Prečo sa stroje učia

podľa
Anila Ananthaswamyha, 2024

Obsah

- Hľadáme vzory, vzory v dátach
- Prvý umelý neurón, McCulloch a Pitts (1943)
- Hebbov mechanizmus učenia
- Učenie sa z chýb
- Práca s vektormi a maticami
- Skalárny súčin
- Perceptrón, Rosenblatt, Mark I (1957)
- Trénovací algoritmus perceptrónu,
- Konvergencia váh perceptrónu k riešeniu
- Hopfieldove siete
- Vrstvové siete

Vzory v dátach

x_1	x_2	y
4	2	10
1	2	7
0	5	15
2	1	5

Vzor pre tieto dáta:

$$x_1 + 3x_2 = y$$

Všeobecnejší zápis:

$$w_1x_1 + w_2x_2 = y,$$

kde $w_1 = 1$ a $w_2 = 3$

- Lineárny vzťah medzi y a $x_1,\ x_2$
- Konštanty w_1 a w_2 - koeficienty alebo váhy
- Ako nájsť (naučiť sa) váhy?
- K čomu poslúžia váhy?
- Dáta vyjadrujú určitú koreláciu medzi vstupom a výstupom - trénovacie dáta
- Učenie s učiteľom (supervised learning)

Zovšeobecnenie: $y = w_1x_1 + \cdots + w_nx_n = \sum_{i=1}^{n} w_ix_i$

Prvý umelý neurón

Warren McCulloch a Walter Pitts, logické funkcie

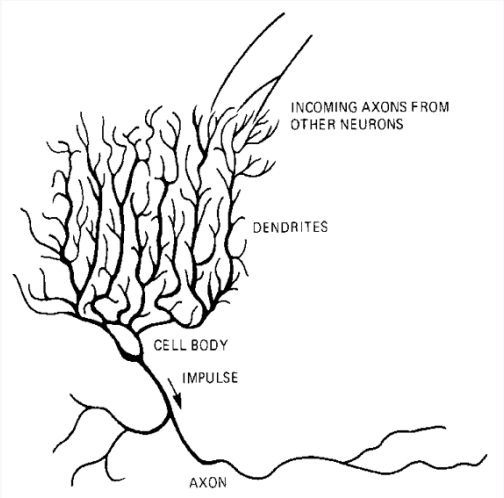


Figure: Biologický neurón

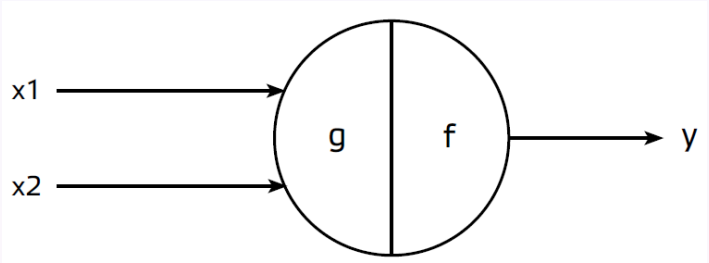


Figure: Umelý neurón

$x_1, x_2 \in \{0, 1\}$, θ -prahová hodnota

Výpočet: $suma = x_1 + x_2$; Ak $suma \geq \theta : y = 1$, inak $y = 0$

Pracujeme s umelým neurónom

Úlohy: Ako nastaviť θ?

- Logická funkcia AND

x_1	x_2	y
0	0	0
1	0	0
0	1	0
1	1	1

- Logická funkcia OR

x_1	x_2	y
0	0	0
1	0	1
0	1	1
1	1	1

- Logická funkcia XOR

x_1	x_2	y
0	0	0
1	0	1
0	1	1
1	1	0

AND: Ľahko overíme, že $\theta = 1.5$ vyhovuje
OR: $\theta = ?$
XOR: $\theta = ?$

Zovšeobecnenie pre n premenných

$$g(x) = x_1 + x_2 + \cdots + x_n = \sum_{i=1}^{n} x_i$$

$$f(z) = \begin{cases} 0, & z < \theta \\ 1, & z \geq \theta \end{cases}$$

$$y = f(g(x)) = \begin{cases} 0, & g(x) < \theta \\ 1, & g(x) \geq \theta \end{cases}$$

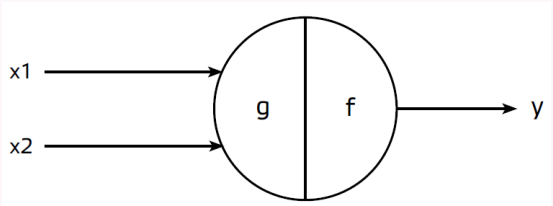


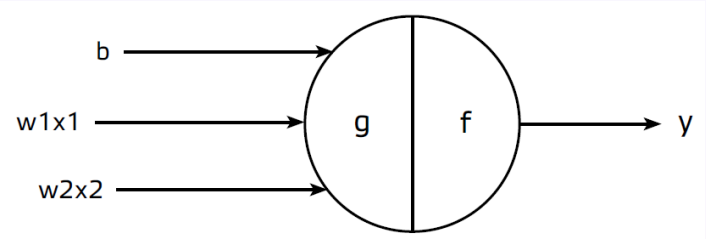
Figure: Stroj, ktorý dokáže počítať, ale nie učiť sa.

Učenie sa z chýb

V súvislosti s biologickými neurónmi navrhol Hebb mechanismus učenia, ktorý sa často stručne, ale trochu mylne popisuje takto: "Neuróny, ktoré sa súčasne aktivujú, sa zároveň aj prepájajú." (Neurons that fire together wire together.)

Hebbovské učenie: Naše mozgy sa učia, pretože spojenia medzi neurónmi sa posilňujú, keď sa výstup jedného neurónu stabilne podieľa na aktivácii iného, a oslabujú, keď to tak nie je.

Rosenblattov perceptrón



Výpočet: b je skreslenie (bias);
$x_1, x_2, w_1, w_2, b \in \mathbf{R}$
$suma = w_1 x_1 + w_2 x_2 + b$
Ak $suma > 0 : y = 1$ inak $y = -1$

Zovšeobecnenie:

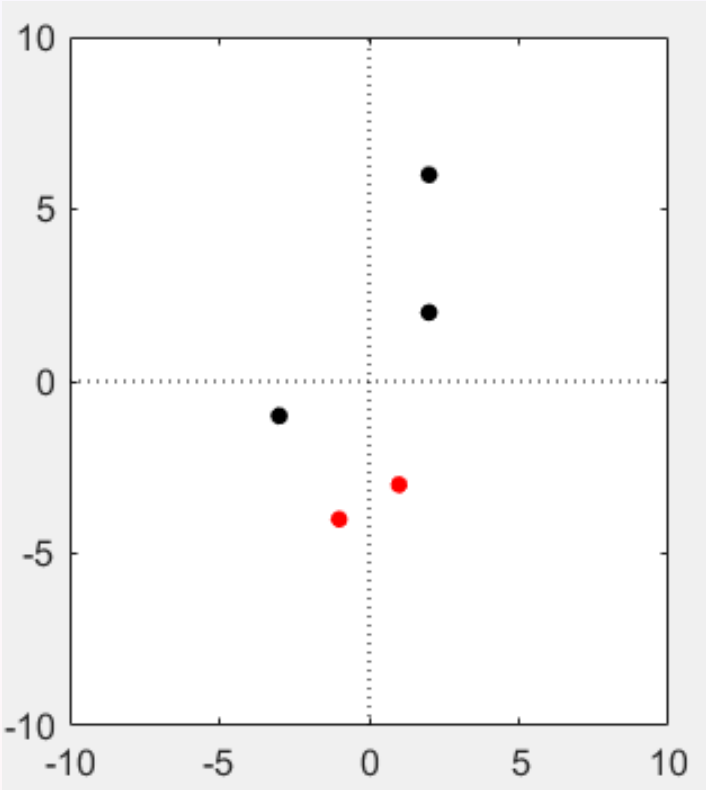
$$g(x) = w_1 x_1 + w_2 x_2 + \ldots w_n x_n + b = \sum_{i=1}^{n} w_i x_i + b$$

$$f(z) = \begin{cases} -1, & z \leq \theta \\ 1, & z > \theta \end{cases}$$

$$y = f(g(x)) = \begin{cases} -1, & g(x) \leq \theta \\ 1, & g(x) > \theta \end{cases}$$

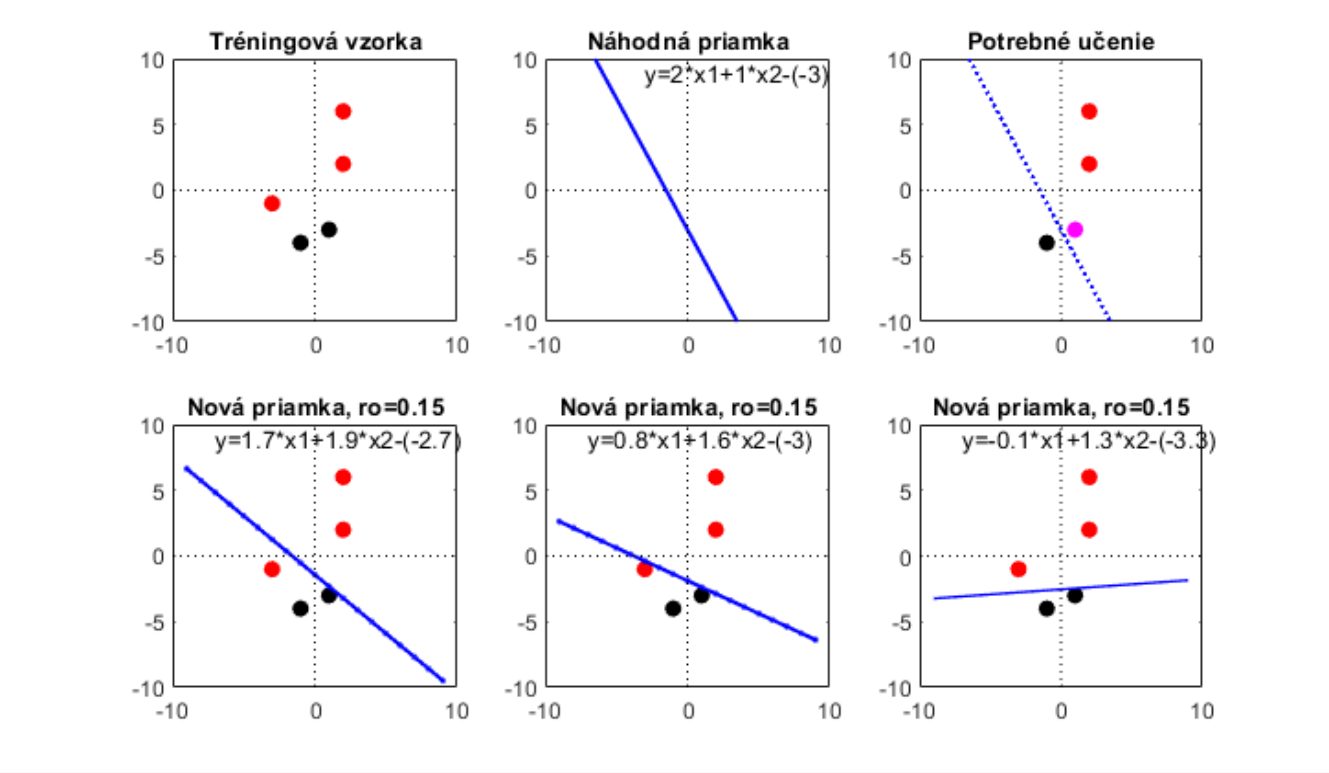
2 triedy v dátach oddelené nadrovinou.

Rosenblattov perceptrón



- Úlohou perceptrónu je "naučiť" sa hodnoty w_1, w_2 a hodnotu skreslenia b tak, aby správne klasifikoval každý bod v súbore dát správne.
- Keď sa perceptrón naučí správne hodnoty váh w_1 a w_2 a skreslenia b, je pripravený dávať **predikcie** o ďalších bodoch.

Príklad učenia



- V roku 1959 Rosenblatt zostrojil svoj perceptrón Mark I z množstva takých jednotiek.
- Stroj dokázal spracovať obrázky s rozmermi 20 x 20 pixelov, pričom každému pixelu zodpovedala vstupná hodnota x.
- Perceptrón sa učí korelácie, bez toho aby mu to bolo explicitne povedané, o aké korelácie sa jedná.
- **Je rozpoznávanie korelácií to isté, ako myslenie a uvažovanie?**
- Zostrojenie perceptrónu bolo veľkým úspechom.
- Ešte väčším úspechom bol matematický dôkaz, že jedna vrstva perceptrónov vždy nájde lineárne oddeľujúcu nadrovinu, ak sú dáta lineárne oddeliteľné.

Vektory ako čísla

- Vektory v geometrii
- Číselné vyjadrenie vektorov
- Jednotkové vektory: $\mathbf{i} = (1, 0)$; $\mathbf{j} = (0, 1)$
- Zápis vektora pomocou jednotkových vektorov
- Operácie na vektoroch - vynásobenie skalárom, dĺžka vektora
- Skalárny súčin dvoch vektorov $\mathbf{a} = (a_1, a_2)$ a $\mathbf{b} = (b_1, b_2)$, označenie $\mathbf{a} \cdot \mathbf{b}$ - veľkosť vektora $\mathbf{a}$ nasobená priemetom vektora $\mathbf{b}$ do vektora $\mathbf{a}$

$$\mathbf{a} \cdot \mathbf{b} = (a_1 b_1, a_2 b_2)$$

- Ďalšie operácie na vektoroch - sčítanie, rozdiel, vektorový súčin
- **Úloha**: Nech sú dané 2 vektorry: $\mathbf{a} = (2, 5)$ a $\mathbf{b} = (3, -2)$. Urobte na nich tieto operácie. Ukážte aj geometricky.

Stroje a vektory

Geometrická interpretácia toho, čo sa deje pri učení.

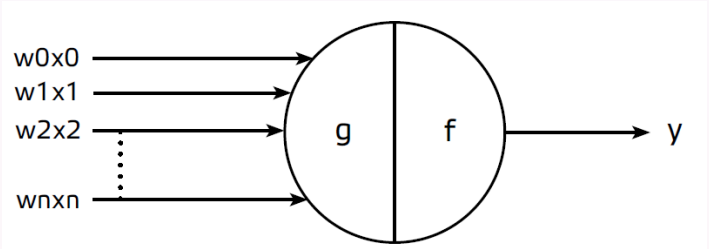
- Perceptrón počíta:

$$y = \begin{cases} -1, & \text{ak } w_1 x_1 + w_2 x_2 + b \leq \theta \\ 1, & \text{ak } w_1 x_1 + w_2 x_2 + b > \theta \end{cases}$$

- Váhy tvoria vektor $\mathbf{w} = (w_1, w_2)$, nech $\mathbf{u}$ je jednotkový vektor v rovnakom smere.
- Uvažujme nadrovinu určenú rovnicou $w_1 x_1 + w_2 x_2 = 0$. Prechádza počiatkom súradnicového systému.
- Čo vieme povedať o nadrovine $w_1 x_1 + w_2 x_2 + b - \theta = 0$?
- Vektor $\mathbf{w}$ je kolmý na obe nadroviny.
- Naučenie váh znamená nájdenie nadroviny, ktorá rozdelí rovinu na kladnú a zápornú polrovinu.

Ďalšie úvahy k učeniu

- Vždy sa dá nájsť taká nadrovina? Pre každé dáta?
- Perceptrón sa učí váhy a hodnotu skreslenia, aby predikoval, kam sa vzhľadom k nadrovine nejaký doposiaľ nepoznaný datový bod dostane.
- **Zovšeobecnenie:** Všeobecný perceptrón so vstupným vektorom $\mathbf{x} = (x_0, x_1, x_2, \ldots, x_n), x_0 = 1$ a váhovým vektorom $\mathbf{w} = [w_0, w_1, w_2, \ldots, w_n]$ vypadá následovne:



- A počíta

$$y = \begin{cases} -1 & \text{ak } \mathbf{w}^T\mathbf{x} \leq 0 \\ 1 & \text{ak } \mathbf{w}^T\mathbf{x} > 0 \end{cases}$$

Dôležité vedieť

- Váhový vektor $\mathbf{w}$ je kolmý na priamku alebo nadrovinu, ktorá rozdeľuje dátové body do dvoch tried;
- Keď sa perceptrón naučí váhový vektor, potom pri zadávaní nového dátového bodu, ktorý má byť klasifikovaný, je potrebné vypočítať

$$\mathbf{w}^{\mathbf{T}}\mathbf{x}$$

pre novú inštanciu dát $\mathbf{x}$ a zistiť, či patrí na jednu alebo druhú stranu nadroviny, a podľa toho ju klasifikovať.

Algoritmus učenia

1. Inicializovať váhový vektor na nulový: nastav $\mathbf{w} = \mathbf{0}$
2. Pre každý dátový bod x v trénovacej množine dát urobiť nasledujúce:

   ak $y\mathbf{w}^{\mathbf{T}}\mathbf{x} \leq 0$, kde $y \in \{1, -1\}$:
   – váhový vektor je chybný, preto ho aktualizuj:

$$\mathbf{w_{novy}} = \mathbf{w_{stary}} + y\mathbf{x}$$

3. Ak v kroku nedošlo k aktualizácii váhového vektora, algoritmus končí, inak prechod na krok 2;

Matematická otázka: Ako si môžeme byť istí, že tento proces skončí?
Prečo nebude pokračovať donekonečna?
Dôkaz existuje a je v knihe.

Úloha:

m	x^m	d^m
1	(2,6)	1
2	(-1,-4)	-1
3	(2,2)	1
4	(1,-3)	-1
5	(-3,-1)	1

Nájsť perceptrón, ktorý bude klasifikovať dáta v tabuľke do dvoch predpísaných tried.

Sledovať zmenu váh.
Ukázať aj geometricky.

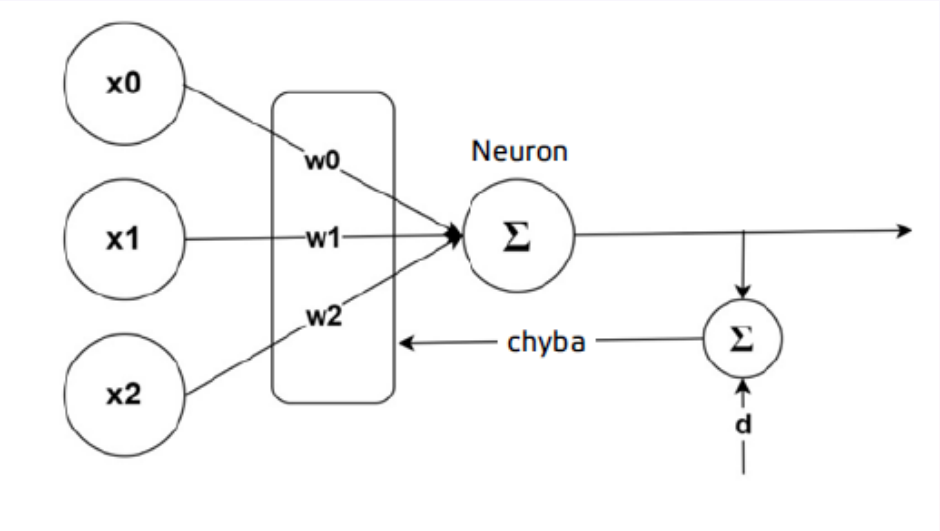
$$y = \begin{cases} -1 & \text{ak } \mathbf{w}^T\mathbf{x} \leq 0 \\ 1 & \text{ak } \mathbf{w}^T\mathbf{x} > \end{cases}$$

Úloha:

$\mathbf{w} = (0, 0, 0)$

- 1. príklad – $(1, 2, 6)$, dáva $y = -1$. Prečo?
  Klasifikácia je chybná.
  Nové váhy: $\mathbf{w} = (-1, -2, -6)$. Prečo?
- 2. príklad – $(1, -1, -4)$, dáva $y = 1$.

Widrow-Hoffov adaptívny neurón



Neurón počíta

$$y = \mathbf{w}^T \mathbf{x}$$

Chyba

$$chyba(e) = d - y = d - \mathbf{w}^T \mathbf{x}$$

kde d je očakávaná hodnota.
Úprava váh

$$\mathbf{w}_{nove} = \mathbf{w}_{stare} + \mu(-\Delta)$$

kde μ je veľkosť kroku a Δ je gradient

$$(\frac{\partial J}{\partial w_0}, \frac{\partial J}{\partial w_1}, \frac{\partial J}{\partial w_2})$$

Widrow - Hoffovo učenie, algoritmus najmenších štvorcov

$$\mathbf{w}_{nove} = \mathbf{w}_{stare} + \mu(-\Delta_{odhad})$$

Tento odhad vychádzal z jediného dátového bodu. Nezahŕňal výpočet očakávanej hodnoty kvadratickej chyby.

$$\mathbf{w}_{nove} = \mathbf{w}_{stare} + 2\mu\varepsilon\mathbf{x}$$

kde

- μ je veľkosť kroku,
- $\varepsilon = d - \mathbf{w}^T\mathbf{x}$ je chyba jedného dátového bodu,
- $\mathbf{x}$ jeden dátový bod

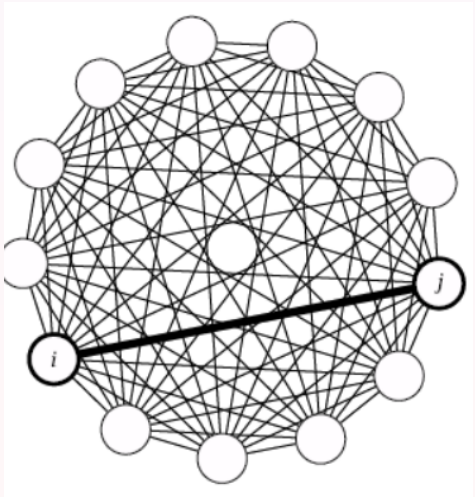
W-H overili, že algoritmus funguje tak, že zostrojili jeden adaptivný neurón – skutočný hardwarový neurón –**ADALINE**

Nobelova cena za fyziku, 2024

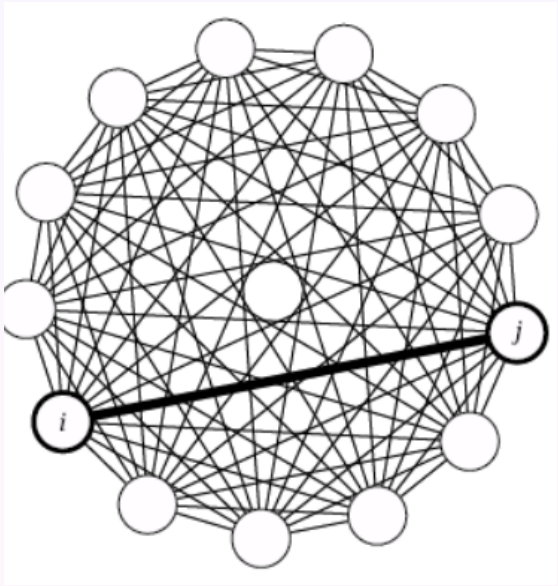
Za základné princípy, ktoré umožňujú strojové učenie pomocou umelých neurónových sietí

Hopfieldove siete

- Hopfield pátral po základnom a potenciálne ďalekosiahlom poznatku o tom, ako v mozgu prebieha počítanie.
- Asociatívna pamäť
- Mohla by sieť umelých neurónov, ktoré by v sebe mali uložené nejaké spomienky, vyvolať určitú spomienku, keby z nej dostala len určitý malý kúsok?
- Hopfield je teoretický fyzik, Isingov model spinov
- Hopfieldove neurónové siete

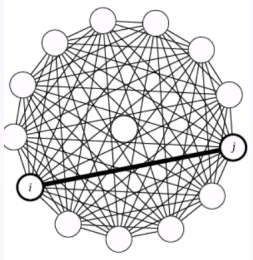


Hopfieldove neurónové siete (HS)



- Úplný graf. Každý neurón je vstupný aj výstupný.
- **Asociatívna pamäť** - sieť, ktorá používa metódu na ukladanie a opätovné vytváranie vzorov.
- Neuróny sa môžu nachádzať v jednom z dvoch stavov. Aktivačná funkcia: $f(x) = sign(x)$.
- HS využíva fyziku, ktorá popisuje vlastnosti materiálu pomocou atómového spinu – vlastnosť elementárnych častíc.

Hopfieldove neurónové siete



- **Energia** asociovaná so sieťou:

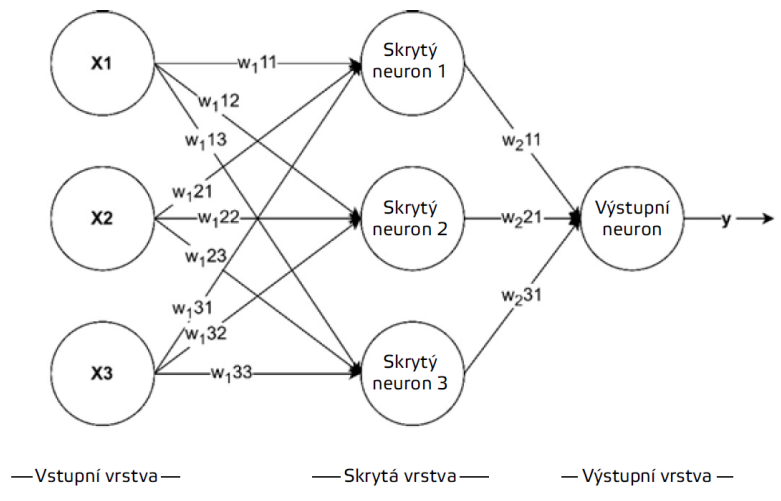
$$E = -\frac{1}{2}\sum_{j}\sum_{i=1}^{n} w_{ji}s_i s_j$$

- Po zadaní vstupu sa sieť snaží dostať do stavu s minimálnou energiou (do stabilného stavu).
- Výpočet končí, keď sieť skonverguje a je v stabilnom stave.

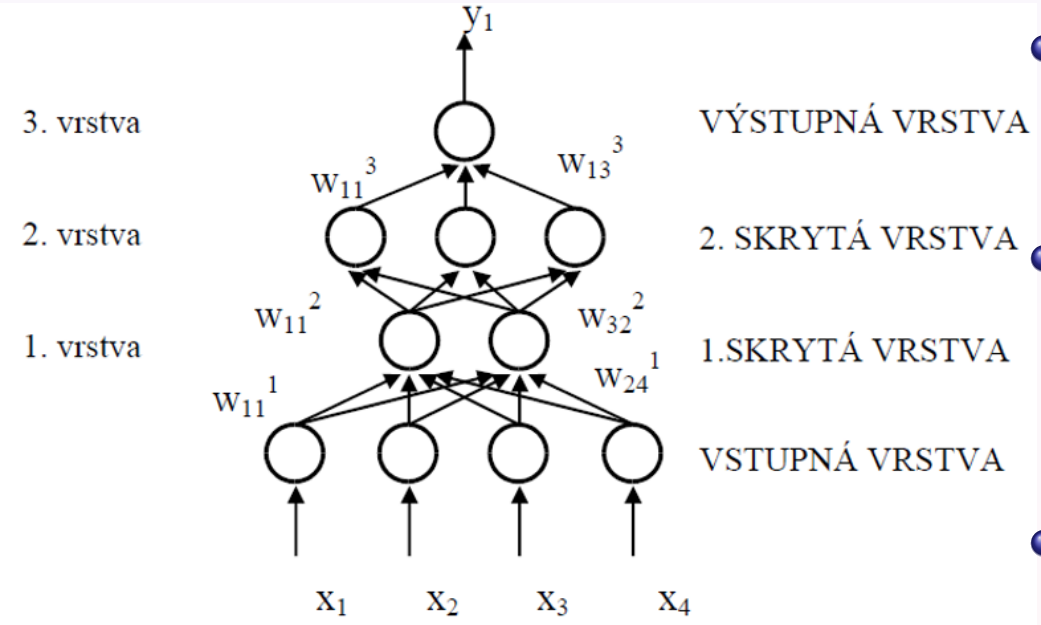
- Sieť tak postupne nájde zapamätaný obrázok, ktorý sa najviac podobá nedokonalému obrázku, ktorý bol daný na vstup.
- Dosiahne niekedy Hopfieldova sieť stabilný stav (skonverguje)?
- **Učenie** – zapamätanie vzorových obrázkov.
- HS – riešenie optimalizačných úloh.

Vrstvové siete

V roku 1986 David Rumelhart, Geoffrey Hinton a Ronald Williams publikovali v časopise *Nature* zásadný článok, v ktorom ukázali silné stránky trénovacieho algoritmu nazvaného spätné šírenie chyby (backpropagation).



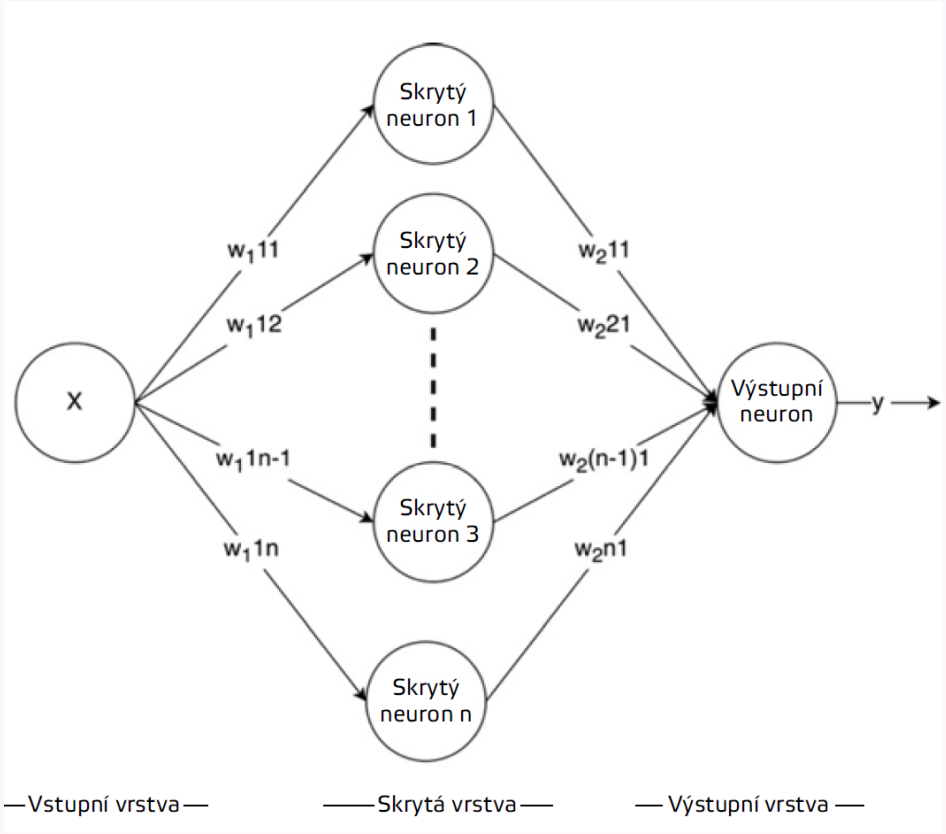
Vrstvové siete, algoritmus učenia



- Výpočet po vrstvách

- Kontrolované učenie, poznáme výsledok
- Rozdiel vypočítaného a známeho výsledku (chyba) znamená úpravu váh
- **Učenie:** Backpropagation algoritmus, rozloženie chýb na všetky váhy

Dostatočný počet neurónov v jednej vrstve dokáže aproximovať akúkoľvek spojitú funkciu



Úloha

Trénovanie neurónovej siete počítajúcej hodnoty danej funkcie v určenom intervale.
Trénovacie, t. j. vstupné príklady je potrebné pripraviť pomocou uvedenej funkcie (očakávané hodnoty sú vlastne funkčné hodnoty).
Postup pri riešení:

1. Generovanie syntetickej tréningovej vzorky (100 trénujúcich príkladov).
2. Vytvorenie vrstvovej neurónovej siete (návrh siete) s minimálne 2 skrytými vrstvami.
3. Príprava siete na učenie, $\eta = 0.03$, inicializácia náhodnými hodnotami.
4. Učenie siete pomocou BP trénovacieho algoritmu na všetkých príkladoch s presnosťou $\varepsilon = 0.3$. Určenie počtu iterácií.
5. Vyhodnotenie výsledkov po trénovaní pre 10x zopakované vstupy, pre 50x zopakované vstupy). Vyhodnotiť chybu pre každý vstup.
6. Porovnať grafické výstupy.