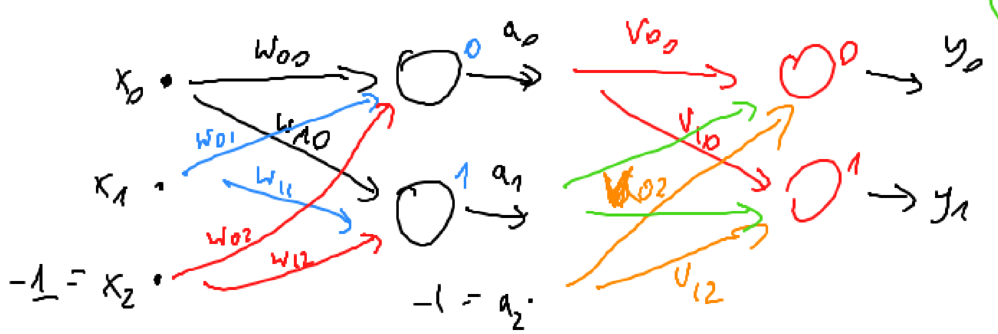


1. warstwa   2. warstwa   ...

Sieć neuronowa jest funkcją powstałą przez złożenie warstw

Nr.

$$\mathbb{R}^2 \ni (x_0, x_1) \longrightarrow (y_0, y_1)$$



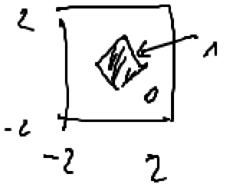
$$y_0 = \sigma\Big(v_{00}\,\sigma(w_{00}x_0 + w_{01}x_1 - w_{02}) + v_{01}\,\sigma(w_{10}x_0 + w_{11}x_1 - w_{12}) - v_{02}\Big)$$

$$\left\{\begin{aligned}
a_0 &= \sigma(w_{00}x_0 + w_{01}x_1 + w_{02}x_2)\\
a_1 &= \sigma(w_{10}x_0 + w_{11}x_1 + w_{12}x_2)\\
y_0 &= \sigma(v_{00}a_0 + v_{01}a_1 + v_{02}a_2)\\
y_1 &= \sigma(v_{10}a_0 + v_{11}a_1 + v_{12}a_2)
\end{aligned}\right.$$

$$\begin{bmatrix} a_0 \\ a_1 \end{bmatrix} = \sigma\Bigg(\overbrace{W \cdot \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix}}^{\begin{bmatrix} w_{00}x_0 + \dots \\ w_{10}x_0 - \dots \end{bmatrix}}\Bigg)$$

$$W = \begin{bmatrix} w_{00} & w_{01} & w_{02} \\ w_{10} & w_{11} & w_{12} \end{bmatrix}$$

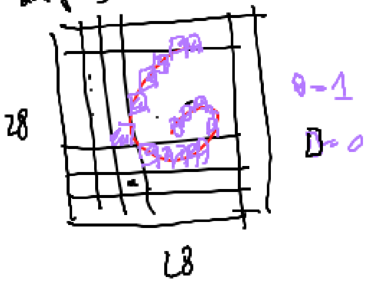
nakładamy σ na poszczególne współrzędne



argmax(y)

Zadanie: MNIST handwritten digits

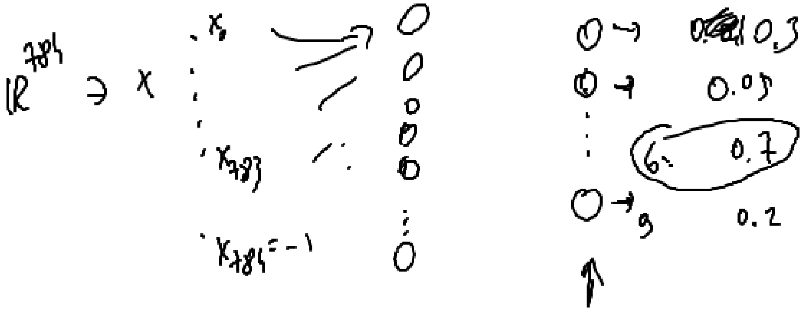
wejście: $x \in \mathbb{R}^{28\times 28} = \mathbb{R}^{784}$



$\to \in \{0, 1, \dots, 9\}$

dane: (x, y) $y \in \{0, \dots, 9\}$ 70 000

cel: zaimplementować sieć neuronową, która modeluje funkcję jw.



argmax daje predykcję wartości

warstwa

1. warstwa

10 neuronów

być może jeszcze jakieś warstwy (niekoniecznie)

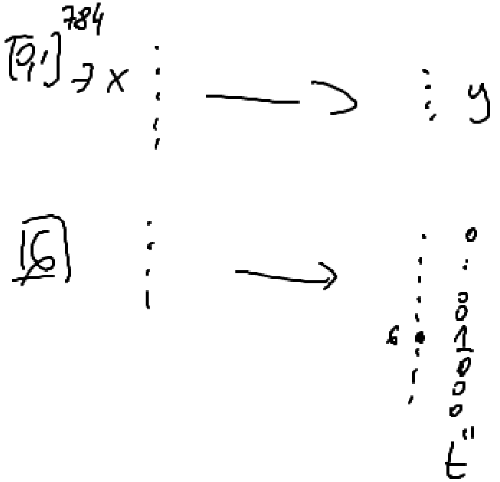
Metoda iteracyjna dopasowywania wag (backward propagation, propaguje wstecz)

Wprowadzamy funkcję kosztu:

$$L(t,y) = \frac{1}{2}\|t-y\|_2^2 = \frac{1}{2}\sum_{\cancel{j}k=0}^{9}(t_k - y_k)^2$$

t – „wartość prawdziwa"

y – modelowana

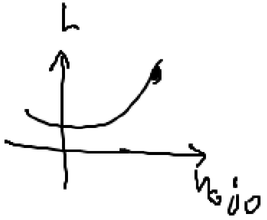


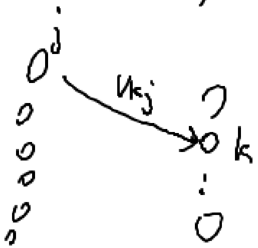
$$y_k = \sigma\left(\sum_j v_{kj}\, a_j\right)$$

$v = v_{k_0 j_0}$

$$\frac{\partial L}{\partial v_{k_0 j_0}} = \frac{\partial L}{\partial v} = \frac{\partial L}{\partial y_{k_0}}(t,y) \cdot \frac{\partial y_{k_0}}{\partial v} = \underbrace{\frac{\partial L}{\partial y_{k_0}}(t,y)}_{(t_{k_0} - y_{k_0})} \cdot \sigma'\left(\sum_j v_{k_0 j}\, a_j\right) \cdot a_{j_0}$$

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





$$\text{nowe } v_{k_0, j_0} = v_{k_0 j_0} - c \cdot \frac{\partial L}{\partial v_{k_0 j_0}}$$

$c = 0.01$

Jak sprawdzić dopasowanie modelu?

dzielimy dane na 2-3 części:

$\boxed{\theta}$

- dane treningowe 50k
- dane testowe 10k

(• dane weryfikacyjne) 10k