



Strojové učenie a neurónové siete - **zadanie 1** - analýza dát, jednoduché neurónové siete 2

Zadanie 1

- Zadanie a bodovanie je k dispozícii na dokumentovom serveri v AIS
- Máte sa naučiť:
 1. Načítanie dát
 2. Základná analýza dát
 3. Čistenie dát, spracovanie textových stĺpcov, vyberanie množín
 4. Príprava dát na vstup do siete (škálovanie)
 5. Použitie neurónovej siete na regresiu/klasifikáciu
 6. Základné vyhodnocovanie úspešnosti sietí
- Dataset: dostupný v AIS aj s popisom stĺpcov (prescreening pacientov na srdcovú chorobu)

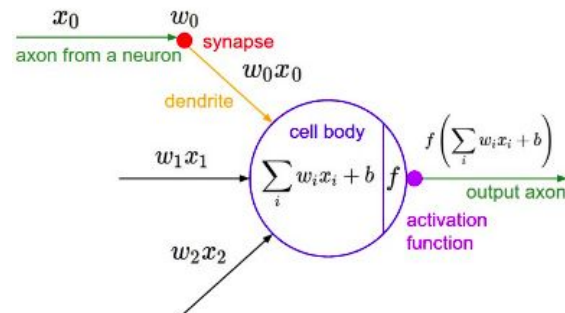
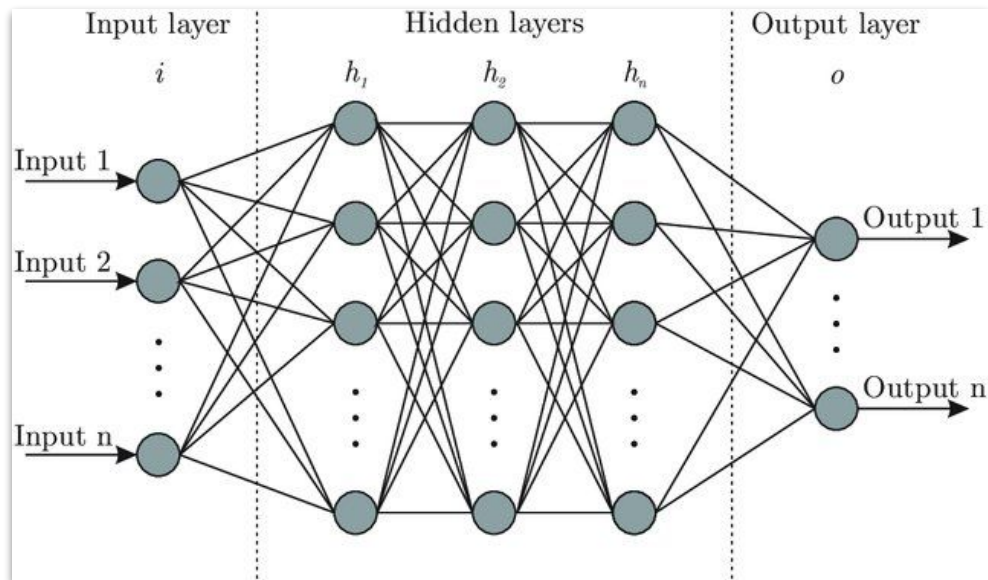


Obsah tejto prezentácie

- Kriteiálne funkcie
- Príklad spätnej propagácie
- Architektúra umelej neurónovej siete
- Proces trénovania
- “Sanity checks”
- Postup riešenia demonštrovaný na datasete zo zdroja, využité knižnice numpy, pandas, plotly, keras



Umelé neurónové siete - slovníček



- Neurón:
 - Váhy
 - Prah
 - Aktivačná funkcia
- Neurónová sieť:
 - Vstupná vrstva
 - Výstupná vrstva
 - Skryté vrstvy
- Trénovanie:
 - Kriteiálna funkcia
 - Spätne šírenie chyby
 - Solver
 - Parameter rýchlosti učenia
 - Zastavovacia podmienka

Spätná propagácia - opakovanie parciálne derivácie

- Pri sieťach máme funkciu $f(\mathbf{x})$, \mathbf{x} sú premenné (vstupy, váhy a prahy)

- Násobenie:

$$f(x, y) = xy \rightarrow \frac{\partial f}{\partial x} = y \qquad \frac{\partial f}{\partial y} = x$$

$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right] = [y, x]$$
$$\frac{df(x)}{dx} = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h}$$

- Sčítanie:

$$f(x, y) = x + y \rightarrow \frac{\partial f}{\partial x} = 1 \qquad \frac{\partial f}{\partial y} = 1$$

- ReLU :):

$$f(x, y) = \max(x, y) \rightarrow \frac{\partial f}{\partial x} = 1(x \geq y) \qquad \frac{\partial f}{\partial y} = 1(y \geq x)$$



Spätná propagácia - príklad 1.

Reťazové pravidlo: $\frac{\partial f}{\partial x} = \frac{\partial f}{\partial q} \frac{\partial q}{\partial x}$

Napr. zoberme funkciu:

$$f(x, y, z) = (x + y)z$$

Dá sa prepísať:

$$q = x + y$$

$$f = qz$$

Výpočty gradientov:

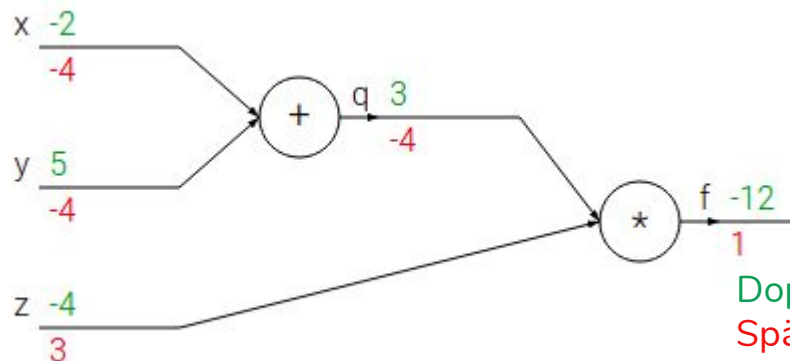
$$\frac{\partial f}{\partial z} = \frac{\partial qz}{\partial z} = q$$

$$\frac{\partial f}{\partial q} = \frac{\partial qz}{\partial q} = z$$

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial q} \frac{\partial q}{\partial x} = z * 1$$

$$\frac{\partial f}{\partial y} = \frac{\partial f}{\partial q} \frac{\partial q}{\partial y} = 1 * z$$

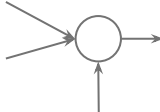
Pre konkrétne hodnoty:

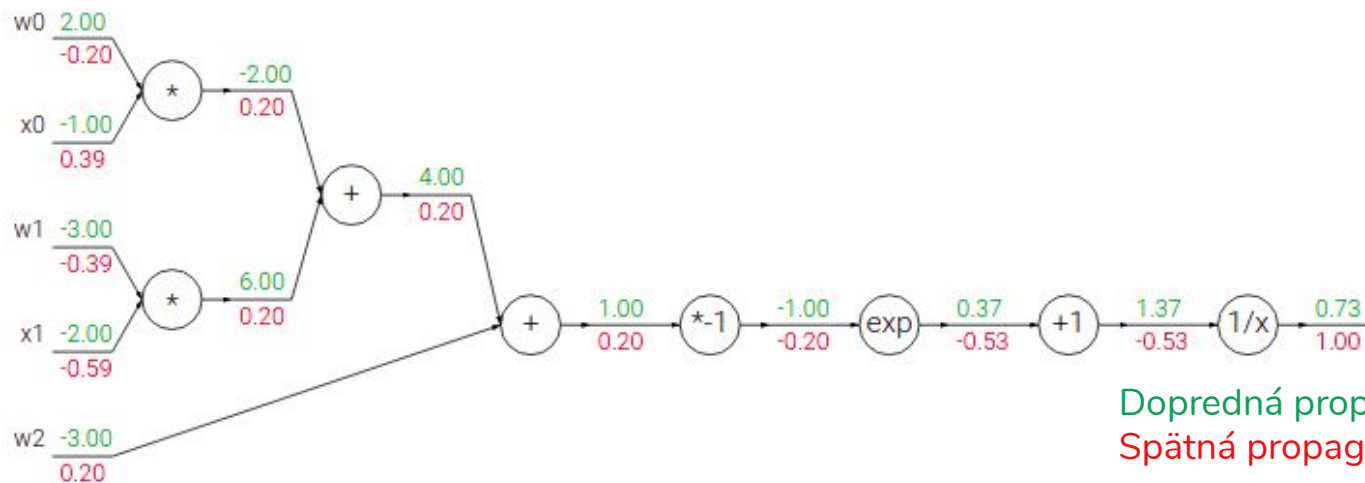
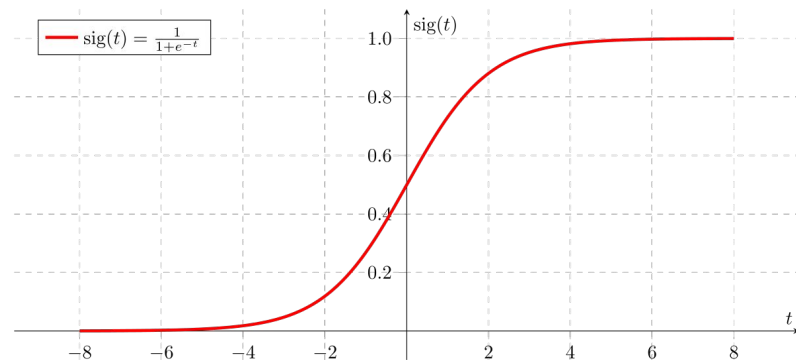


Dopredná propagácia
Spätná propagácia

Spätná propagácia - príklad neurón

Neurón s dvoma vstupmi a sigmoidou ako kritériálnou funkciou:

$$f(w, x) = \frac{1}{1 + e^{-(w_0x_0 + w_1x_1 + w_2)}}$$




Dopredná propagácia
Spätná propagácia

Ťahák:

$$f(x) = \frac{1}{x} \rightarrow \frac{df}{dx} = -1/x^2$$

$$f_c(x) = c + x \rightarrow \frac{df}{dx} = 1$$

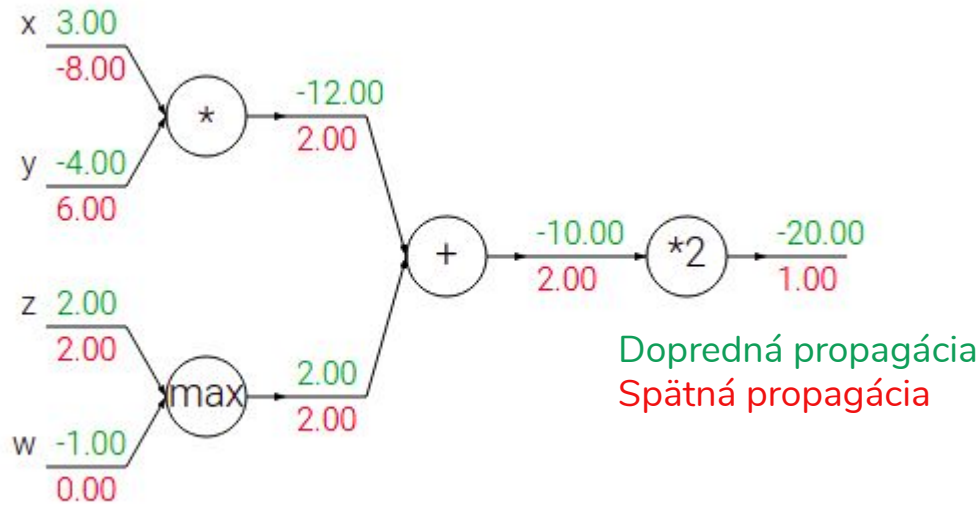
$$f(x) = e^x \rightarrow \frac{df}{dx} = e^x$$

$$f_a(x) = ax \rightarrow \frac{df}{dx} = a$$

$$\sigma(x) = \frac{1}{1 + e^{-x}} \rightarrow \frac{d\sigma(x)}{dx} = \frac{e^{-x}}{(1 + e^{-x})^2} = \left(\frac{1 + e^{-x} - 1}{1 + e^{-x}} \right) \left(\frac{1}{1 + e^{-x}} \right) = (1 - \sigma(x)) \sigma(x)$$



Spätná propagácia - príklad 3.



Diskusia:

- Sumovacia brána, max brána, násobiaca brána
- Normalizácia
- Rozmery vstupu, dot product
- Prečo trénujeme iteratívne?

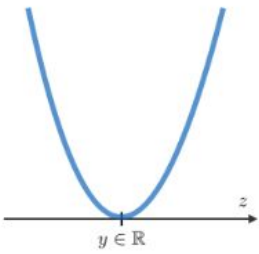
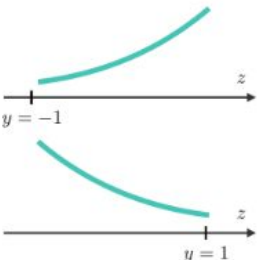
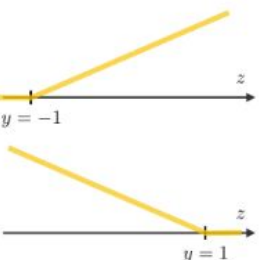
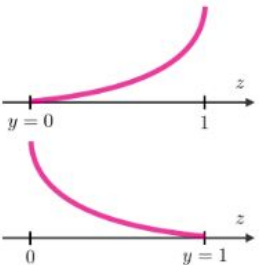


Kritériálna funkcia

Chybová funkcia, loss function, error function, cost function, objective function

Trénovanie =[dopredná propagácia → **výpočet kritériálnej funkcie** → spätná propagácia → (dopredná propagácia)] x počet iterácií

- Hodnotí ako dobre sieť ohodnocuje vstupy vzhľadom na skutočné výstupy, najčastejšie:

Least squared error	Logistic loss	Hinge loss	Cross-entropy
$\frac{1}{2}(y - z)^2$	$\log(1 + \exp(-yz))$	$\max(0, 1 - yz)$	$-\left[y \log(z) + (1 - y) \log(1 - z)\right]$
			
Linear regression	Logistic regression	SVM	Neural Network

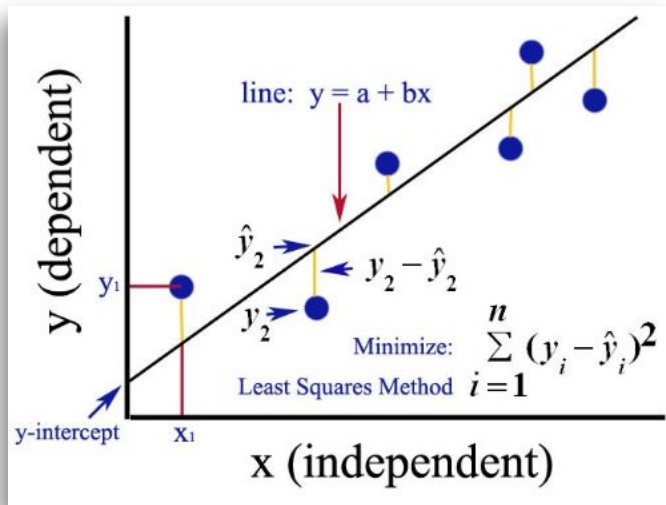
y - skutočný výstup
z - predikovaný výstup



Kritériálna funkcia - príklady

Ordinary least squares (OLS)

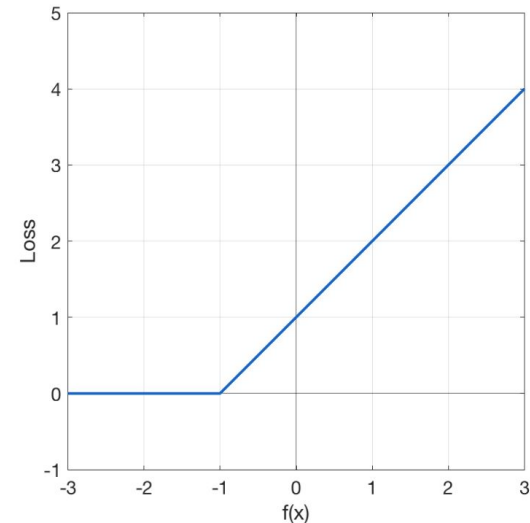
- Lineárna regresia
- sklearn.neural_network.MLPRegressor



Hinge loss

$$\max(0, 1 - y(f(x)))$$

- Binárna klasifikácia
- Primárne so SVM
- Chceme mať výstup pre triedy rozdelený "marginom"
- y je buď -1 alebo 1

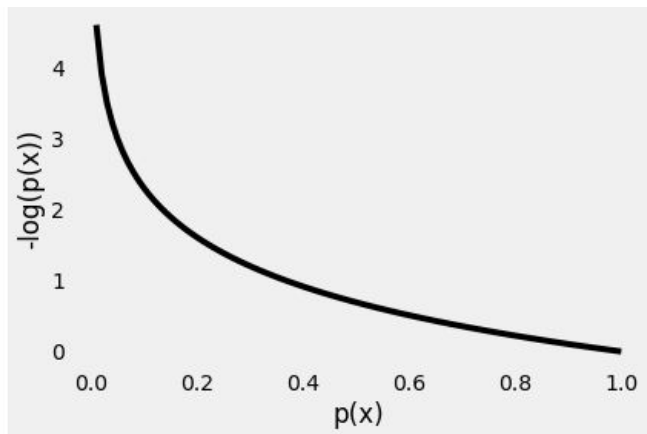


Kritériálna funkcia - príklady

Binary Cross Entropy

$$y * \log(f(x)) + (1 - y) * \log(1 - f(x))$$

- Binárna klasifikácia
- y je 0 alebo 1
- Normalizované pravdepodobnosti príslušnosti ku triede



Cross Entropy (log loss, softmax)

$$-\log \frac{e^{f_{y_i}}}{\sum_j e^{f_j}}$$

-Rozšírenie BCE pre viacero tried

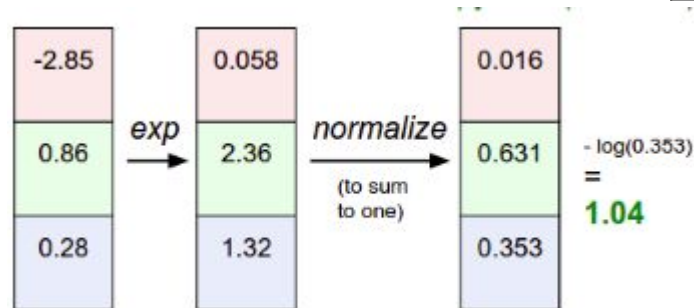
-Softmax - normalizácia do sumy 1

$$\frac{e^{z_j}}{\sum_k e^{z_k}}$$

-One-hot encoding

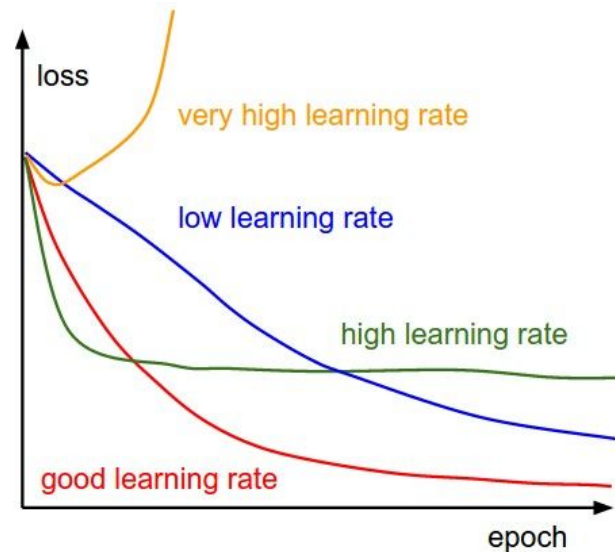
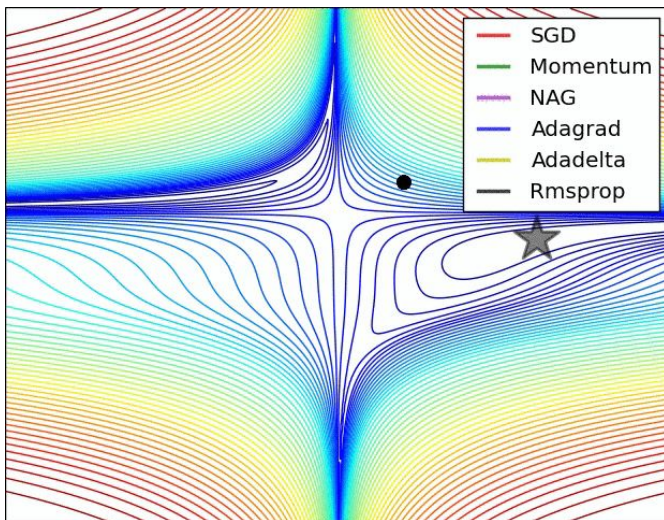
-sklearn.neural_network.MLPClassifier

y_i	2
	0
	0
	1



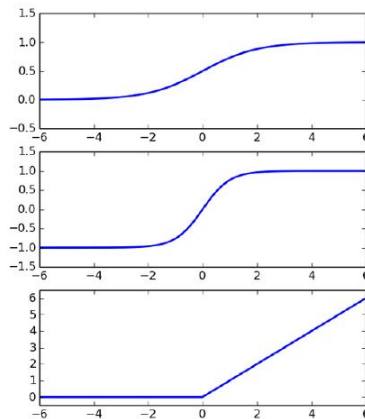
Trénovanie neurónovej siete

- Sieť trénujem postupne v iteráciách (gradientný zostup)
- O tom, ako veľmi mením váhy rozhoduje parameter rýchlosti učenia (medzi 0 a 1) - dôležitý pre správne tréovanie
- O spôsobe zostupu rozhoduje solver (SGD, Adam, Adagrad...)



Architektúra neurónovej siete

- Veľkosti vrstiev (počty neurónov vo vrstvách):
 - Vstupná - podľa veľkosti vzorky zo vstupu
 - Výstupná - jeden (regresia, binárny problém) alebo počtom kategórii (multiclass)
 - Skryté - nevieme :(, vyberáme my (a potom sledujeme správanie siete počas tréningu); možno:
 - Čím hlbšie tým lepšie (min. pri CNN) - ale je ťažké dobre natréňovať
 - Počet neurónov medzi veľkosťou vstupu a výstupu?
 - Dobré je mať 10 stupňov voľnosti medzi počtom vstupných vzoriek a tréňovanými parametrami (mám 300 vzoriek môžem tréňovať 30 váh)
- Aktivačná funkcia:
 - Sigmoid, logaritmická
 - Miznúci gradient, nie je centrovaná na nule, pomalá konvergencia
 - TanH
 - Miznúci gradient
 - ReLU
 - Mŕtve neuróny



Sigmoid

$$\phi(z) = \frac{1}{1 + e^{-z}}$$

Hyperbolic Tangent

$$\phi(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$

Rectified Linear

$$\phi(z) = \begin{cases} 0 & \text{if } z < 0 \\ z & \text{if } z \geq 0 \end{cases}$$



Nastavenia tréovania

- Zastavovacie podmienky:
 - Počet iterácií
 - Hodnota kritériálnej funkcie (úspešnosti) - tréovanie, validácia
 - Smer zmeny kritériálnej funkcie (úspešnosti) za posledných x iterácií
- Nevyhnutné je nájsť dobrý parameter rýchlosti učenia (prvá kontrola, keď sa sieť netrénuje)
- Často skúmam solver a jeho nastavenia, architektúru
- Menej často skúmam aj napr. inicializáciu váh
- Podľa priebehu tréovania - vyhýbam sa pretrénovaniu/podtrénovaniu - regularizačné techniky (nabudúce)



Prečo sa moja sieť netrénuje - základné kontroly

- V prvom kroku je hodnota kritériálnej funkcie približne rovnaká, ako by mala byť pri náhodnom rozhodovaní
 - Napr. pri 10 kategóriách: $\text{softmax} \sim -\ln(1/10) = 2.3$
- Zvýšenie regularizácie zvyšuje hodnotu kritériálnej funkcie.
 - ...
- Na malej podmnožine trénovacej množiny viem sieť pretrénovať
 - Viem dostať hodnotu kritériálnej funkcie na nulu.





Priestor na otázky