Clasificare Non-parametrică

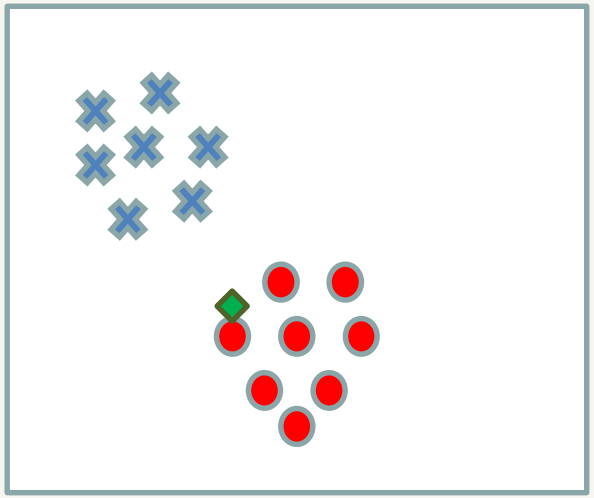
Cel mai apropiat vecin

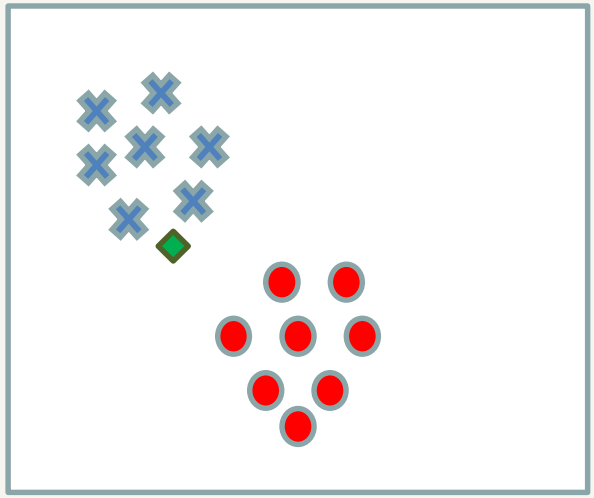
Date și model

- Clasificare
- Setul de antrenament conține **vectorii X_i** cu **etichete Y_i**
- Nu există proces de învțare în acest caz
- Datele sunt doar stocate
- Testare:
  - Un exemplu din setul de antrenament capătă eticheta exemeplului din setul de antrement de care este **cel mai apropiat**

Distanță

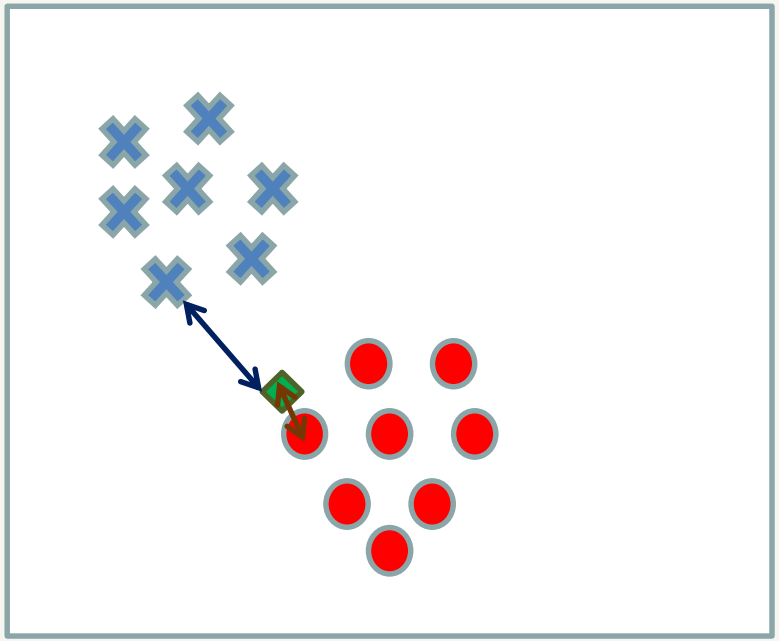
În ce clasă ar trebui să fie punctul verde? De ce?





Distanță

În ce clasă ar trebui să fie punctul verde? De ce?



Distanța de la punctul curent la punctul albastru

Distanța de la punctul curent la punctul rosu

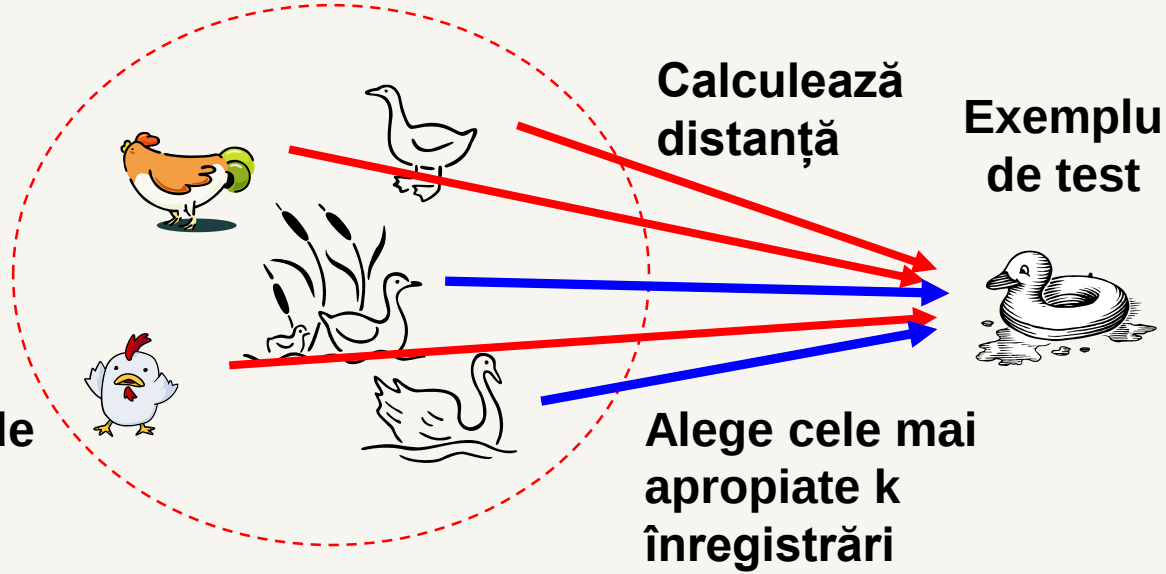
Distanța eucldiană

$$d(\vec{x_1}, \vec{x_2}) = (x_{12} - x_{21})^2 + (x_{21} - x_{22})^2$$

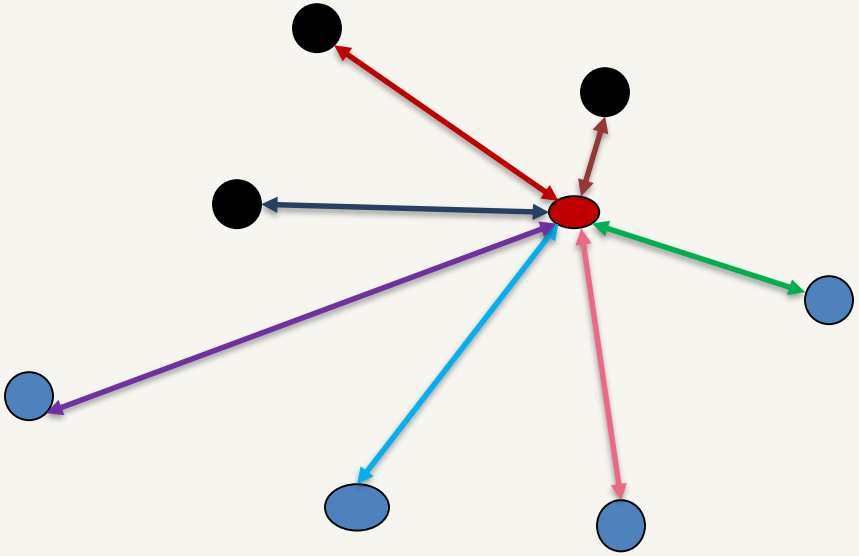
$$\vec{x_1} = [x_{11}, x_{12}],\ \vec{x_2} = [x_{21}, x_{22}]$$

Cel mai apropiat vecin

- Idee:
  - “Dacă merge ca o rață, măcăne ca o rața, probabil e rață”



Cum funcționează?

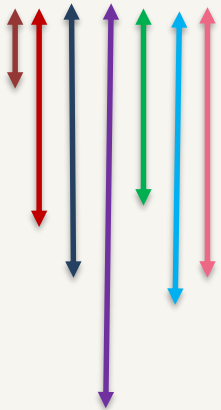


Date (noi) de test

Date de antrenare din clasa 1

Date de antrenare din clasa 2

Calculează distanța

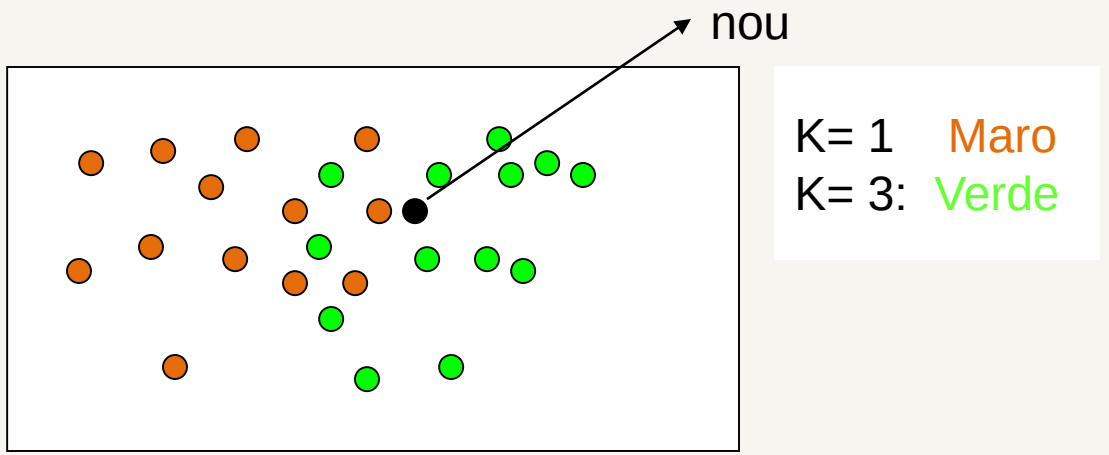


Alege-o pe cea mai mică

Cei mai apropiați k-vecini

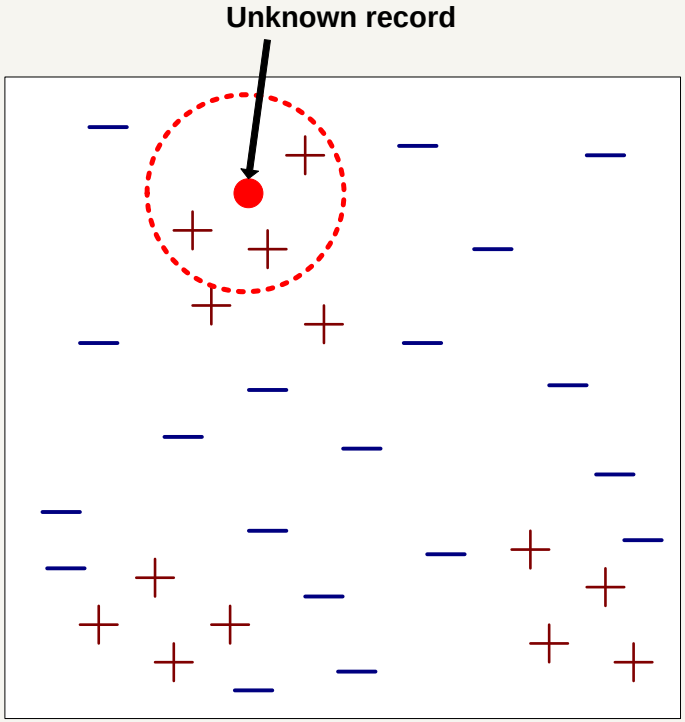
- Considerăm nu doar pe primul (**1**) ci pe primii **k**
- Votul majorității pentru Cei mai apropiați k-vecini

$$\hat{Y}(x) = \frac{1}{k} \sum_{x_i \in N_k(x)} y_i$$



K= 1 Maro
K= 3: Verde

Clasificator de tipul “Cei mai apropiați K-Vecini”

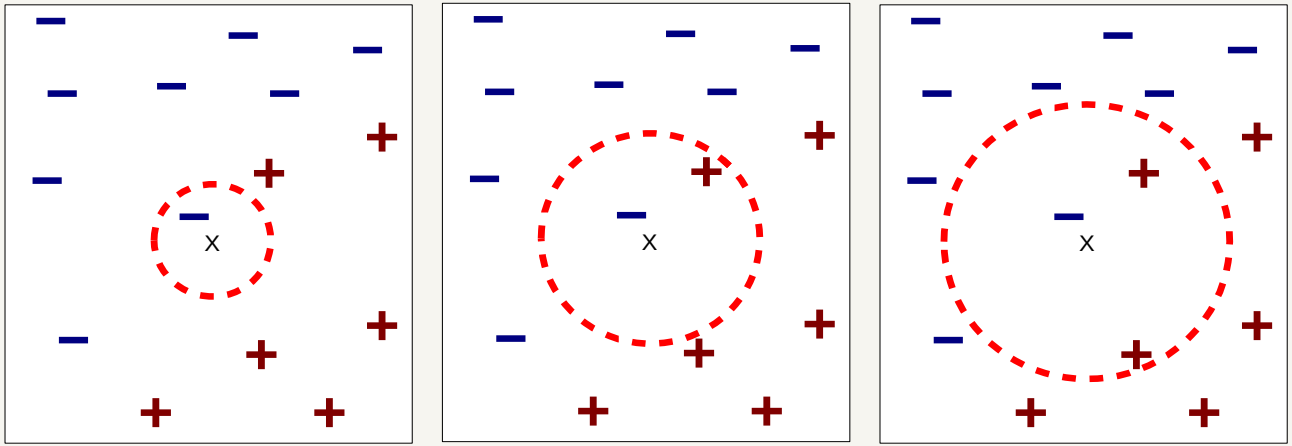


E nevoie de 3 lucruri

- Ținem minte datele - **memorie**
  - distanță de calculat la toți vecinii - **timp**
  - Valoarea lui ***k***, (numărul de vecini de întors)

- Pentru a clasifica o data noua:
  - Se calculează distanța la **toate** celelalte date din setul de antrenament
  - Se identifică cel mai apropiați k vecini
  - Pe baza etichetelor vecinilor selectați se determină clasa datei necunoscute (prin alegerea votului **majoritar**)

Definitia celui mai apropiat vecin

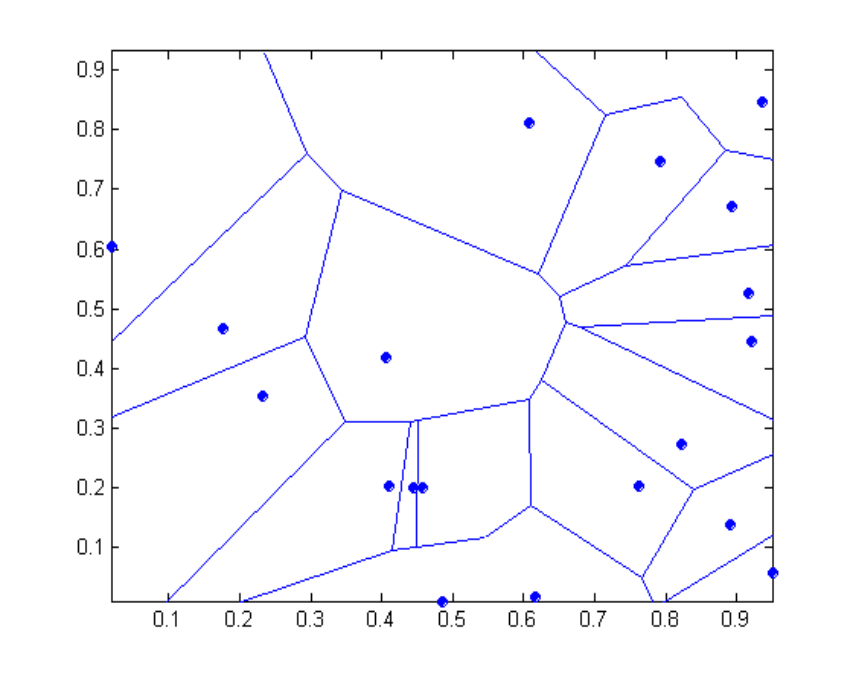


(a) 1-nearest neighbor (b) 2-nearest neighbor (c) 3-nearest neighbor

Cei mai apropiați k vecini a unei date **x** sunt punctele care au cea mai mică distanță la **x**

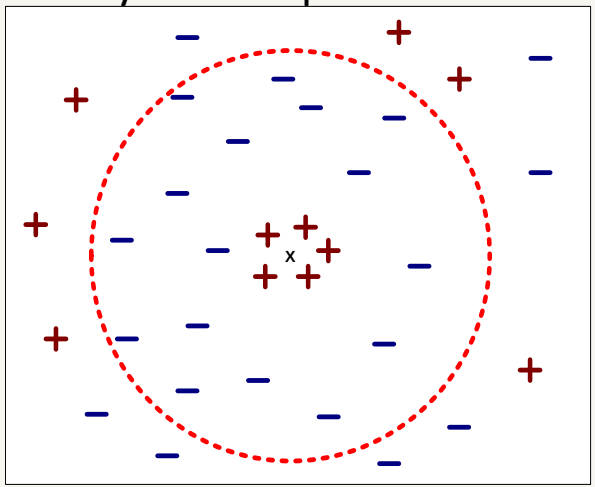
1 nearest-neighbor

Voronoi Diagram is partitioned according to the influence of each existing point



Cel mai apropiat vecin Classification…

- Choosing the value of k:
  - If k is too small, sensitive to noise points
  - If k is too large, neighborhood may include points from other classes



Distanțe

- Distanța euclideană (L_2)

$$d_2(p,q) = \sqrt{\sum_i (p_i - q_i)^2}$$

- Distanța Manhattan (L_1)

$$d_1(p,q) = \sum_i |p_i - q_i|$$

- Distanța Chebyshev (L_∞) :

$$d_\infty(p,q) = \max_i |p_i - q_i|$$

- Distanța Cosine :

$$d_{\cos}(p,q) = \frac{\sum_{i=1}^{N} p_i q_i}{\sqrt{\sum_{i=1}^{N} p_i} \sqrt{\sum_{i=1}^{N} q_i}}$$

- Coefficient Hellinger (distanța)

$$d_H(p,q) = \frac{1}{\sqrt{2}} \sqrt{\sum_i (\sqrt{p_i} - \sqrt{q_i})^2}$$

- Divergență Kullback-Leibler

$$d_{KL}(p,q) = \sum_i p_i \log \frac{p_i}{q_i}$$

Masuri de similaritate

- Funcționeaza în opoziție distanțelor: mai mică e mai rău
  - Coefficientul de corelație Pearson

$$r(p,q) = \frac{N\sum_{i=1}^{N} p_i q_i - \sum_{i=1}^{N} p_i \sum_{i=1}^{N} q_i}{\sqrt{N\sum_{i=1}^{N} p_i^{\,2} - \left(\sum_{i=1}^{N} p_i\right)^2}\sqrt{N\sum_{i=1}^{N} q_i^{\,2} - \left(\sum_{i=1}^{N} q_i\right)^2}}$$

  - |r| între 0 și 1

Scala atributelor

- Problema scalei
  - Atributele (dimensiunile vectorile de date) pot fi scalate astfel încât săa prevină distanță să fie dominată de un atribut
  - Exemplu:
    - înălțimea unei persoane poate varia de la 1.5m la 1.8m
    - greutatea unei persoane 45kg la 180kg
    - venitul unei persoane de la 1000 ron la 1M
- Normalizare: aducem toate atributele în intervalul [0,1]

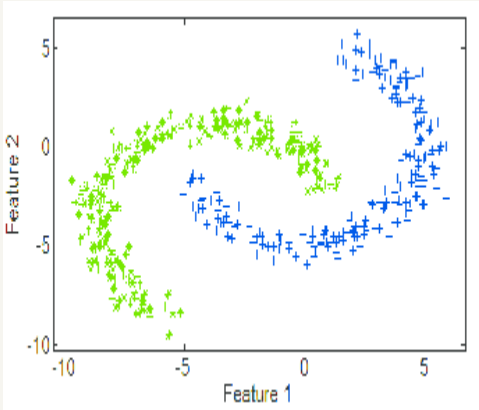
K-NN cu reprezentanți

În loc de *Cel mai apropiat vecin* peste întregul set de date, putem folosi doar câțiva reprezentați pentru fiecare clasă

Un reprezentant per clasă

- Aceștia pot fi centroizii (media) fiecărei clase
  - Media este centrul de greutate al punctelor din nor
  - Se discută în detaliu la Grupare

Uneori e bine să fie mai mulți reprezentanți pe clasă



Cel mai apropiat vecin …

- k-NN este un sistem leneș (lazy learner)
  - Nu construiește un model explicit
  - Leneș (Lazy) = nu încearcă să înțeleagă structura datelor
  - Determinarea unei etichete este costisitoare

GRUPARE

CLUSTERING

Învațare nesupervizată

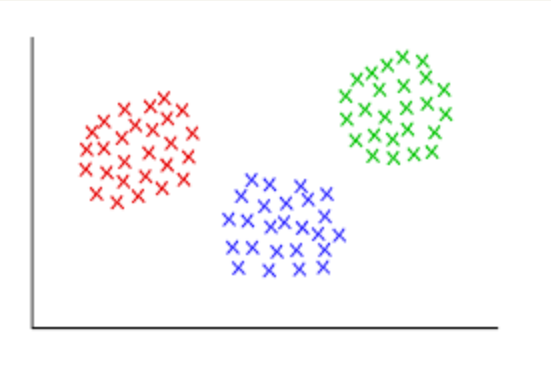
- Setul de antrenament nu are etichete
  - Este același lucru cu a avea etichete dar le ignorăm
  - Scopul este să înțelegem și să organizăm mai bine datele
    - Construim un model
  - În testare, o nouă dată este structurată conform unui model învățat

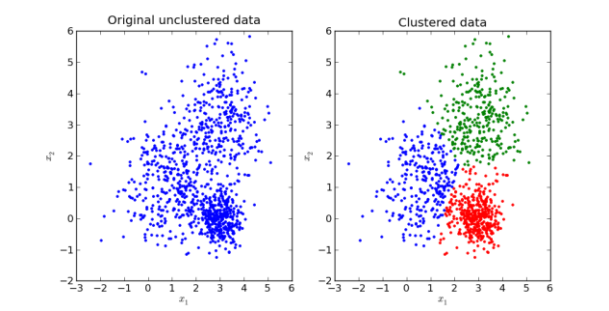
Învațare nesupervizată

- Grupare (Clustering)
  - Datele sunt organizate în **grupuri** (clusters)
  - De ce: să reducem cantitatea, să o structurăm
- Data este structurată conform unei optimizarea unei funcții obiectiv

Grupare

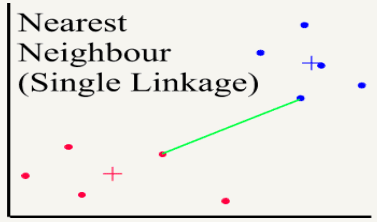
- Data este grupată în clustere bazată pe similaritate
- Datele dintr-un cluster trebuie să fie **similare** una cu altul
- Datele din clustere **diferite** trebuie să fie suficient de **asimilare**
- Avem din nou o măsura a similaritate



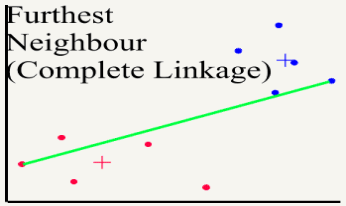


Distanțe folosite în grupare

$$D_{\min}(C_i, C_j) = \min_{x \in C_i,\ y \in C_j} \|x - y\|^2$$



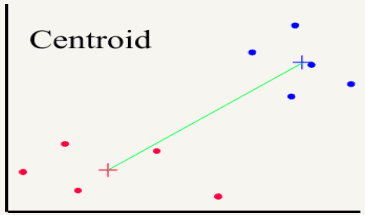
$$D_{\max}(C_i, C_j) = \max_{x \in C_i,\ y \in C_j} \|x - y\|^2$$



Se evită grupurile alungite.

$$D_{\text{avg}}(C_i, C_j) = \frac{1}{|C_i||C_j|} \sum_{x \in C_i,\ y \in C_j} \|x - y\|^2$$

$$D_{\text{means}}(C_i, C_j) = \|\mu_i - \mu_j\|^2$$



Grupare de tip K-means

- Un algoritm simplu
  - Foarte popular
- Iterează între
  - Se actualizează cărui cluster îi aparține fiecare dată
  - Se actualizează sumarul grupului
    - Fiecare cluster are un ***centroid*** (dată unică care reprezentă întregul grup)
- La sfârșit avem o structură și un model astfel încât un punct nou este atașat unui noi cluster

Grupare de tip K-means

- Să presupunem că avem ***K*** grupuri, ***c=1...K***
  - Reprezintă grupurile prin locații $^{1}_{c}$
  - Exemplului i este reprezentat de vectorul $\boldsymbol{x_i}$
  - Se reprezintă asocierea date $\boldsymbol{i^{th}}$ la un grup prin variabila $\boldsymbol{z_i}$ in ***1...K***
- Se iterează până la convergență:
  - Pentru fiecare dată, se identifică cel mai apropiat grup (Cel mai apropiat vecin !)

$$z_i = \arg\min_c \|x_i - \mu_c\|^2 \qquad \forall i$$

  - Pentru fiecare grup se calculează media tuturor punctelor care sunt atașate lui:

$$\forall c, \qquad \mu_c = \frac{1}{N_c} \sum_{i \in S_c} x_i \qquad\qquad S_c = \{i : z_i = c\},\ N_c = |S_c|$$

K-means : Utilizare

- Pregătim datele pentru clasificări viitoare
  - etichete vor apărea mai târziu
- Pentru clasificatorul de tip “Cel mai apropiat vecin” – se memorează doar centroizii
  - Cel mai apropiat vecin cu reprezentanți

Choosing the number of clusters

- With cost function

$$C(\underline{z}, \underline{\mu}) = \sum_i \|x_i - \mu_{z_i}\|^2$$

  what is the optimal value of k?
  (can increasing k ever decrease the cost?)

The cost is the (mean) square error between each data instance and the centroid of its cluster

- This is a model complexity issue
  - Much like choosing lots of features – they only (seem to) help
  - But we want our clustering to *generalize* to new data

- One solution is to penalize for complexity
  - Bayesian information criterion (BIC)
  - Add (# parameters) * log(N) to the cost
  - Now more clusters can increase cost, if they don't help "enough"