

Curs 7-8:

Gruparea datelor

Structura

- Gruparea datelor
 - Concepte de bază
 - Evaluarea calității grupării
- Algoritmi partiționali
 - kMeans
 - Fuzzy cMeans
- Algoritmi ierarhici
 - Algoritmi aglomerativi
 - Algoritmi divizivi
- Algoritmi bazați pe estimarea densității datelor
- Algoritmi bazați pe modele probabiliste

Scopul grupării (reminder)

Ce se cunoaște?

- un **set de date** (nu neapărat structurate)
- O măsură de **similaritate/disimilaritate** între date (măsura e specifică problemei de rezolvat) pe baza căreia se construiește o **matrice de similaritate/disimilaritate**

Ce se dorește?

- Un **model** care descrie modul în care se **grupează datele** în clustere (grupuri) astfel încâte datele care aparțin aceluiași cluster sunt mai similare între ele decât cele care aparțin unor clustere diferite

Care este scopul final?

- Să se poată verifica dacă două date aparțin aceluiași cluster
- Să se determine clusterul de care aparține o dată

Scopul grupării (reminder)

Exemple:

- **Customer segmentation** = identificarea grupurilor de clienți care au comportamente similare
- **User profiles extraction** = identificarea grupurilor de utilizatori ai unui serviciu web caracterizați prin comportamente similare
- **Document clustering** = identificarea unor grupuri de documente similare din punct de vedere al conținutului
- **Image segmentation** = identificare unor regiuni omogene într-o imagine
- **Gene expression analysis** = gruparea genelor cu roluri similare

Gruparea permite:

- sumarizarea și/sau vizualizarea datelor în alt mod cu scopul de a înțelege mai bine datele

Particularități ale grupării

Este un proces nesupervizat:

- Setul de antrenare conține doar valori ale atributelor
- Eticheta clasei nu e cunoscută

Problema grupării este rău-definită:

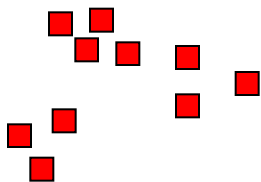
- Identificarea grupurilor este dificilă
- Decizia poate avea caracter subiectiv



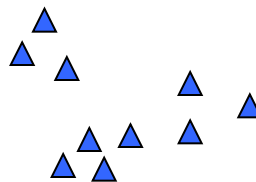
Câte clustere?



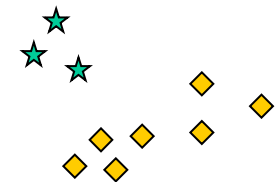
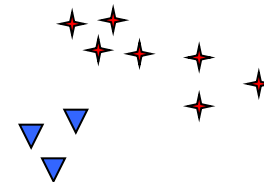
Șase clustere



Două clustere



patru clustere



Concepte de bază

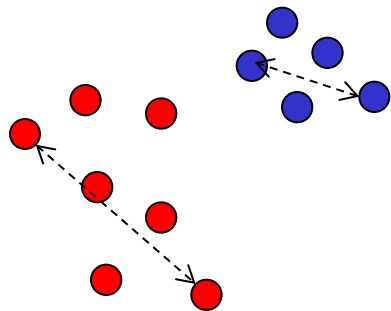
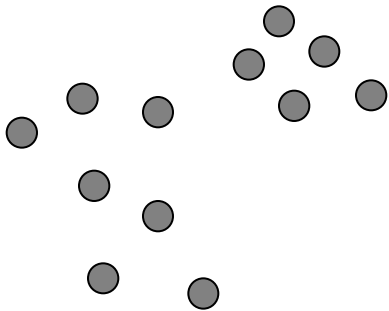
- **Cluster** = grup de date care sunt similare
- **Matrice de (di)similaritate** pt un set de n date = matrice $n \times n$ in care pe linia i coloana j se află valoarea similarității/disimilarității între data i și data j
- **Clustering (grupare)** = proces de identificare a clusterelor
- **Prototip (reprezentant)**= “obiect” reprezentativ pentru datele dintr-un cluster
 - **Centroid** = media datelor dintr-un cluster – centroidul nu este neapărat un element din setul de date
 - **Medoid** = data din cluster care este cea mai apropiată de media clusterului – medoidul aparține setului de date
- **Raza clusterului** = media distanțelor dintre datele din cluster și prototipul acestuia
- **Diametrul clusterului** = distanța (disimilaritatea) maximă dintre oricare două date ale clusterului

Tipuri de clustering

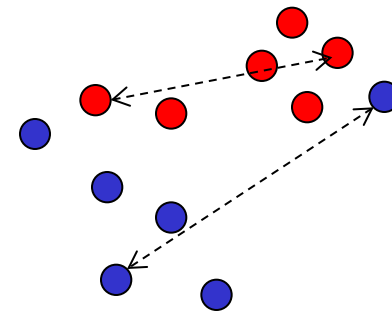
- Crisp vs fuzzy clustering
 - Crisp clustering = fiecare dată aparține unui singur cluster
 - Fuzzy clustering = o dată poate aparține mai multor cluster (grad de apartenență pentru fiecare cluster)
- Flat vs hierarchical clustering
 - Flat (partitional) clustering = rezultatul este un set de cluster (o partiție)
 - Hierarchical clustering = rezultatul este o ierarhie de partiții
- Variante de algoritmi
 - Algoritmi partiționali (ex: kMeans, Fuzzy cMeans)
 - Algoritmi hierarhici (alg. aglomerativi, alg. divizivi)
 - Algoritmi bazați pe densitate (ex: DBSCAN)
 - Algoritmi bazați pe modele probabiliste (ex: EM = Expectation Maximization)

Măsuri de calitate

- Nu există un indicator unic pentru evaluarea calității unei grupări
- Cea mai comună abordare constă în estimarea:
 - **Compacității** clusterelor (variabilitate **intra-cluster** – ar trebui să fie mică)
 - Gradului de **separare** dintre datele aparținând unor clusterare diferite (variabilitate **inter-cluster** – ar trebui să fie mare)



Calitate bună



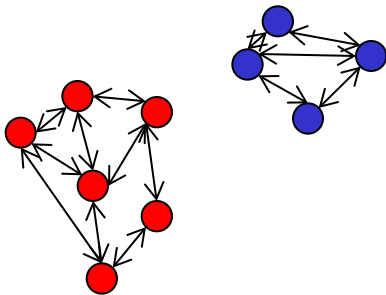
Calitate slabă

Măsuri de calitate

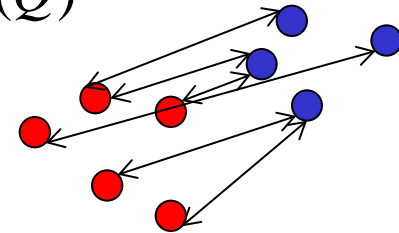
- Intra-cluster to inter-cluster distance ratio = Intra/Inter (valori mici corespund unei calități mai bune)
- Fie P setul de date ce aparțin aceluiași cluster și $Q = D \times D - P$ (restul perechilor: o dată din pereche aparține unui cluster iar cealaltă unui alt cluster)

$$Intra = \sum_{(x_i, x_j) \in P} d(x_i, x_j) / card(P)$$

$$Inter = \sum_{(x_i, x_j) \in Q} d(x_i, x_j) / card(Q)$$



Exemple de perechi de date utilizate în calculul distanțelor intra-cluster



Exemple de perechi de date utilizate în calculul distanțelor inter-cluster

Măsuri de calitate

- Silhouette coefficient

Obs:

$$S_i = \frac{D \min_i^{out} - D_{avg_i}^{in}}{\max\{D \min_i^{out}, D_{avg_i}^{in}\}}$$

- S ia valori în (-1,1)
- **Valori mai mari** indică o grupare mai bună

$$S = \frac{1}{n} \sum_{i=1}^n S_i$$

$D_{avg_i}^{in}$ = media distantelor dintre
 x_i și celelalte date din clusterul lui x_i

$D_{avg_i}^j$ = media distantelor dintre
 x_i și datele din clusterul j ($j \neq i$)

$$D \min_i^{out} = \min_j D_{avg_i}^j$$

kMeans

- **Input:** set de date $D=\{x_1, x_2, \dots, x_N\}$, $K = \text{nr de clustere}$
- **Output:** partiție $P=\{C_1, C_2, \dots, C_K\}$ a setului de date D

kMeans (D,k)

initialize the centroids c_1, c_2, \dots, c_K (by **random** selection from the data set)

repeat

- **assign** each data from D to the cluster corresponding to the closest centroid (with respect to a similarity/distance)
- **update** each centroid as mean of the data belonging to the corresponding cluster

until <the partition does not change>

kMeans

- **Caracteristici**

- kMeans este o metodă bazată pe prototipuri care are ca scop minimizarea is sumei pătratelor erorilor (SSE) – distanțele dintre date și centroizii corespunzători

$$SSE = \sum_{k=1}^K \sum_{x \in C_k} d^2(x, c_k) = \sum_{k=1}^K \sum_{x \in C_k} \sum_{j=1}^n (x_j - c_{kj})^2$$

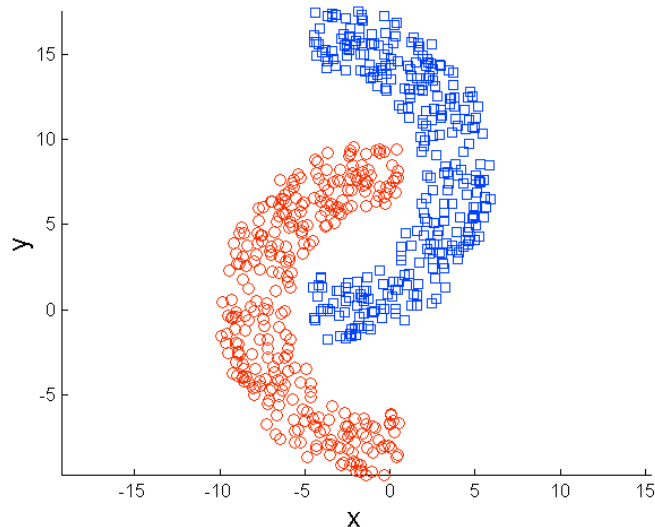
(în cazul distanței euclidiene)

- **Complexitate:** $O(n \cdot N \cdot K \cdot \text{iterații})$ (n =nr de atribute, N =nr de date, K =nr de clustere)
- **Pre-procesare utilă:** normalizare
- **Post-procesare utilă:**
 - Eliminarea clusterelor mici
 - Fragmentarea clusterelor caracterizate prin variabilitate mare
 - Reunirea clusterelor apropiate

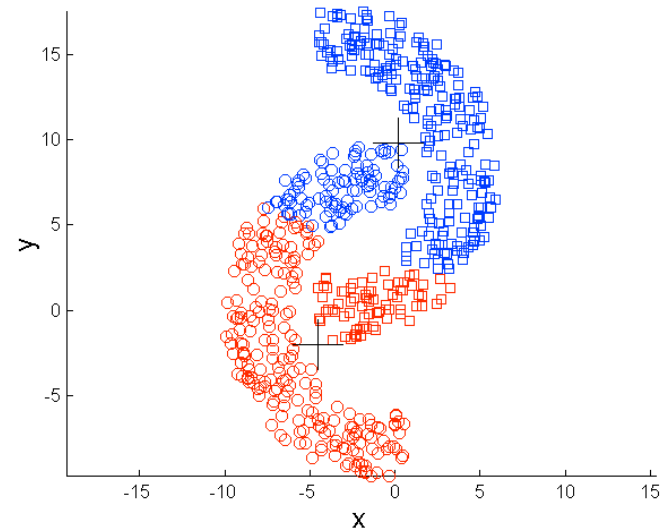
kMeans

Limite:

- Nu funcționează bine dacă datele nu sunt “sferice” sau bine separate (distribuțiile de probabilitate corespunzătoare clusterelor nu sunt neapărat normale)
 - **Soluție:** utilizarea altor abordări (e.g. clustering bazat pe densitate)
- Necesită specificarea numărului de cluster.
 - **Soluție:** se aplică pentru mai multe valori ale numărului de cluster



Clusterelor reale



Rezultatul Kmeans

kMeans

Limite:

- Rezultatul poate fi ușor influențat de erorile din date sau de datele atipice
 - **Soluție:** utilizarea medianei în locul mediei în construirea centroizii (reprezentanții clusterilor)
 - **kMedian** – se determină mediana la nivel de componentă (corespunde cazului în care se utilizează distanța Manhattan)
 - **kMedoid** – se determina elementul din setul de date care este cel mai apropiat de media clusterului

kMeans

Limite: necesită cunoașterea apriori a numărului de clustere

- **Soluții:**
 - aplică algoritmul pt diferite valori ale lui K și selectează varianta care corespunde celor mai bune valori ale criteriilor de calitate
 - post-procesarea rezultatelor procesului de clustering prin partiționarea clusterelor cu variabilitate mare și reunirea clusterelor apropiate (ex: alg. **ISODATA**)

ISODATA

Idei principale ale alg ISODATA

- Dacă dimensiunea unui cluster este mai mică decât N_{min} (un parametru care specifică numărul minim de date dintr-un cluster) atunci clusterul ar trebui reunit cu cel mai apropiat alt cluster
- Dacă distanța dintre două cluster (de exemplu the distanța dintre prototipurile clusterelor) este mai mică decât D_{min} (un parametru care specifică distanța minimă dintre cluster) atunci clusterele ar trebui reunite
- Dacă varianța unui cluster este mai mare decât V_{max} și numărul de date conținute este mai mare decât $2 \cdot N_{min}$ atunci clusterul poate fi divizat în două alte cluster:
 - Identifică atributul j pt care varianța este maxmă
 - Din prototipul c_k sunt construite două alte prototipuri c' și c'' prin înlocuirea valorii atributului j din c_k cu $c_k(j)-b$ respectiv $c_k(j)+b$, r (b este un parametru setat de către utilizator)

Fuzzy cMeans

Ideea grupării fuzzy (soft):

- O dată nu aparține unui singur cluster ci poate aparține mai multor cluster (cu un anumit grad de apartenență pentru fiecare cluster)
- Rezultatul unei grupări fuzzy este o matrice M de dimensiune $N \times K$
($N =$ nr date, $K =$ nr cluster);
 $M(i,j)$ = o valoare din $[0,1]$ care corespunde gradului de apartenență a datei i la clusterul j

Obs: Fuzzy cMeans poate fi utilizată în segmentarea imaginilor

Fuzzy cMeans

Algorithm

- Initialize the membership matrix (M)
- **Repeat**
 - Compute the centroids ($c_k, k=1, \dots, K$)
 - Update the membership values ($m_{ij}, i=1, \dots, N, j=1, \dots, K$)
- until** <no significant changes in the membership function>

Obs: dacă e necesar fiecare dată este asignată la clusterul ce corespunde celui mai mare grad de apartenență

Calculul centrozilor

$$c_j = \frac{\sum_{i=1}^n M_{ij}^p x_i}{\sum_{i=1}^n M_{ij}^p}, \quad j = \overline{1, K}$$

$p > 1$ is a parameter (e.g. $p=2$)

Calculul gradului de apartenență

$$M_{ij} = \frac{1}{\|x_i - c_j\|^{2/(p-1)} \sum_{k=1}^K 1 / \|x_i - c_k\|^{2/(p-1)}}$$

$i = \overline{1, n}, j = \overline{1, K}$

Algoritmi ierarhici

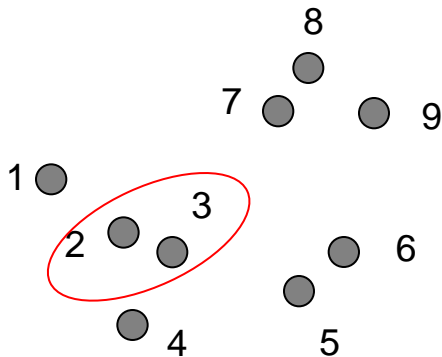
Obs: una dintre principalele limite ale algoritmilor partiționali e faptul că necesită cunoașterea numărului de clustere.

Altă abordare: se construiește o ierarhie de partiții – conduce la o structură arborescentă numită **dendrogramă**

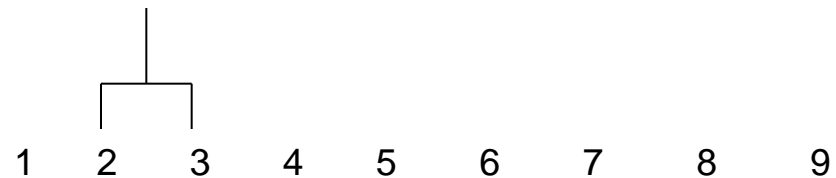
- In varianta **bottom-up** (metoda **aglomerativă**)
 - Se pornește cu o partiție de clustere ce conțin fiecare câte o singură dată (reprezintă frunze în arbore)
 - Se reunesc clusterelor care sunt similare între ele – procesul de reunire se repetă până se ajunge la un singur cluster (reprezintă rădăcina arborelui)
- In varianta **top-down** (metoda **divizivă**)
 - Se pornește cu o partiție ce conține un singur cluster (cu toate datele)
 - Se partiționează clusterii mari aplicând o tehnica partițională (ex: kMeans) – procesul continuă până când se ajunge la clustere ce conțin câte o singură dată.

Metoda aglomerativă

Idee: se identifică la fiecare etapă care sunt cele mai similare clusteruri și se reunesc

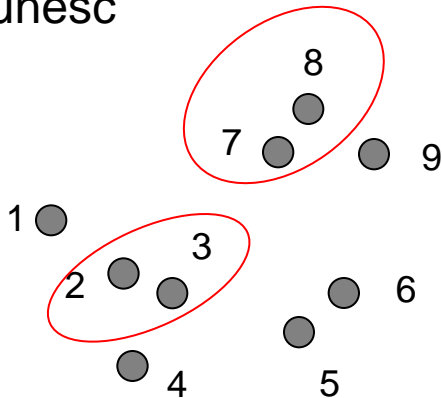


	1	2	3	4	5	6	7	8	9
1	0	2	3	4	7	8	6	8	10
2	2	0	1	2	4	6	7	8	9
3	3	1	0	2	3	5	6	8	9
4	4	2	2	0	3	6	9	10	11
5	7	4	3	3	0	1	4	6	5
6	8	6	5	6	1	0	3	4	3
7	6	7	6	9	4	3	0	1	2
8	8	8	8	10	6	4	1	0	2
9	10	9	9	11	5	3	3	2	0

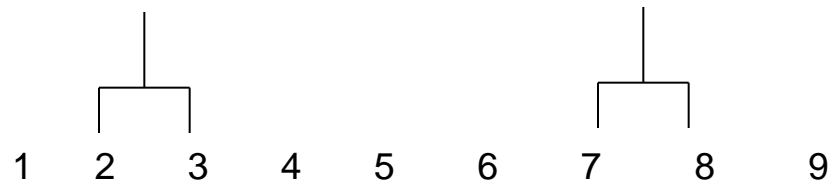


Metoda aglomerativă

Idee: se identifică la fiecare etapă care sunt cele mai similare clusteruri și se reunesc

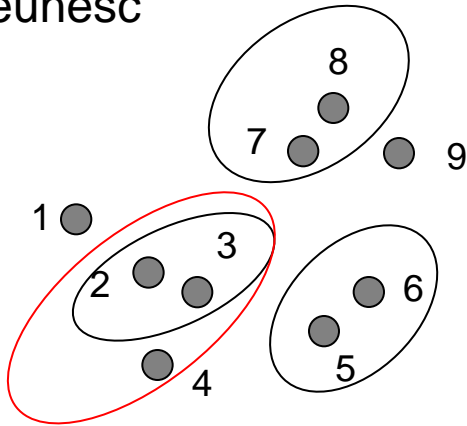


	1	2	3	4	5	6	7	8	9
1	0	2	3	4	7	8	6	8	10
2	2	0	1	2	4	6	7	8	9
3	3	1	0	2	3	5	6	8	9
4	4	2	2	0	3	6	9	10	11
5	7	4	3	3	0	1	4	6	5
6	8	6	5	6	1	0	3	4	3
7	6	7	6	9	4	3	0	1	2
8	8	8	8	10	6	4	1	0	2
9	10	9	9	11	5	3	3	2	0

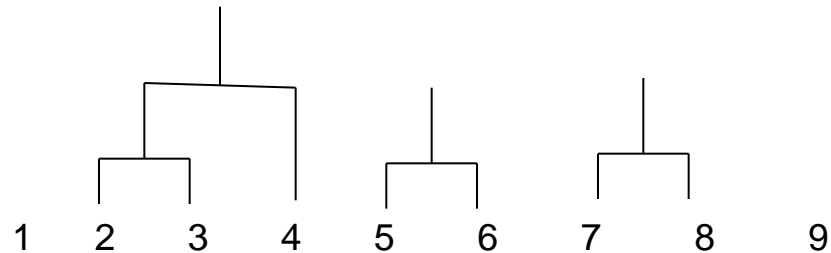


Metoda aglomerativă

Idee: se identifică la fiecare etapă care sunt cele mai similare clusteruri și se reunesc

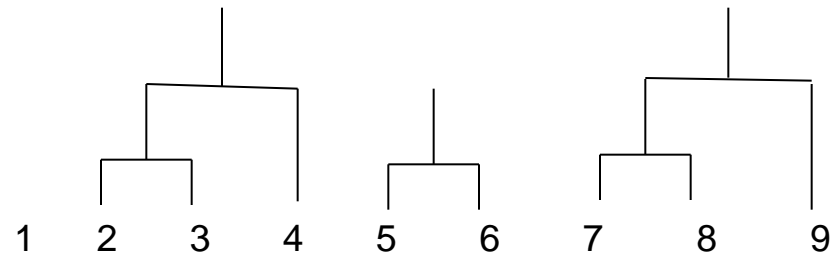
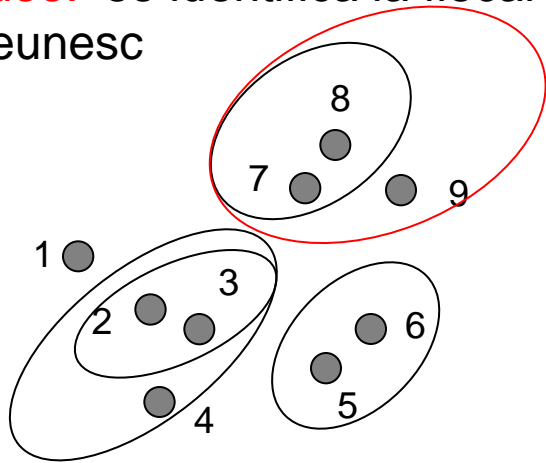


	1	2	3	4	5	6	7	8	9
1	0	2	3	4	7	8	6	8	10
2	2	0	1	2	4	6	7	8	9
3	3	1	0	2	3	5	6	8	9
4	4	2	2	0	3	6	9	10	11
5	7	4	3	3	0	1	4	6	5
6	8	6	5	6	1	0	3	4	3
7	6	7	6	9	4	3	0	1	2
8	8	8	8	10	6	4	1	0	2
9	10	9	9	11	5	3	3	2	0



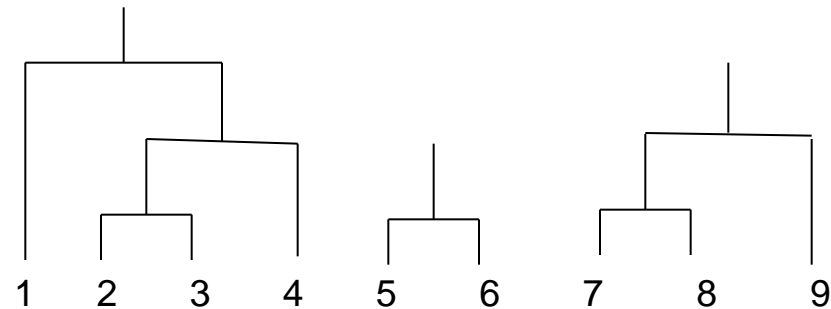
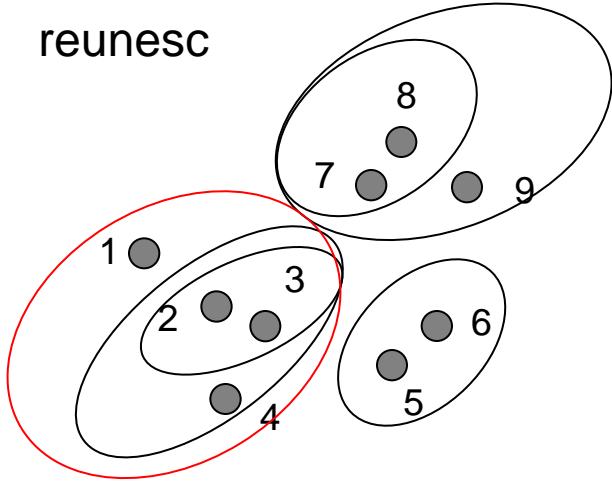
Metoda aglomerativă

Idee: se identifică la fiecare etapă care sunt cele mai similare clusteruri și se reunesc



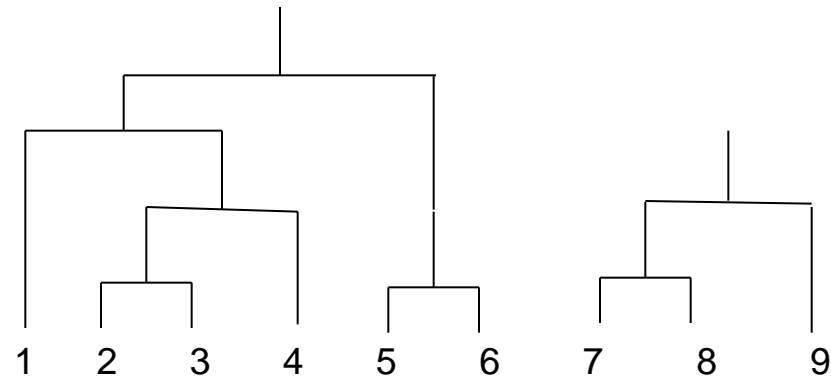
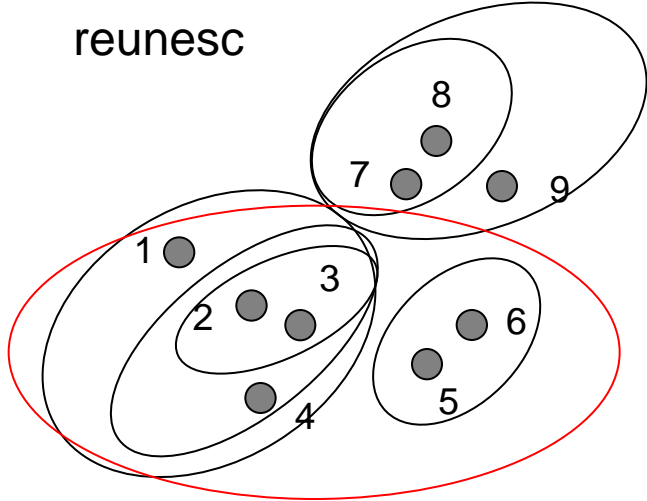
Metoda aglomerativă

Idee: se identifică la fiecare etapă care sunt cele mai similare clusteruri și se reunesc



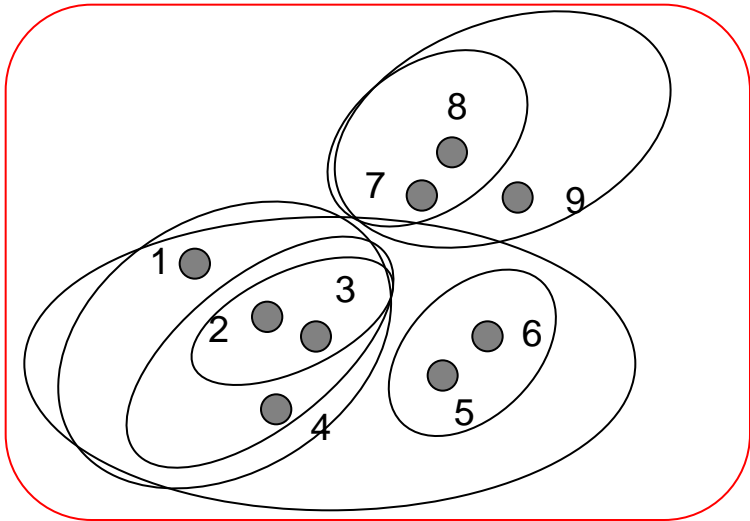
Metoda aglomerativă

Idee: se identifică la fiecare etapă care sunt cele mai similare clusteruri și se reunesc

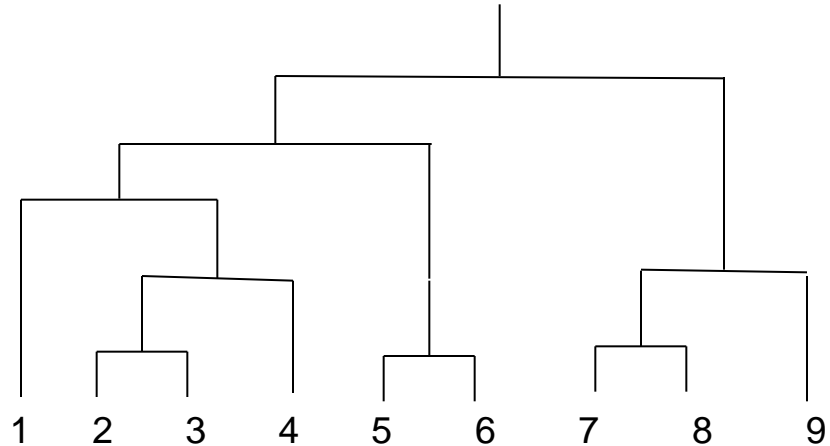


Metoda aglomerativă

Idee: se identifică la fiecare etapă care sunt cele mai similare clusteruri și se reunesc



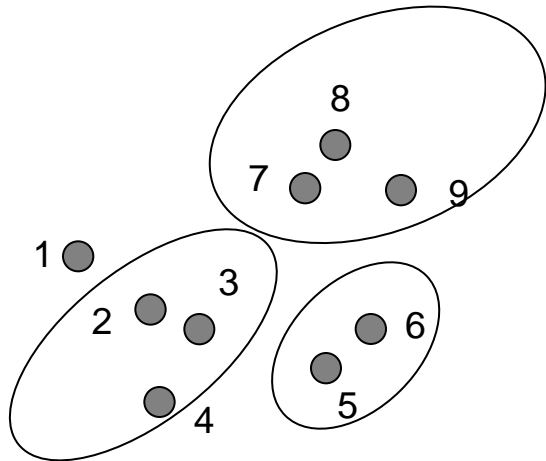
- Dendrograma rezultată



- **Reprezentarea unei dendrograme:** ca set de triplete ordonate (nivel, nr de clusteruri, clusteruri)

$\{(0,9,\{\{1\},\{2\},\dots,\{9\}\}) , (1,6,\{\{1\},\{2,3\},\{4\},\{5,6\},\{7,8\},\{9\}\}) ,$
 $(2,4,\{\{1\},\{2,3,4\},\{5,6\},\{7,8,9\}\}) , (3,3,\{\{1,2,3,4\},\{\{5,6\},\{7,8,9\}\}) ,$
 $(4,2,\{\{1,2,3,4,5,6\},\{7,8,9\}) , (5,1,\{\{1,2,3,4,5,6,7,8,9\}\})\}$

Metoda aglomerativă



Partiție:

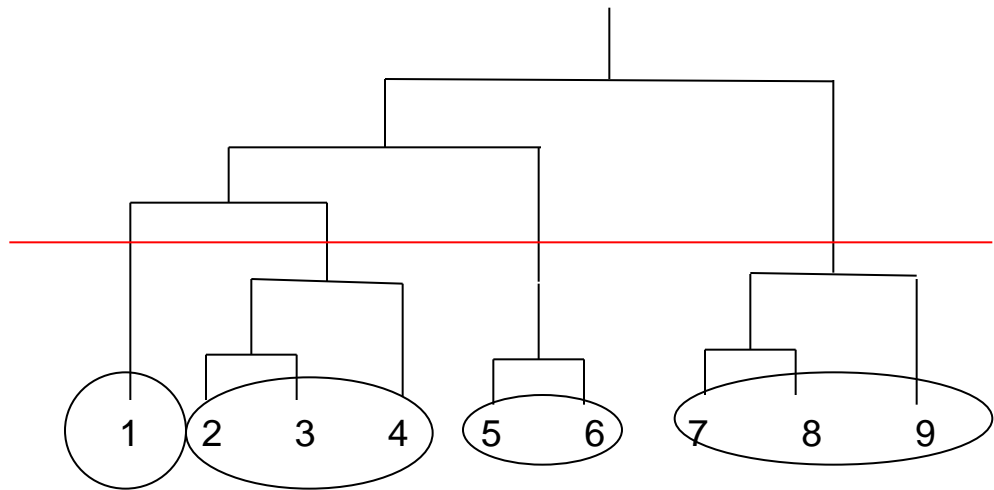
$C1=\{1\}$

$C2=\{2,3,4\}$

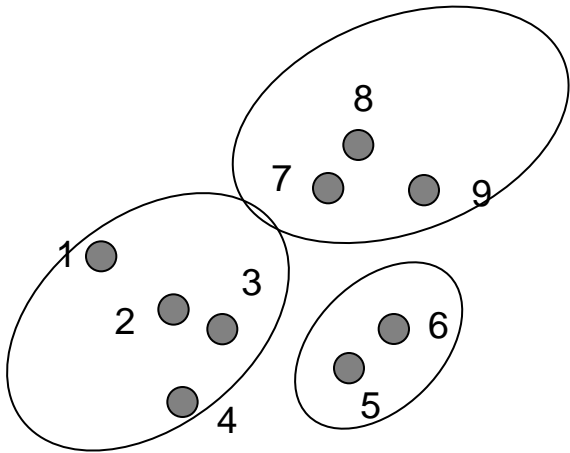
$C3=\{5,6\}$

$C4=\{7,8,9\}$

- Pentru a obține o partiție dendrograma trebuie secționată



Metoda aglomerativă



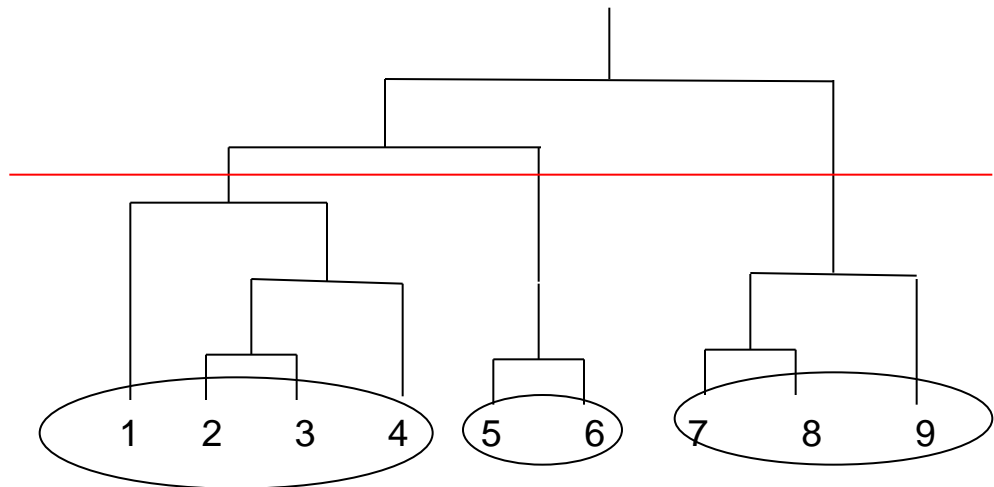
Partition:

$C1=\{1,2,3,4\}$

$C2=\{5,6\}$

$C3=\{7,8,9\}$

- Schimbând nivelul de secționare se obține o altă partiție

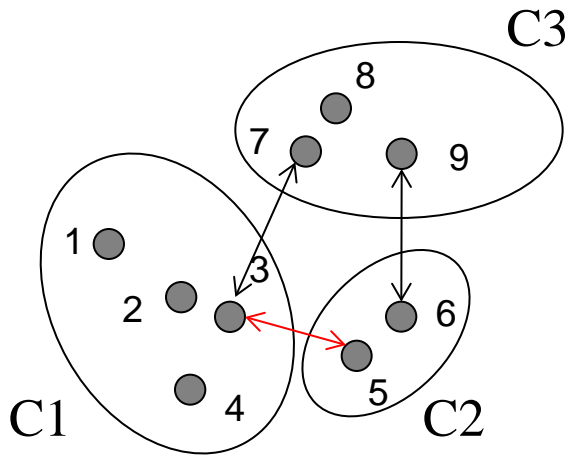


Metoda aglomerativă

Problema: care e criteriul de selecție a clusterelor care se reunesc?

Răspuns: se folosește o măsură de disimilaritate între cluster; sunt mai multe moduri de a calcula această măsură:

- **Single-linkage:** cea mai mică disimilaritate (distanță) între datele aparținând unor cluster diferite



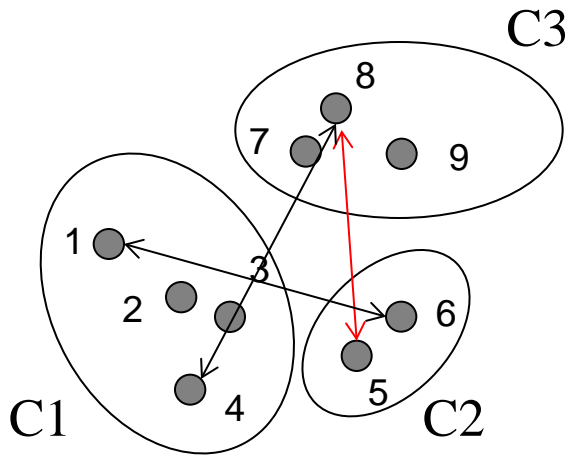
$$D_{SL}(C_i, C_j) = \min_{x \in C_i, y \in C_j} d(x, y)$$

Metoda aglomerativă

Problema: care e criteriul de selecție a clusterelor care se reunesc?

Răspuns: se folosește o măsură de disimilaritate între cluster; sunt mai multe moduri de a calcula această măsură

- **Complete-linkage:** cea mai **mare** disimilaritate (distanță) între datele aparținând unor cluster diferite



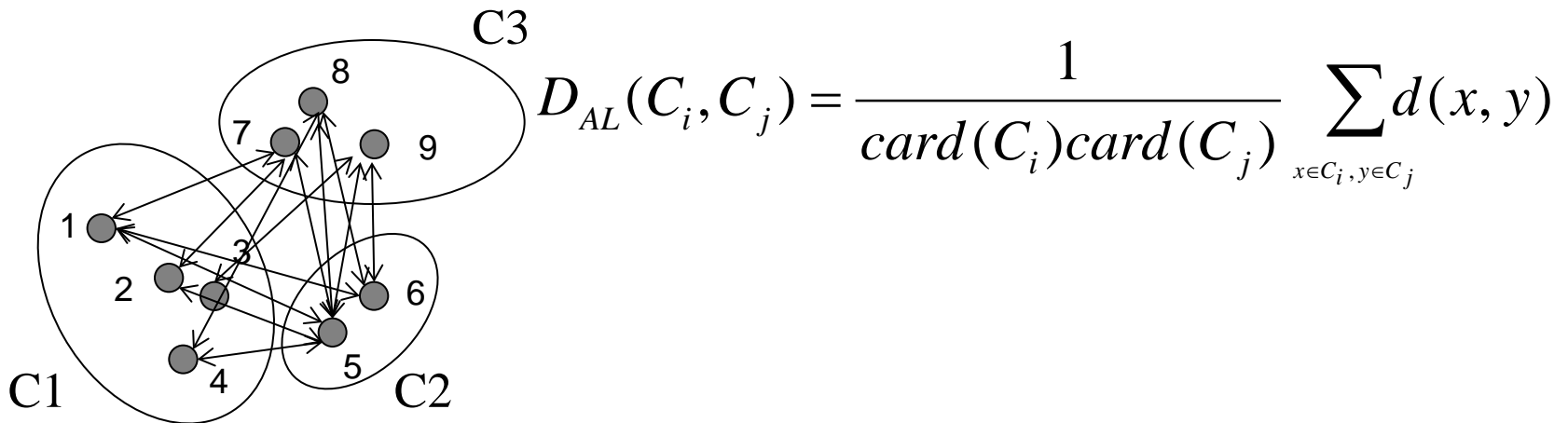
$$D_{CL}(C_i, C_j) = \max_{x \in C_i, y \in C_j} d(x, y)$$

Metoda aglomerativă

Problema: care e criteriul de selecție a clusterelor care se reunesc?

Răspuns: se folosește o măsură de disimilaritate între cluster; sunt mai multe moduri de a calcula această măsură

- **Average-linkage:** media distanțelor dintre datele aparținând unor cluster diferite

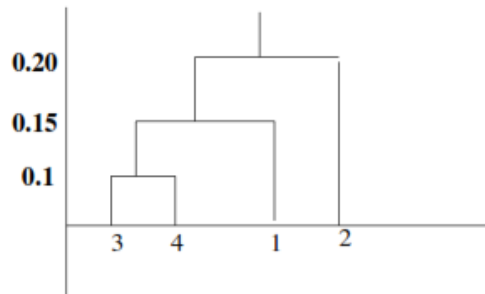


Metoda aglomerativă

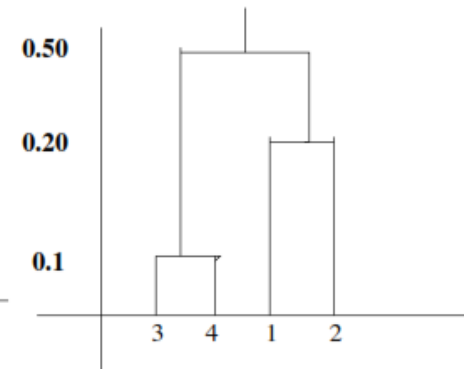
Măsura de disimilaritate folosită influențează rezultatul grupării:

	1	2	3	4
1	0.0	0.20	0.15	0.30
2	0.20	0.0	0.40	0.50
3	0.15	0.40	0.0	0.10
4	0.30	0.50	0.10	0.0

(a) Dissimilarity Matrix



(b) Single Link



(c) Complete Link

[Data Clustering: Algorithms and Applications, 2014]

Metoda aglomerativă

Algoritm

Input : set de date cu N instanțe

$X=\{x_1, x_2, \dots, x_N\}$ + matrice disimilaritate D

Output: dendrograma (set de triplete)

agglomerative(X,D)

level=0; k=N

$C=\{\{x_1\}, \{x_2\}, \dots, \{x_N\}\}$; $DE=\{(level, k, C)\}$

repeat

 oldk=k

 level=level+1

 (k,C)=mergeClusters(k,C,D)

 D=recompute the dissimilarity matrix using
 single/complete/average linkage

 DE=union (DE, (level,k,C))

until k=1

Obs

- Funcția **mergeClusters** identifică cele mai apropiate clustere și le reunește
- Algoritmul are complexitate pătratică în raport cu numărul de date din set ($O(N^2)$)
- Este sensibil la erorile din date

Metoda divizivă

Structura generică

Input : set de date cu N instanțe $X=\{x_1,x_2,\dots,x_N\}$

Output: dendrograma (tree) T

divisive(X,D)

Initialize the tree T with a root node containing the entire data set

Repeat

select a leaf node L from T (based on a specific criterion)

use a **flat clustering algorithm** to split L into L_1,L_2,\dots,L_k

Add L_1,L_2,\dots,L_k as children of L in T

until <a stopping criterion>

Obs: algoritmul partițional poate fi kMeans; un caz particular este **bisecting kMeans** care se bazează pe partiționarea unui cluster în două alte cluster (aplicând kMeans pt $k=2$)

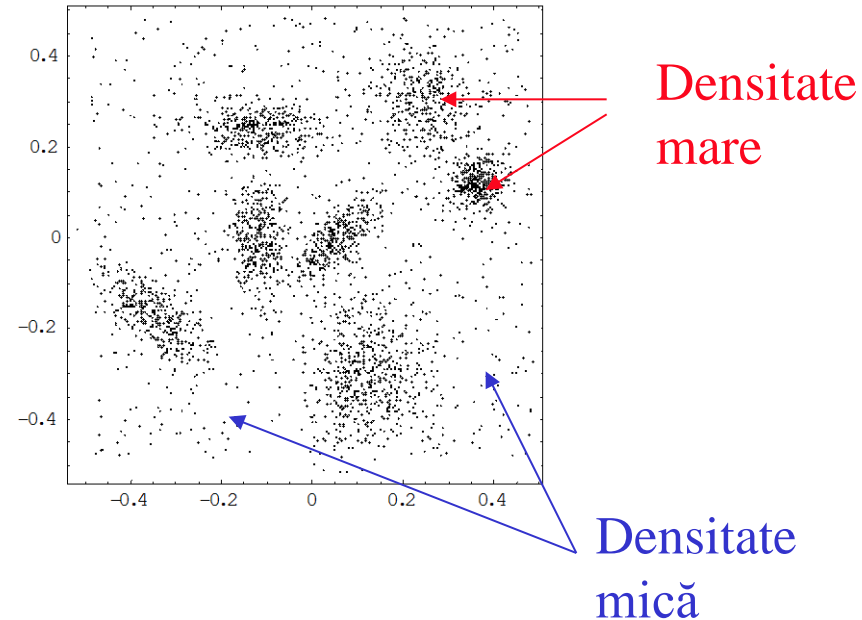
Bisecting Kmeans

- Varianta de algoritm de bisectie bazat pe Kmeans

```
1: Initialize the list of clusters to contain the cluster containing all points.
2: repeat
3:   Select a cluster from the list of clusters
4:   for  $i = 1$  to number_of_iterations do
5:     Bisect the selected cluster using basic K-means
6:   end for
7:   Add the two clusters from the bisection with the lowest SSE to the list of clusters.
8: until Until the list of clusters contains  $K$  clusters
```

Metode bazate pe densitate

- **Cluster** = grup dens de date similare separate de regiuni cu densitate mai mică de date
- **Idee de bază**: estimarea densității locale a datelor
 - se determină numărul de date din vecinătatea punctului analizat (**DBSCAN**)
 - se utilizează funcții de influență pt estimarea densității (**DENCLUE**)
- **Problema principală**:
 - Cum se estimează densitatea?



DBSCAN

DBSCAN [M.Ester, H Kriegel et al, 1996] este un algoritm de grupare bazat pe următoarele elemente:

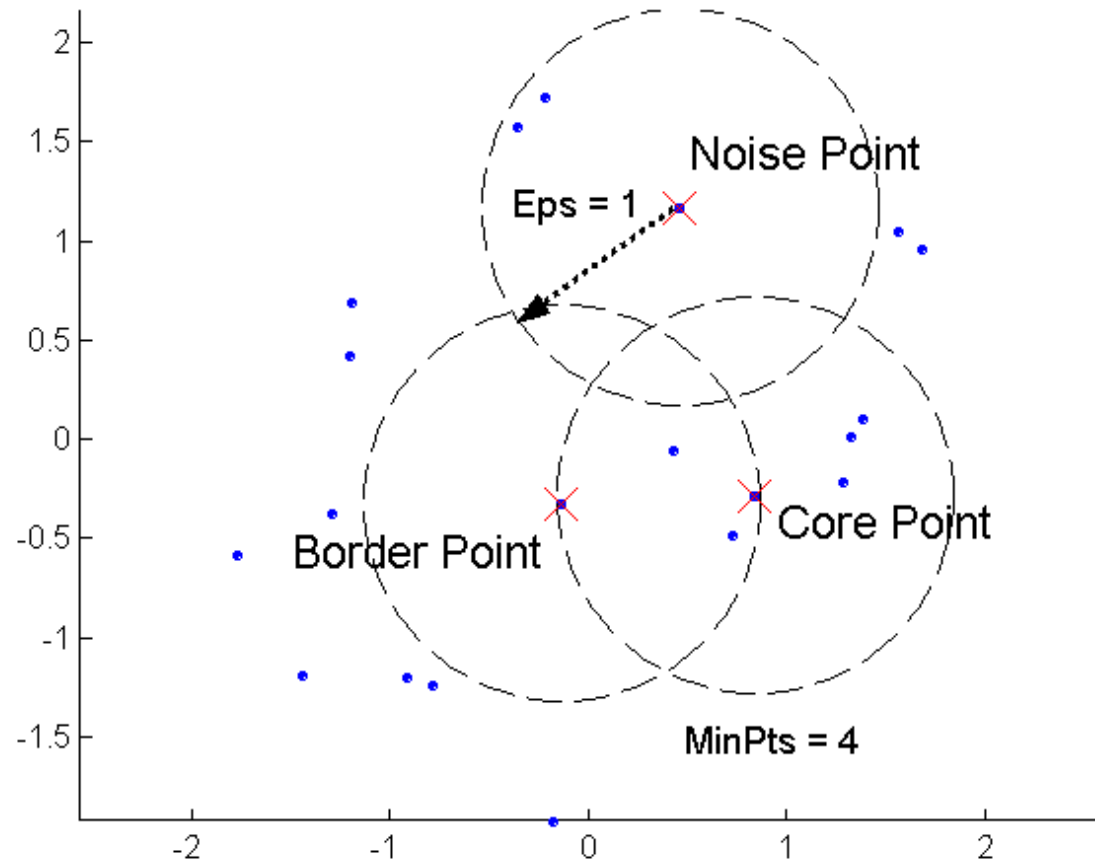
- Densitatea estimată într-un punct = numărul de puncte aflate în vecinătatea definită de o anumită rază (**Eps**)
- Un punct este considerat **punct nucleu (core point)** dacă numărul de puncte din vecinătatea sa depășește un prag(**MinPts**); acestea sunt puncte considerate a fi în interiorul clusterului
- Un **punct frontieră (border point)** are un număr de vecini mai mic decât MinPts dar este în vecinătatea unui punct nucleu; Două puncte sunt considerate **conectate** dacă unul este în vecinătatea celuilalt

DBSCAN

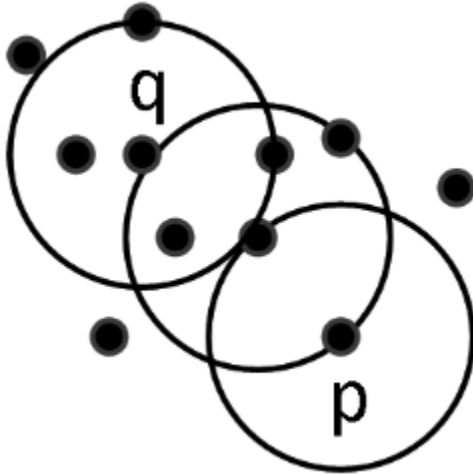
DBSCAN [M.Ester, H Kriegel et al, 1996] este un algoritm de grupare bazat pe următoarele elemente:

- Un punct q este accesibil direct (**directly density reachable**) dintr-un punct nucleu p dacă e în vecinătatea lui p ; accesibilitatea în sens general este definită ca fiind închiderea tranzitivă a relației de accesibilitate directă (există un lanț de **puncte nucleu** ce începe cu p și se termină cu q cu proprietatea că fiecare e direct accesibil dintr-un punct anterior)
- Un punct de tip zgomot (**noise point**) este orice punct care nu e nici nucleu nici frontieră

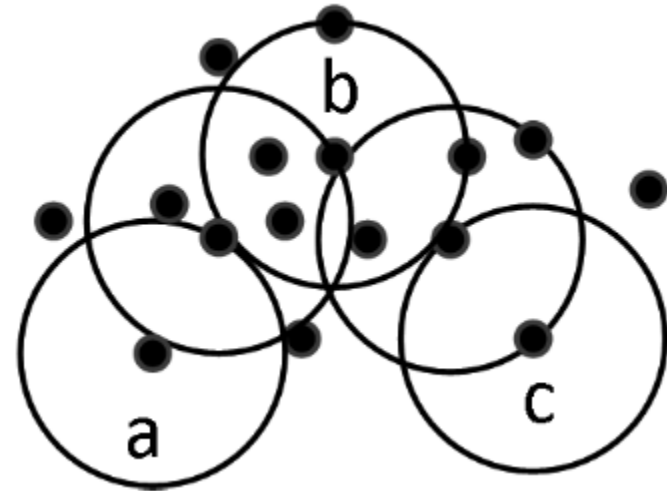
DBSCAN



DBSCAN



p este accesibil din q



a este accesibil din b

c este accesibil din b

=> a și c sunt conectate

Obs:

- Două pcte, a și b, dacă există un punct c astfel încât c este accesibil atât din a cât și din b
- Două puncte conectate ar trebui să aparțină aceluiași cluster => un cluster definit pe baza densității este un set maximal de date conectate

DBSCAN

```
DBSCAN (SetOfPoints, Eps, MinPts)    // SetOfPoints is UNCLASSIFIED
ClusterId = nextId(NOISE);
FOR i = 1, SetOfPoints.size DO
    Point = SetOfPoints.get(i);
    IF Point.ClusterId == UNCLASSIFIED THEN
        IF ExpandCluster(SetOfPoints, Point, ClusterId, Eps, MinPts)
            THEN ClusterId = nextId(ClusterId)
        END IF
    END IF
END FOR
END;                                // DBSCAN
```

[M.Ester, H Kriegel et al, A Density-Based Algorithm for Discovering
Clusters in Large Spatial Databases with Noise, 1996]

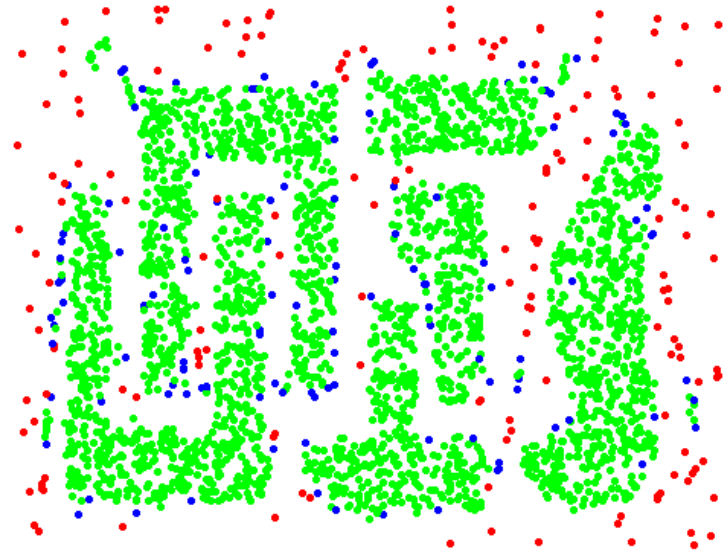
DBSCAN

```
ExpandCluster(SetOfPoints, Point, ClId, Eps, MinPts) : Boolean;
  seeds=SetOfPoints.regionQuery(Point,Eps);    // find neighbouring points
  IF seeds.size < MinPts THEN SetOfPoint.changeClId(Point,NOISE); RETURN False;
ELSE                                           // all points in seeds are density-reachable from Point
  SetOfPoints.changeClIds(seeds,ClId); seeds.delete(Point);
  WHILE seeds != Empty DO
    currentP = seeds.first();  result = SetOfPoints.regionQuery(currentP, Eps);
    IF result.size >= MinPts THEN
      FOR i = 1, result.size DO
        resultP = result.get(i);           // take next point
        IF resultP.ClId IN {UNCLASSIFIED, NOISE} THEN
          IF resultP.ClId == UNCLASSIFIED THEN seeds.append(resultP); END IF;
          SetOfPoints.changeClId(resultP,ClId);
        END IF;                          // UNCLASSIFIED or NOISE
      END FOR; END IF;                     // result.size >= MinPts
    seeds.delete(currentP); END WHILE;    // seeds != Empty
  RETURN True; END IF;  END;
```

DBSCAN



Date (puncte) de
prelucrat



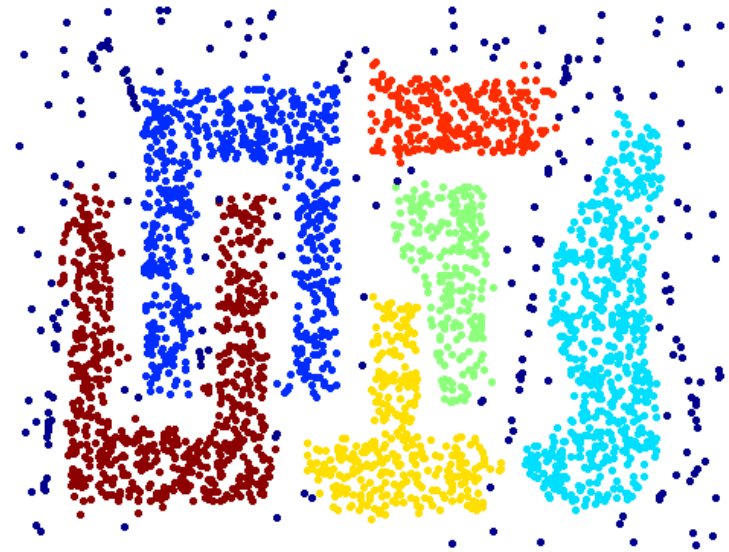
Tipuri de puncte : core,
border și noise

Eps = 10, MinPts = 4

DBSCAN



Date

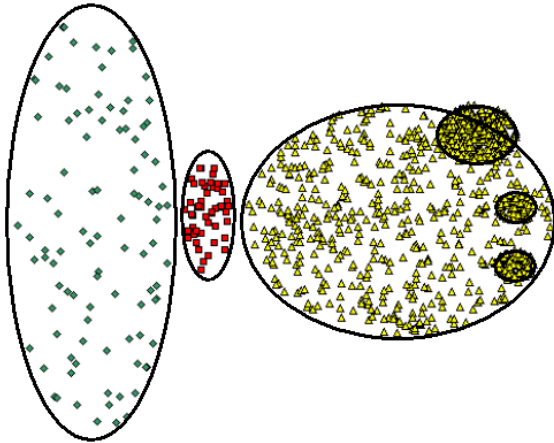


Clustere

Specific DBSCAN:

- Permite identificarea clusterelor de diferite forme
- Robust

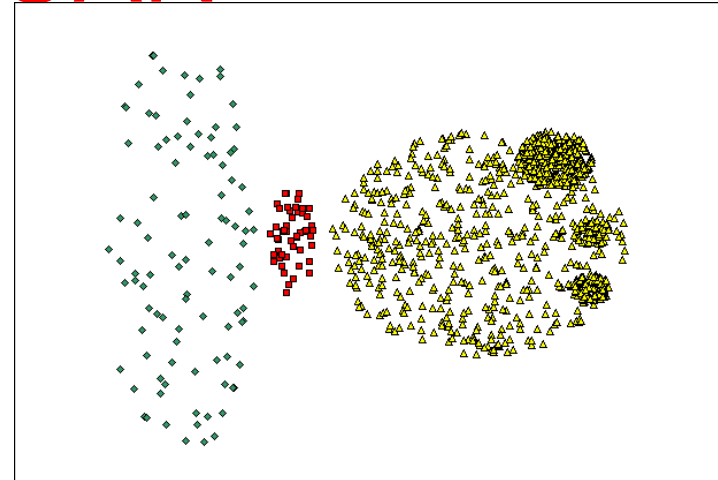
DBSCAN



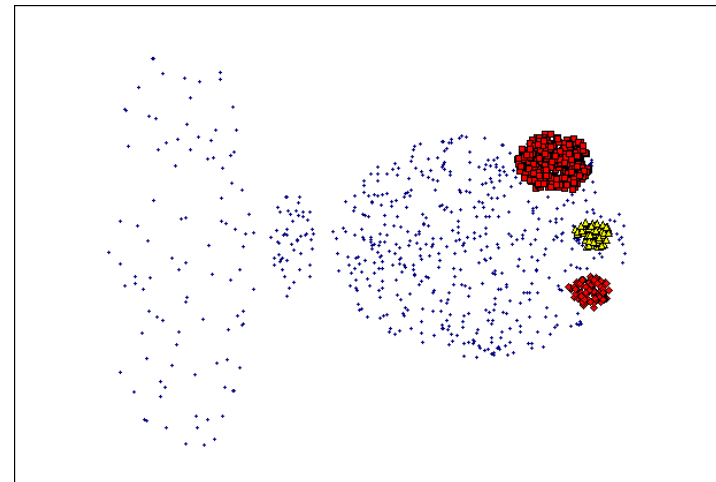
Original Points

Nu este adecvat pt:

- variații în densitatea datelor
- date de dimensiuni mari (cu multe atribute)

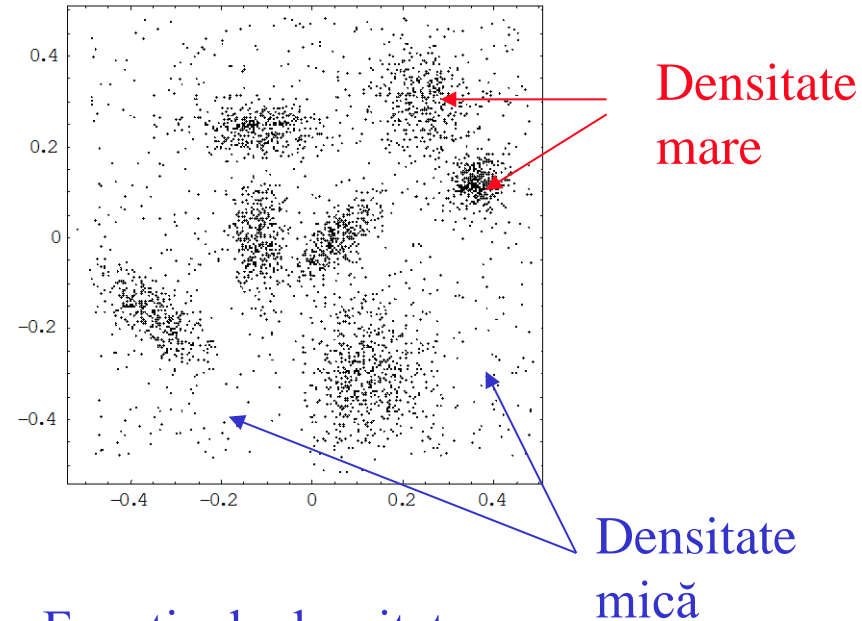


(MinPts=4, Eps=9.75).



DENCLUE

- **Cluster** = grup dens de date similare separate de regiuni cu densitate mai mică de date
- **Idee de bază**: estimarea densității locale a datelor
 - se utilizează funcții de influență pt estimarea densității



Funcție de influență

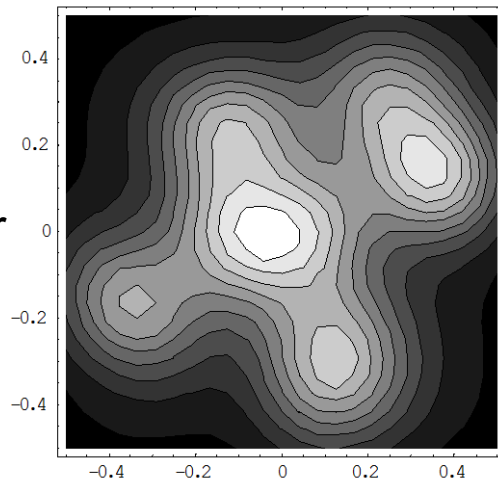
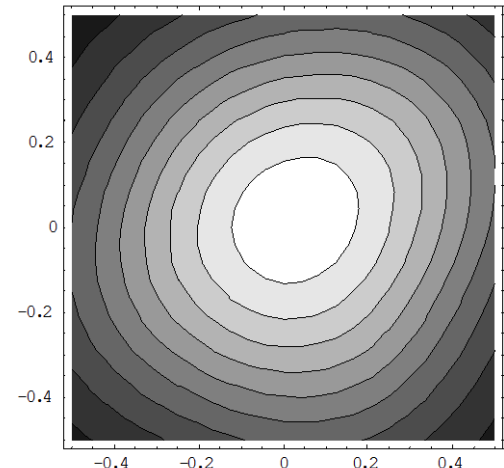
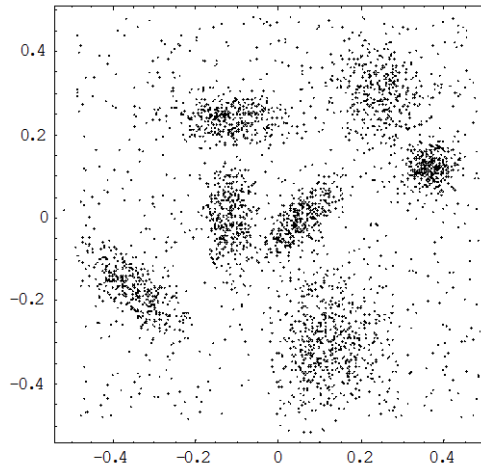
$$I_y(x) = \frac{1}{\sigma^{n/2}} \exp\left(-\frac{\sum_{j=1}^n (x_j - y_j)^2}{2\sigma^2}\right)$$

Funcție de densitate

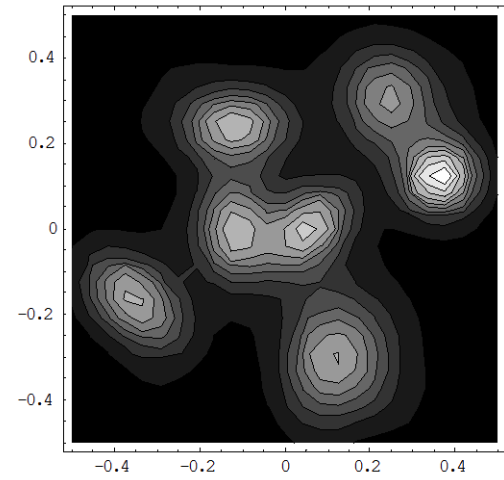
$$f(x) = \frac{1}{N} \sum_{i=1}^N I_{x_i}(x)$$

DENCLUE

- Forma funcției de densitate depinde de valoarea lui σ
- Dacă valoarea lui σ este adecvată, maximele locale ale funcției de densitate corespund reprezentanților clusterelor
- Pt valori mari ale lui σ funcția de densitate are un maxim unic
- Pt valori prea mici ale lui σ maximele locale corespund unor vârfuri izolate și pot fi dificil de detectat



$\sigma=0.1$



$\sigma=0.05$ 47

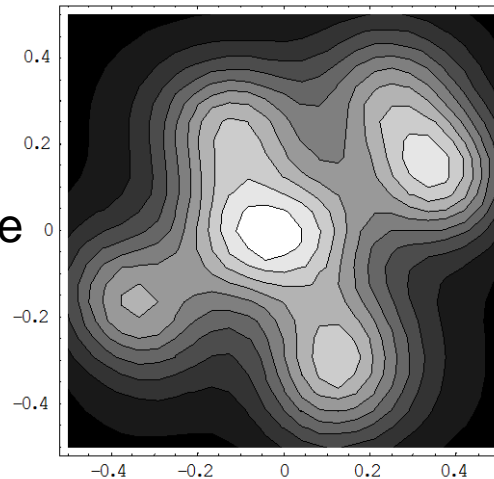
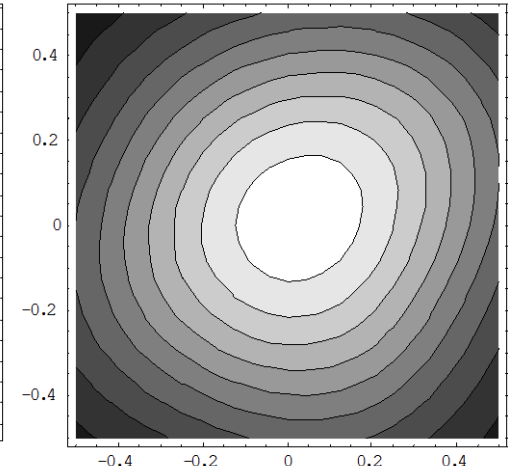
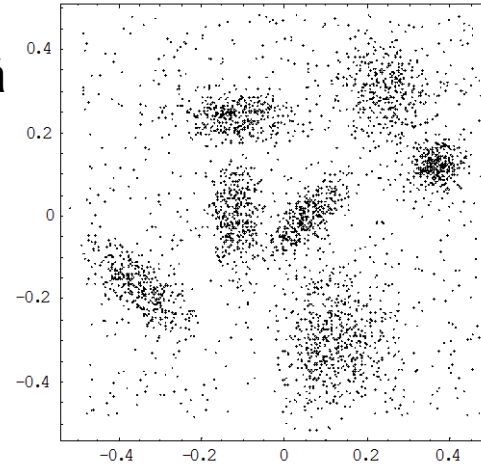
DENCLUE

Ideea algoritmului DENCLUE

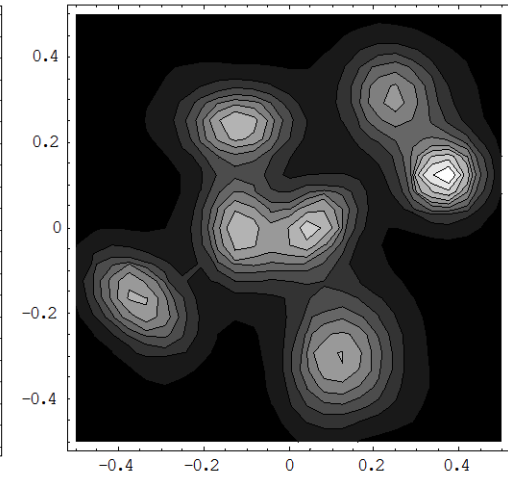
[Hinneburg, Keim – 1998]: se aplică căutare de tip gradient pornind de la punctele din setul de date cu scopul identificării maximelor locale

Variante:

- Fiecare maxim local corespunde unui cluster (clusterelor detectate vor fi sferice sau elipsoidale)
- Un cluster corespunde unui set de maxime locale “învecinate” (se pot identifica cluster de forma arbitrara)



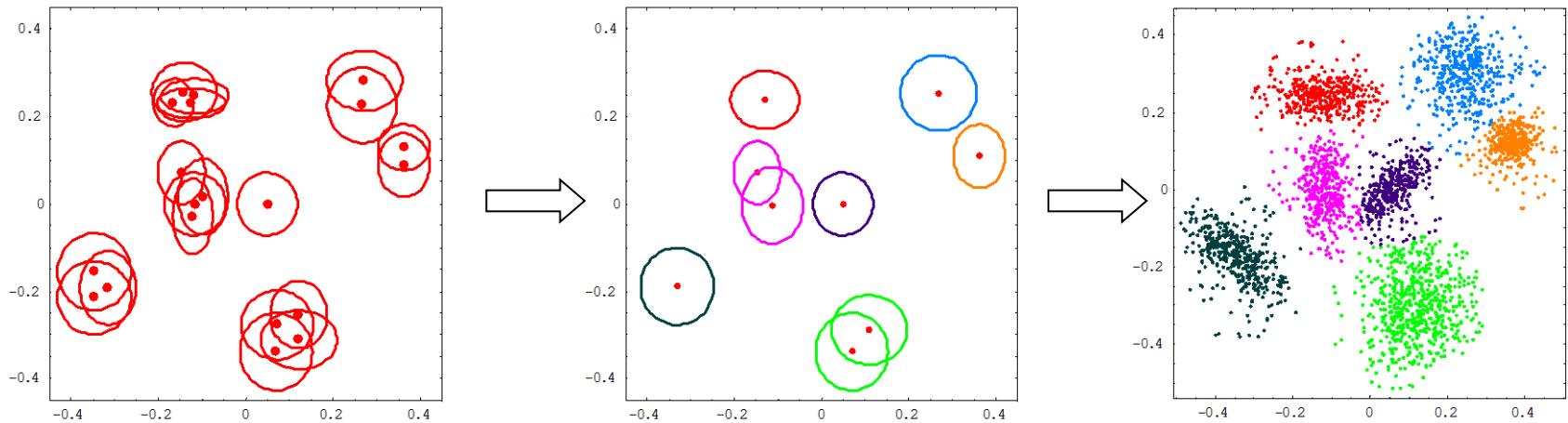
$\sigma=0.1$



$\sigma=0.05$ 48

DENCLUE

Exemple de rezultate



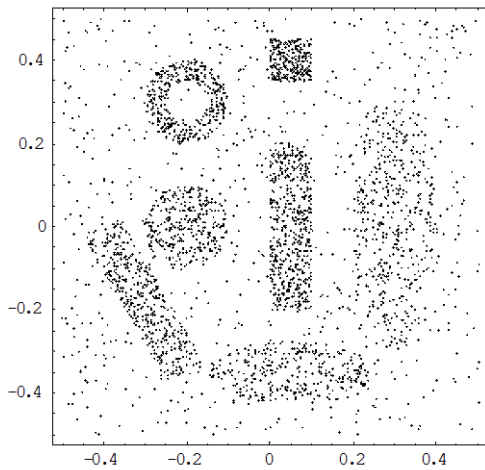
Punctele marcate = puncte de maxim ale funcției de densitate

Regiuni marcate = arii de “influență” (determinate de valorile parametrilor sigma)

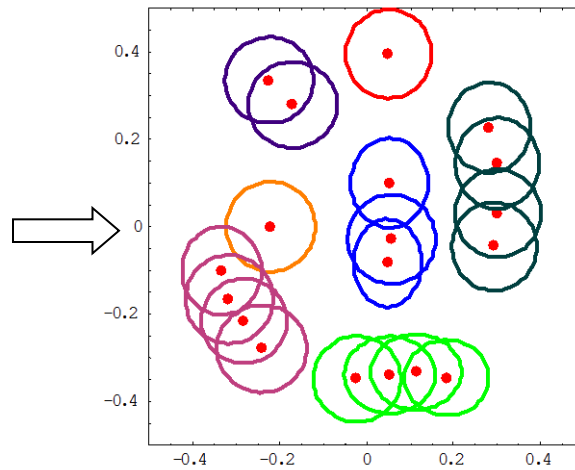
Grupare = distribuirea datelor în clustere se bazează pe valorile asociate funcțiilor de influență

DENCLUE

Exemple de rezultate

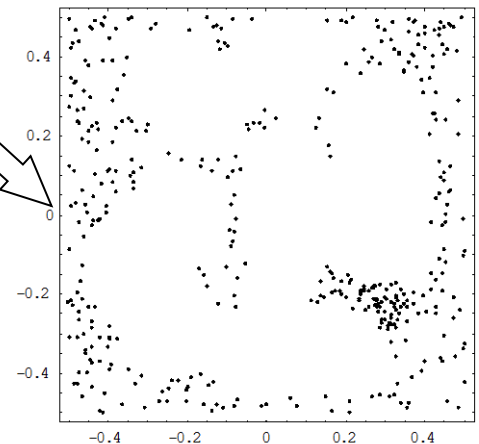
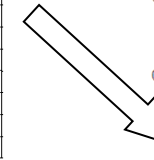
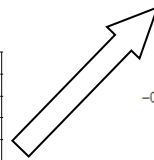
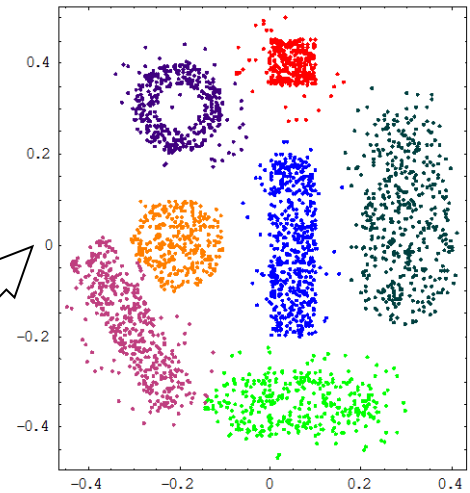


Date inițiale



Descriptori
ai clusterelor

Clustere identificate



Date considerate zgomot 50

Metode probabiliste

Idee de bază:

- Datele sunt generate de un proces stohastic (o **mixtură** de distribuții de probabilitate, fiecare dintre ele corespunzând unui cluster)
- Scopul algoritmului de grupare este de a descoperi modelul probabilist, adică de a identifica distribuțiile de probabilitate

Exemple:

- Algoritmul **Expectation–Maximization (EM)** – se bazează pe următoarele ipoteze:
 - Fiecare dată este generată de o distribuție de probabilitate (tipul de distribuție depinde de natura datelor)
 - În procesul de generare a datelor fiecare dintre distribuțiile de probabilitate este selectată la rândul ei cu o anumită probabilitate

Algoritmul EM

- **Input:** set de date $D=\{x_1, x_2, \dots, x_N\}$, K = număr de clustere
- **Output:** o partiție $P=\{C_1, C_2, \dots, C_K\}$ a setului D

Inițializarea parametrilor modelelor și a probabilităților de selecție

- **(E-Step)** Se determină probabilitatea de asignare a fiecărei date la clustere (folosind valorile curente ale parametrilor)
- **(M-Step)** Se determină parametrii modelului folosind valorile curente ale probabilităților de asignare a datelor la clustere

Obs:

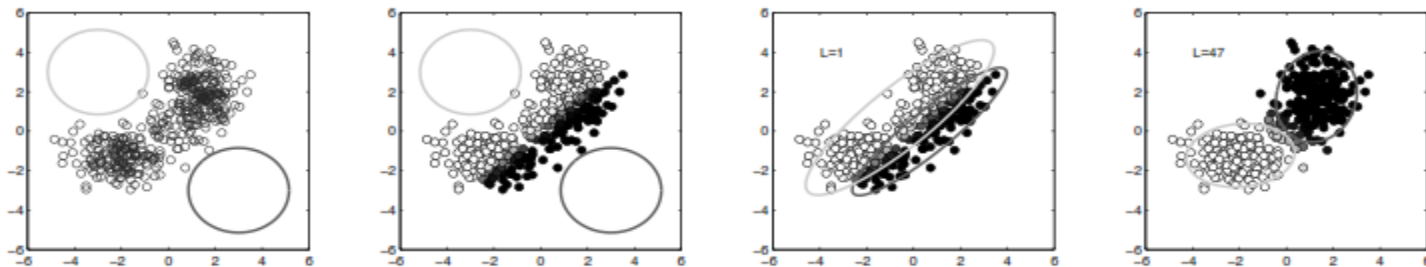
- In cazul datelor numerice se poate folosi distribuția normală multi-dimensională pentru a modela datele aparținând unui cluster (modelul de generare a datelor este o mixtură de gaussiene). Parametrii care se estimează în acest caz sunt **media** și **matricea de covarianță**.

EM algorithm

Algorithm 11 EM for Gaussian Mixtures

Given a set of data points and a Gaussian mixture model, the goal is to maximize the log-likelihood with respect to the parameters.

- 1: Initialize the means μ_k^0 , covariances Σ_k^0 , and mixing probabilities π_k^0 .
 - 2: **E-step**: Calculate the responsibilities $\gamma(z_{nk})$ using the current parameters based on Equation (3.13).
 - 3: **M-step**: Update the parameters using the current responsibilities. Note that we first update the new means using (3.12), then use these new values to calculate the covariances using (3.14), and finally reestimate the mixing probabilities using (3.15).
 - 4: Compute the log-likelihood using (3.10) and check for convergence of the algorithm. If the convergence criterion is not satisfied, then repeat steps 2–4; otherwise, return the final parameters.
-



Data Clustering ^(a) Algorithms and applications ^(b) (ed. CC Aggarwal, CK Reddy, ^(c) ^(d) 2014)

Algoritmul EM

Ecuatii care intervin in algoritmul EM

$$p(\mathbf{x}_n|\Theta) = p(\mathbf{x}_n|\pi, \mu, \Sigma) = \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}_n|\mu_k, \Sigma_k). \quad (3.10)$$

$$\mathcal{N}(\mathbf{x}|\mu, \Sigma) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x} - \mu)^T \Sigma^{-1}(\mathbf{x} - \mu)\right\}$$

$$\mu_k = \frac{\sum_{n=1}^N \gamma(z_{nk}) \mathbf{x}_n}{\sum_{n=1}^N \gamma(z_{nk})} \quad (3.12)$$

$$\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n|\mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n|\mu_j, \Sigma_j)}. \quad (3.13)$$

$$\Sigma_k = \frac{\sum_{n=1}^N \gamma(z_{nk}) (\mathbf{x}_n - \mu_k)(\mathbf{x}_n - \mu_k)^T}{\sum_{n=1}^N \gamma(z_{nk})}. \quad (3.14)$$

$$\pi_k = \frac{\sum_{n=1}^N \gamma(z_{nk})}{N} \quad (3.15)$$

Data Clustering Algorithms and applications (ed. CC Aggarwal, CK Reddy, 2014)

Cursul următor

- Reguli de asociere
- Algoritmul Apriori