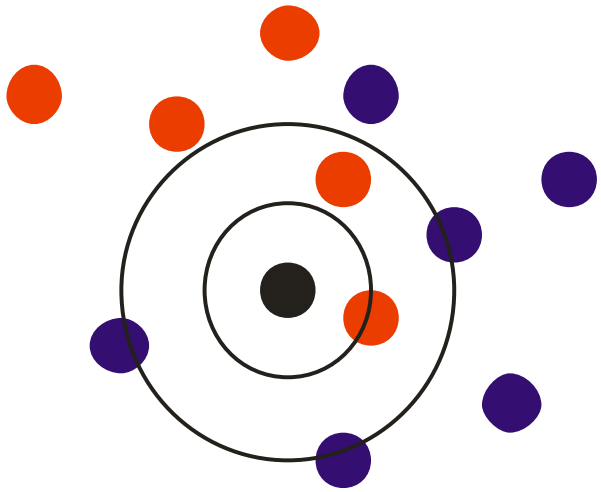
Keresés képi jellemzők alapján

Dr. Balázs Péter

SZTE, Képfeldolgozás és Számítógépes Grafika Tanszék

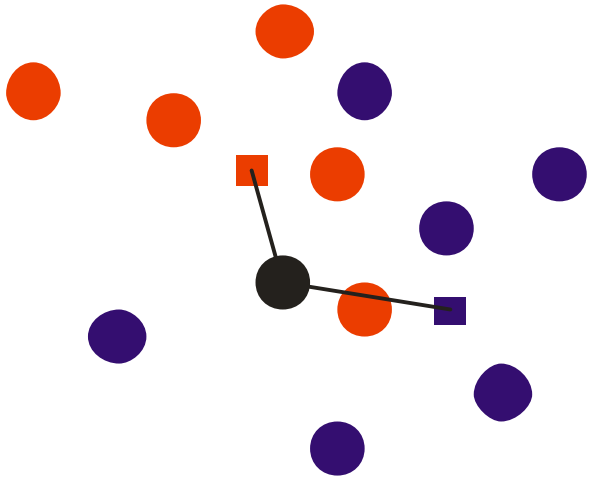
„Lusta” gépi tanulási algoritmusok



Osztályozás:

- k=1: piros
- k=5: kék

k-legközelebbi szomszéd (k=1,3,5,7)



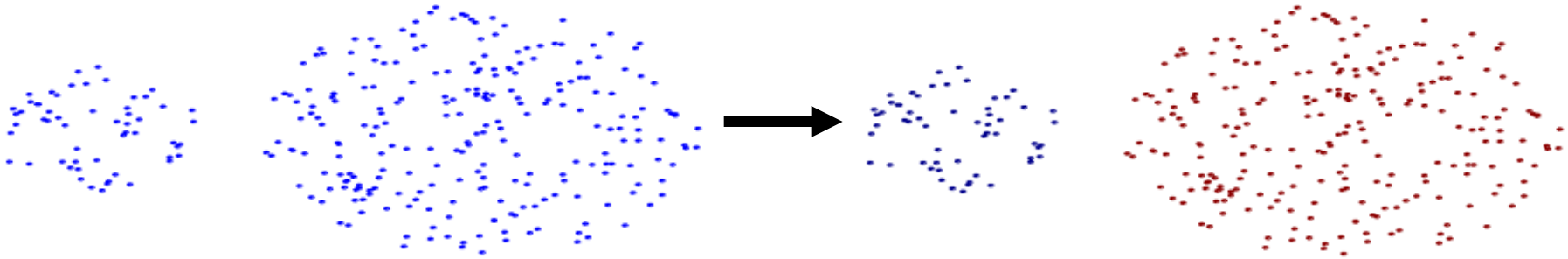
Osztályozás:

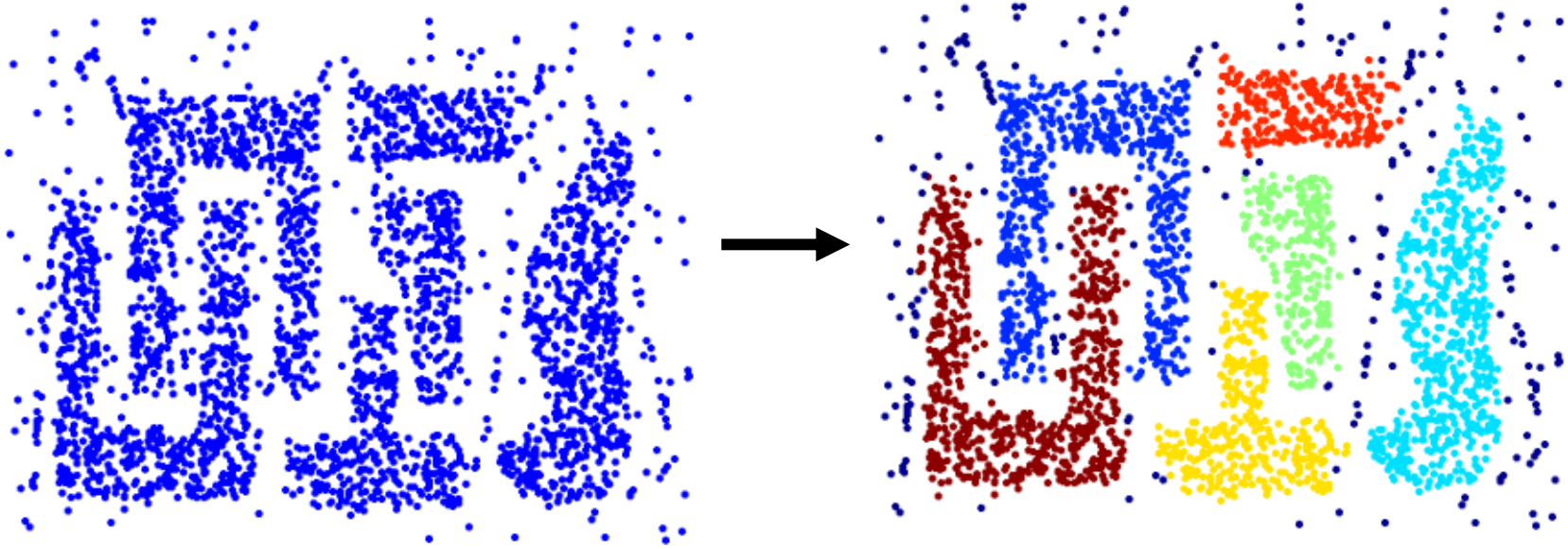
legközelebbi tömegközéppont

módosított k-means

A klaszterezés

- Az elemek csoportosítása „kompakt” halmazokba (az egyes halmazok elemei közel vannak egymáshoz).
- Mivel nem használunk fel (általában nincs is) osztálycímkét a jellemzőtér felosztásakor, így a klaszterezés *felügyelet nélküli tanulás*.





K-means algoritmus

- Adott a meghatározandó klaszterek (olyan részhalmazok, amelyek egymáshoz közeli pontokat tartalmaznak) maximális K darabszáma.
- Cél: Meghatározni K db középvektort - és hozzárendelni az input vektorokat a K db klaszterhez - úgy, hogy az átlagos eltérés a megfelelő középvektoroktól minimális legyen.
- A minimalizálandó célfüggvény:

$$J(X,C) = \sum_{j=1}^{K} E_j$$

$$E_j = \sum_{i=1}^{N_j} \| x_i - c_j \|^2$$

J-edik klaszterbe besorolt input vektorok

Klaszter középpont

Alapalgoritmus

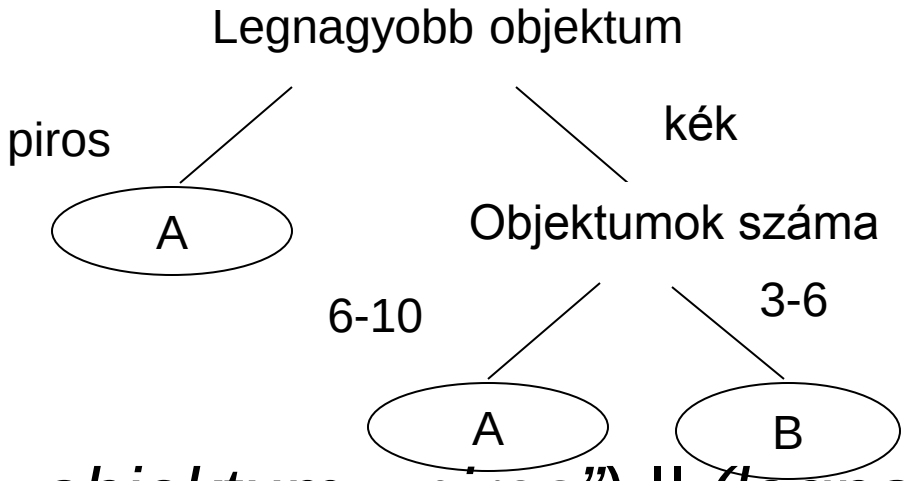
- http://stanford.edu/class/ee103/visualizations/kmeans/kmeans.html
- Inicializálás: alapesetben, a K db klaszter középpont inicializálása egyenletes eloszlás szerint
- Ciklus, egy megállási feltételig (iteráció szám, négyzetes hiba érték szerint):
  - Besorolási lépés: minden példát hozzárendelünk a hozzá legközelebbi klaszter középponthoz (a „legközelebbi” fogalom az aktuálisan használt távolságtól függ).
  - Klaszter középpontok újrabecslése: minden középpont helyzetét újrabecsüljük az előbbi besorolás alapján, a megfelelő középérték kiszámításával.

Döntési fa

- Input: tulajdonsághalmazzal leírt objektum
- Output: igen/nem döntés
- Belső pont: valamely tulajdonság tesztje
- Él: teszt lehetséges értékei
- Levél: milyen értéket kell adni, ha elérjük

Példa: Képkategóriák (A,B)

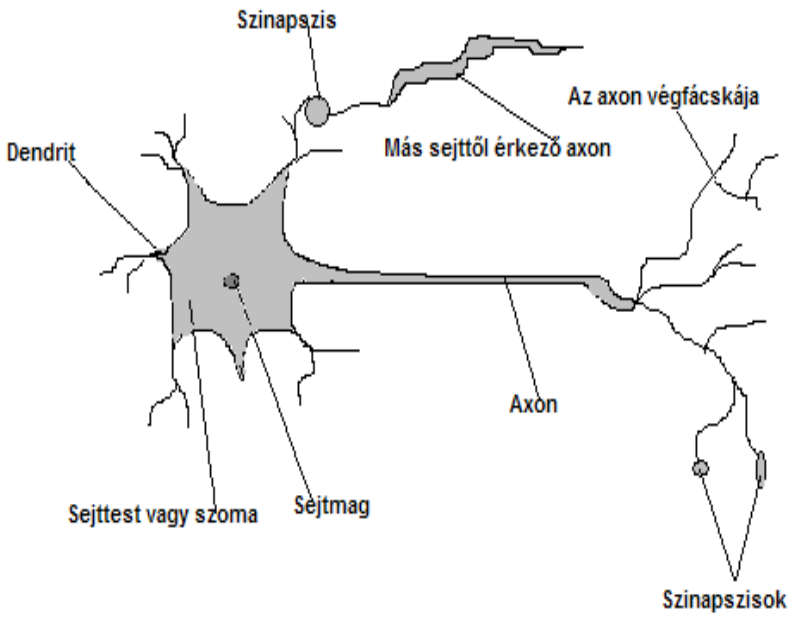
- Kategória (legnagyobb objektum színe, objektumok száma, háttérszín): A vagy B?



- *(legnagyobb_objektum=„piros”)* || *(legnagyobb objektum=kék && objektumok száma=„6-10”)* → *A*

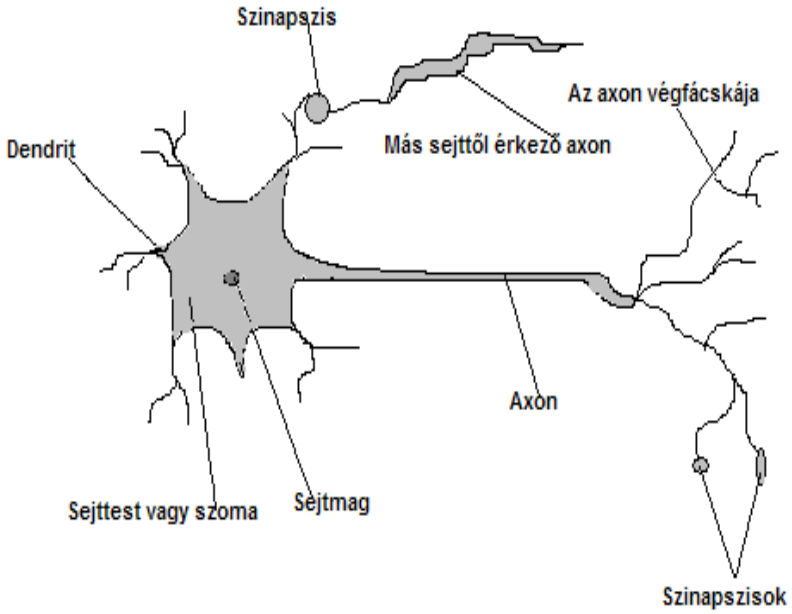
Az agy szerkezete

- több trillió, egymással kapcsolatban álló idegsejt (neuron)
- elektromos jelek (idegimpulzusok) gyors továbbítására képesek, a másodperc törtrésze alatt
- Minden idegsejt rendelkezik:
  - sejttesttel **(szoma)**
  - nyúlvánnyal **(dendrit)**
  - **axonnal**



Az agy szerkezete

- Az ingerület átadása a sejtek között a történik
- A neuronok új kapcsolatokat hozhatnak létre más neuronokkal.
- Úgy gondolják, ezeken a mechanizmusokon alapul a tanulási képesség.



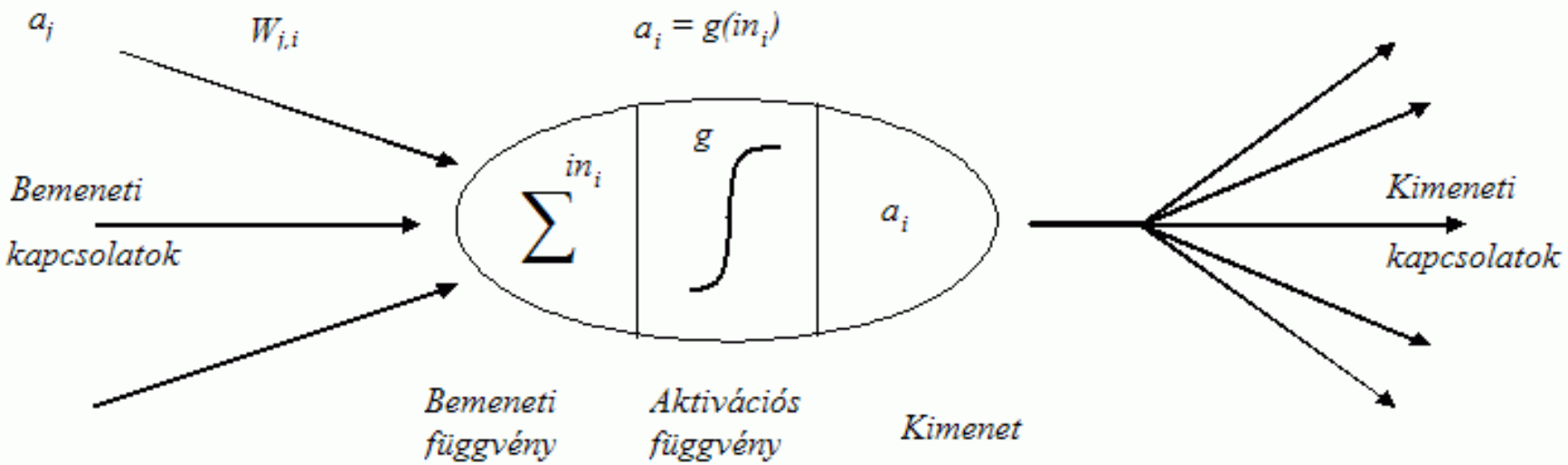
Mesterséges neurális hálózatok

- kapcsolatokkal összekötött csomópontokból (unit, processzáló egység) épül fel. Mindegyik kapcsolathoz tartozik egy hozzárendelt numerikus érték, súly.
- Néhány processzáló egység a külső környezethez kapcsolódik, és bemeneti vagy kimeneti egységként szolgál, mások rejtett rétegekben vannak.
- Minden egység rendelkezik más egységektől érkező bemeneti kapcsolatokkal, más egységekhez menő kimeneti kapcsolatokkal és egy pillanatnyi **aktivációs szinttel**.
- Minden egység egy egyszerű számítást végez (aktivációs érték számítása)

Neurális hálózatok

A számítás két lépése: lineáris lépés vagy bemeneti függvény, és nemlineáris lépés vagy aktivációs függvény.

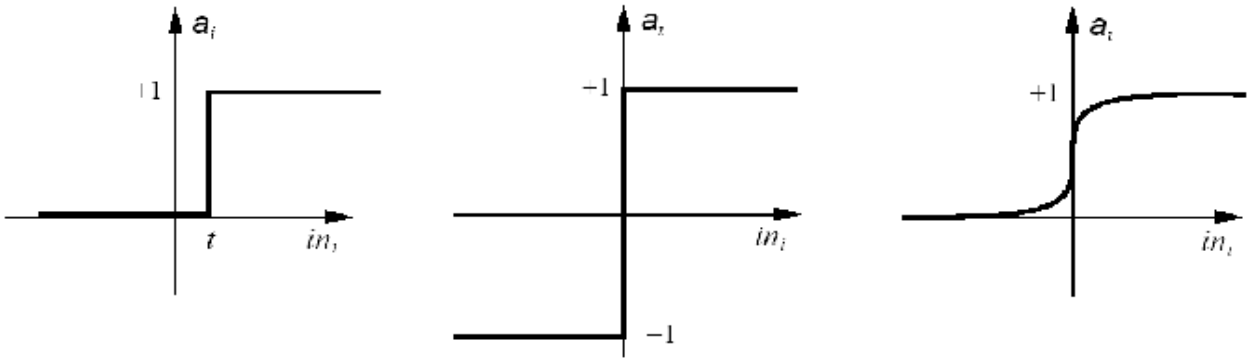
- **Lineáris lépés**
- **Nemlineáris lépés**



Aktivációs függvény

$$a_i = g(\sum_j w_{i,j} a_j)$$

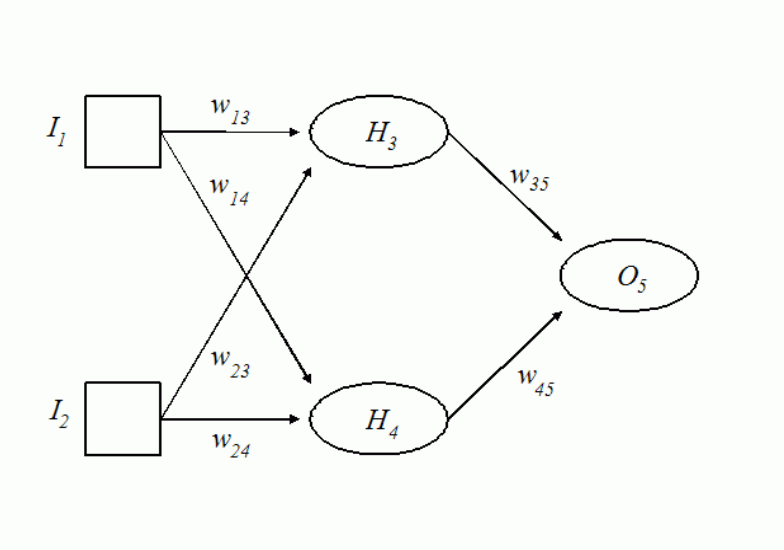
- Szokásos aktivációs függvények:
  - Előjelfüggvény (sgn)
  - Lépcsős függvény
  - Szigmoid függvény $f(x) = 1 / (1+e^x)$



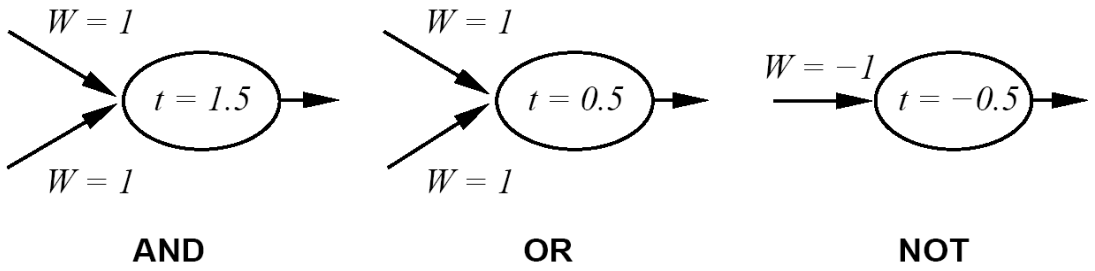
(a) Step function (b) Sign function (c) Sigmoid function

Hálózati struktúrák

- **Előrecsatolt hálókban** a kapcsolatok egyirányúak és nincsen hurok a hálóban.
- A **visszacsatolt hálókban** a kapcsolatok által kialakított topológia tetszőleges.
- rétegekbe szervezettek hálózatok



Logikai függvények



- Használatukkal tetszőleges logikai függvényt kiszámító háló építhető

Backpropagation algoritmus

- http://galaxy.agh.edu.pl/~vlsi/AI/backp_t_en/backprop.html
- A kimeneti hiba minimalizálása érdekében gradiens alapú keresést végez a súlyok terében
- Lokális minimumhoz tart
- A megfigyelt hiba felhasználásával számítsuk ki a kimeneti egységekre a Δ (hibatag) értékeket
- A kimeneti réteggel kezdve ismételjük a háló mindegyik rétegére, amíg a legelső rejtett réteget el nem érjük:
  1. Terjesszük vissza Δ értékeket az előző rétegre
  2. Módosítsuk a két réteg közötti súlyokat

Előnyök

- Egyszerű, könnyen megérthető működési elv
- a többrétegű hálók osztálya az attribútumok bármely függvényének reprezentációjára képes
- a neurális hálók jó általánosításra képesek
- jól tolerálják a zajosságot (ellentmondó példák, hiányos attribútumok)
- hatékonyan párhuzamosítható
- alkalmazni tudják a felhasználó előzetes ismereteit.

Hátrányok

- átláthatatlan, a súlyokból semmilyen használható információ nem nyerhető ki
- a tanuláshoz nagyon sok példára van szükség
- a tanulás hosszú időt vehet igénybe