Curs 7-8:

Gruparea datelor

Structura

- Gruparea datelor
  - Concepte de bază
  - Evaluarea calităţii grupării
- Algoritmi partiţionali
  - 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 cunoaşte?

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

Ce se doreşte?

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

Care este scopul final?

- Să se poată verifica dacă două date aparţin aceluiaşi cluster
- Să se determine clusterul de care aparţine o dată
- Să se identifice excepţii (date care nu aparţin nici unui cluster)

Scopul grupării (reminder)

Exemple:

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

Gruparea permite:

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

Particularităţi ale grupării

Este un proces nesupervizat:

- Setul de antrenare conţine 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?

Şase clustere

Două clustere

patru clustere

[Kumar, 2004]

Concepte de bază

- Cluster = grup de date care sunt similare
- Matrice de (di)similaritate pt un set de n date = matrice nxn in care pe linia i coloana j se află valoarea similarităţii/disimilarităţii între data i şi 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 aparţine setului de date

- Raza clusterului = media distanţelor dintre datele din cluster şi prototipul acestuia
- Diametrul clusterului = distanţa (disimilaritatea) maximă dintre oricare două date ale clusterului

Tipuri de clustering

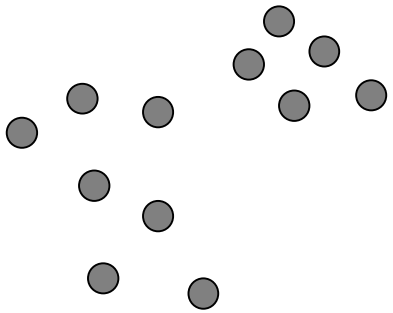
- Crisp vs fuzzy clustering
  - Crisp clustering = fiecare dată aparţine unui singur cluster
  - Fuzzy clustering = o dată poate aparţine mai multor clustere (grad de apartenenţă pentru fiecare cluster)

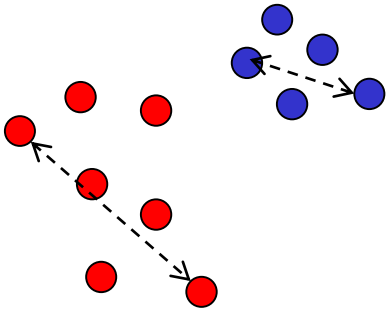
- Flat vs hierarchical clustering
  - Flat (partitional) clustering = rezultatul este un set de clustere (o partiţie)
  - Hierarchical clustering = rezultatul este o ierarhie de partiţii

- Variante de algoritmi
  - Algoritmi partiţionali (ex: kMeans, Fuzzy cMeans)
  - Algoritmi hierarhici (alg. aglomerativi, alg. divizivi)
  - Algoritmi bazaţi pe densitate (ex: DBSCAN)
  - Algoritmi bazați pe modele probabiliste (ex: EM = Expectation Maximization)

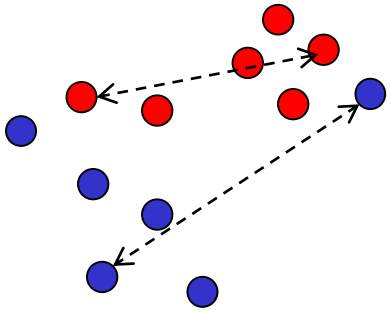
Măsuri de calitate

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





Calitate bună



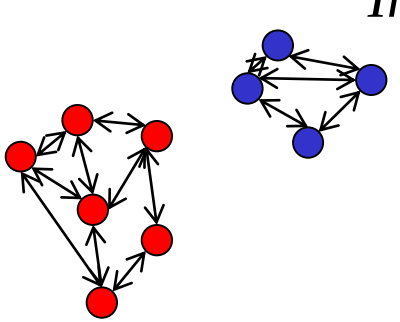
Calitate slabă

Măsuri de calitate

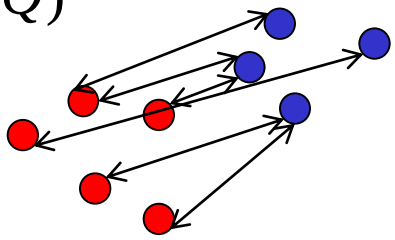
- Intra-cluster to inter-cluster distance ratio = Intra/Inter (valori mici corespund unei calităţi mai bune)
- Fie P setul de date ce aparţin aceluiaşi cluster şi Q=DxD-P (restul perechilor: o dată din pereche aparţine unui cluster iar cealaltă unui alt cluster)

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

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



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



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

Măsuri de calitate

- Silhouette coefficient

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

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

$Davg_i^{in}$ = media distantelor dintre x_i si celelalte date din clusterul lui x_i

$Davg_i^{j}$ = media distantelor dintre x_i si datele din clusterul j $(j \neq i)$

$D\min_i^{out} = \min_j Davg_i^{j}$

Obs:

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

kMeans

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

kMeans (D,k)

initialize the centroids $c_1, c_2, \ldots, c_K$ (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) – distanţele dintre date şi centroizii corespunzători

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

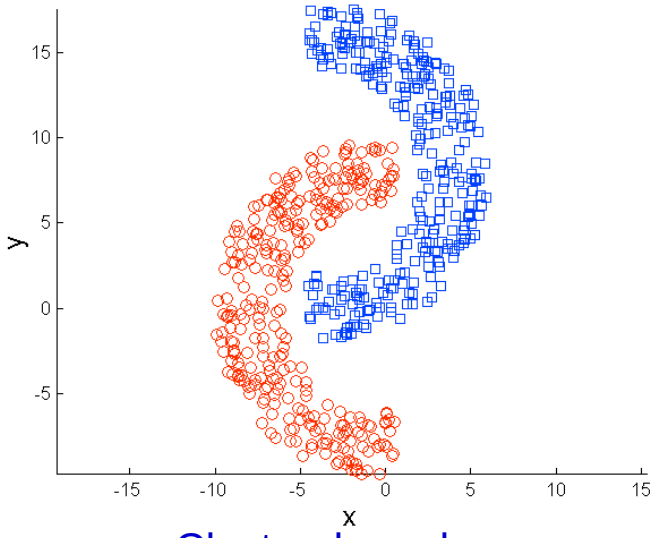
(în cazul distanţei euclidiene)

  - Complexitate: O(n*N*K*iteraţii) (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

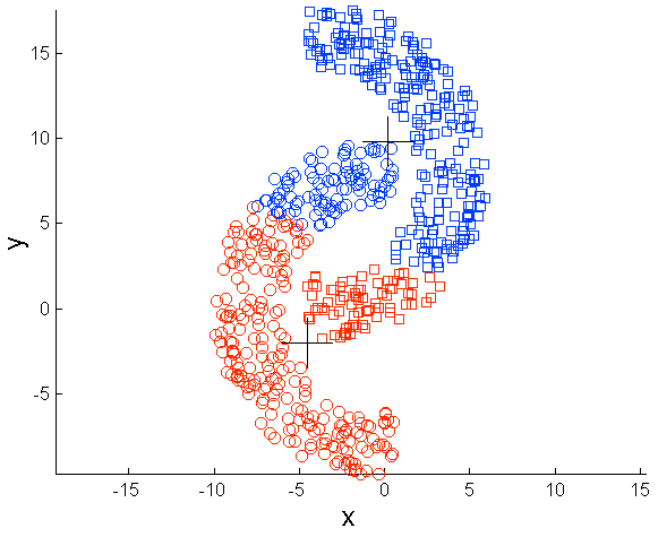
kMeans

Limite:

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



Clusterele reale



Rezultatul Kmeans

[from slides Kumar, 2004]

kMeans

Limite:

- Rezultatul poate fi ușor influențat de erorile din date sau de datele atipice
  - Soluţie: 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 cunoaşterea apriori a numărului de clustere

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

ISODATA

Idei principlale ale alg ISODATA

- Dacă dimensiunea unui cluster este mai mică decât Nmin (un parametru care specifică numărul minim de date dintr-un cluster) atunci clusterul ar trebui reunit cu cel mai apropiat alt cluster

- Dacă distanţa dintre două clustere (de exemplu the distanţa dintre prototipurile clusterelor) este mai mică decât Dmin (un parametru care specifică distanța minimă dintre clustere) atunci clusterele ar trebui reunite
- Dacă varianţa unui cluster este mai mare decât Vmax şi numărul de date conţinute este mai mare decât 2*Nmin atunci clusterul poate fi divizat în două alte clustere:
  - Identifică atributul j pt care varianţa este maxmă
  - Din prototipul c_k sunt construite două alte prototipuri c' şi 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 aparţine unui singur cluster ci poate aparţine mai multor clustere (cu un anumit grad de apartenenţă pentru fiecare cluster)

- Rezultatul unei grupări fuzzy este o matrice M de dimensiune NxK

(N= nr date, K= nr clustere);

M(i,j) = o valoare din [0,1] care corespunde gradului de apartenenţă a datei i la clusterul j

Obs: Fuzzy cMeans poate fi utilizată în segmentarea imaginilor

Fuzzy cMeans

Algoritm

- Initialize the membership matrix (M)
- **Repeat**
  - Compute the centroids(c_k, k=1,…K)
  - Update the membership values (m_{ij}, i=1,…,N, j=1,…,K

  until <no significant changes in the membership function>

Obs: fiecare dată este asignată la clusterul ce corespunde celui mai mare grad de apartenenţă

Calculul centroizilor

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

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

Calculul gradului de apartenenţă

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

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

Algoritmi ierarhici

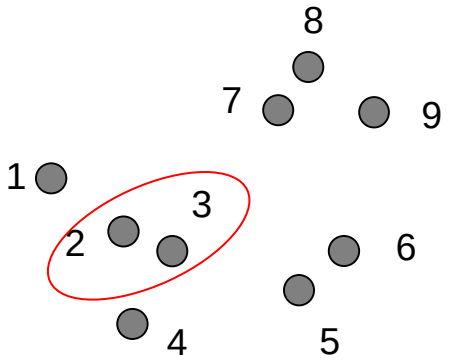
Obs: una dintre principalele limite ale algoritmilor partiționali e faptul că necesită cunoaşterea numărului de clustere.

Altă abordare: se construieşte o ierarhie de partiţii – conduce la o structură arborescentă numită dendrogramă

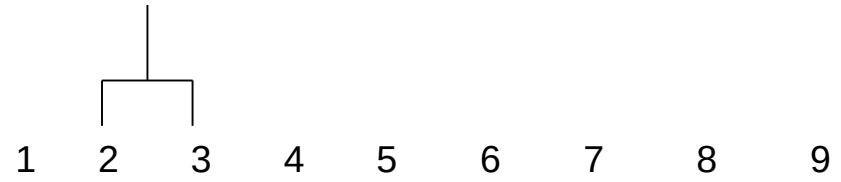
- In varianta bottom-up (metoda aglomerativă)
  - Se porneşte cu o partiţie de clustere ce conţin fiecare câte o singură dată (reprezintă frunze în arbore)
  - Se reunesc clusterele care sunt similare între ele – procesul de reunire se repetă până se ajunge la un singur cluster (reprezintă rădăcina arborelui)
- In varianta top-down (metoda divizivă)
  - Se porneşte cu o partiţie ce conţine un singur cluster (cu toate datele)
  - Se partiţionează clusterii mari aplicând o tehnica partiţională (ex: kMeans) – procesul continuă până când se ajunge la clustere ce conţin câte o singură dată.

Metoda aglomerativă

Idee: se identifică la fiecare etapă care sunt cele mai similare clustere şi se reunesc

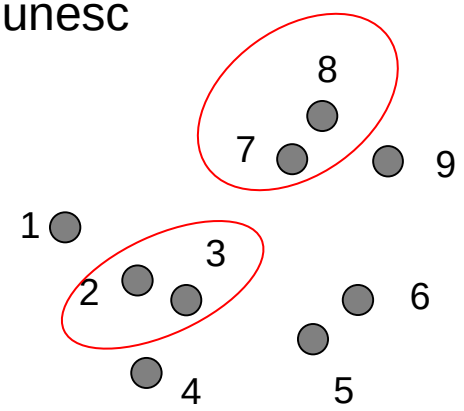


	1	2	3	4	5	6	7	8	9
1	0	2	3	4	7	8	6	8	10
2	2	0	1	2	4	6	7	8	9
3	3	1	0	2	3	5	6	8	9
4	4	2	2	0	3	6	9	10	11
5	7	4	3	3	0	1	4	6	5
6	8	6	5	6	1	0	3	4	3
7	6	7	6	9	4	3	0	1	2
8	8	8	8	10	6	4	1	0	2
9	10	9	9	11	5	3	3	2	0

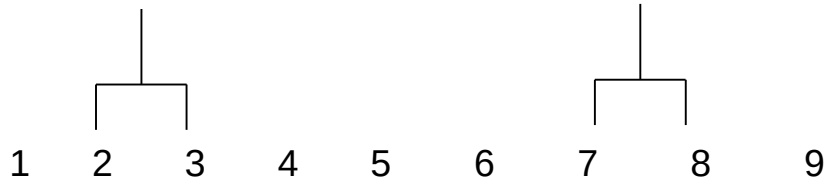


Metoda aglomerativă

Idee: se identifică la fiecare etapă care sunt cele mai similare clustere şi se reunesc

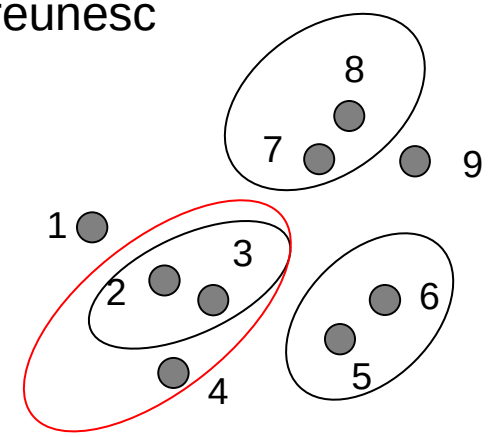


	1	2	3	4	5	6	7	8	9
1	0	2	3	4	7	8	6	8	10
2	2	0	1	2	4	6	7	8	9
3	3	1	0	2	3	5	6	8	9
4	4	2	2	0	3	6	9	10	11
5	7	4	3	3	0	1	4	6	5
6	8	6	5	6	1	0	3	4	3
7	6	7	6	9	4	3	0	1	2
8	8	8	8	10	6	4	1	0	2
9	10	9	9	11	5	3	3	2	0

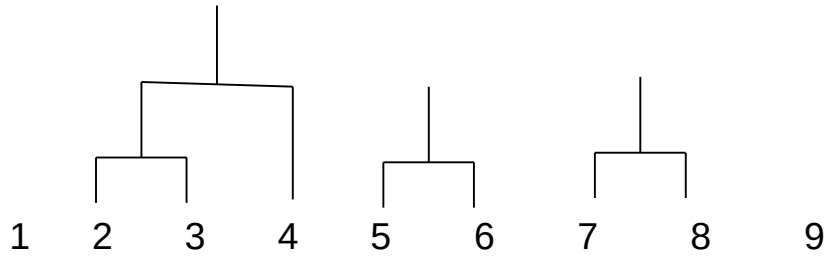


Metoda aglomerativă

Idee: se identifică la fiecare etapă care sunt cele mai similare clustere şi se reunesc

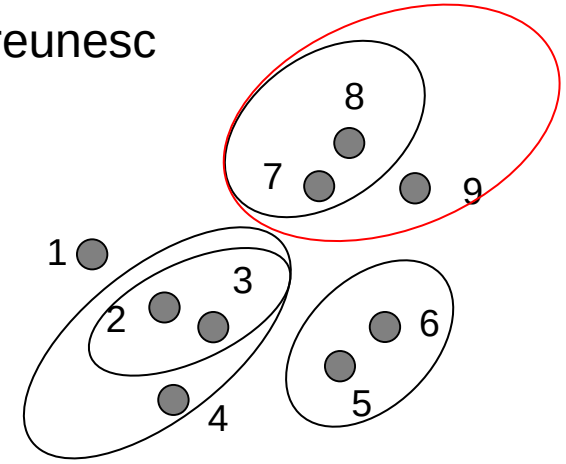


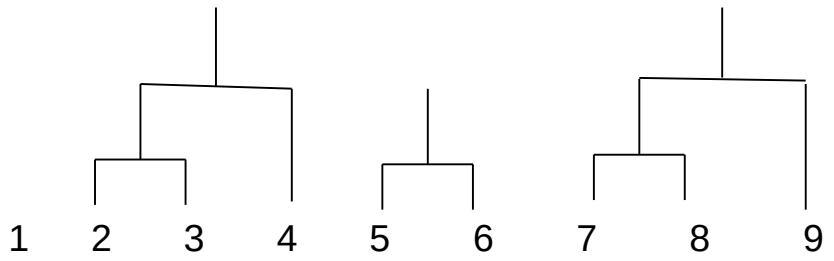
	1	2	3	4	5	6	7	8	9
1	0	2	3	4	7	8	6	8	10
2	2	0	1	2	4	6	7	8	9
3	3	1	0	2	3	5	6	8	9
4	4	2	2	0	3	6	9	10	11
5	7	4	3	3	0	1	4	6	5
6	8	6	5	6	1	0	3	4	3
7	6	7	6	9	4	3	0	1	2
8	8	8	8	10	6	4	1	0	2
9	10	9	9	11	5	3	3	2	0



Metoda aglomerativă

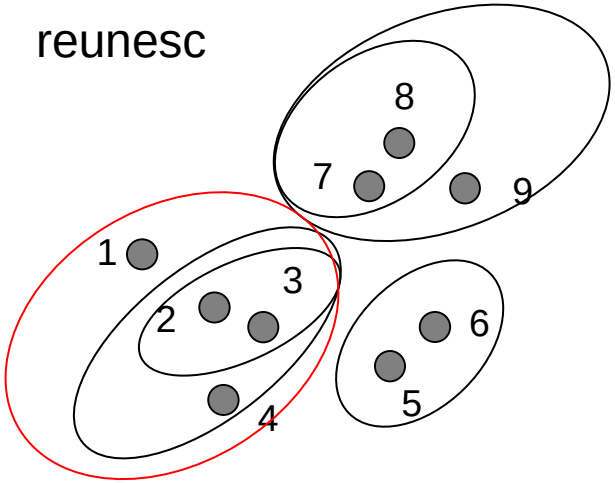
Idee: se identifică la fiecare etapă care sunt cele mai similare clustere şi se reunesc

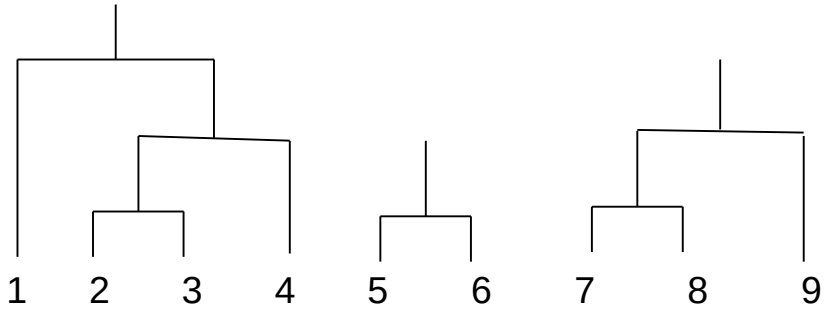




Metoda aglomerativă

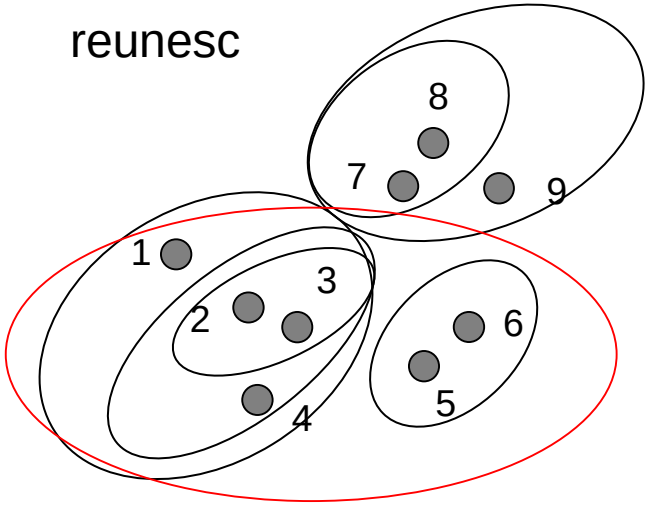
Idee: se identifică la fiecare etapă care sunt cele mai similare clustere şi se reunesc

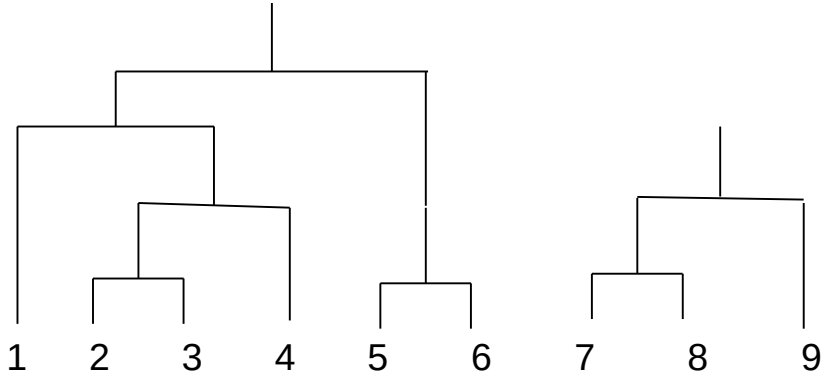




Metoda aglomerativă

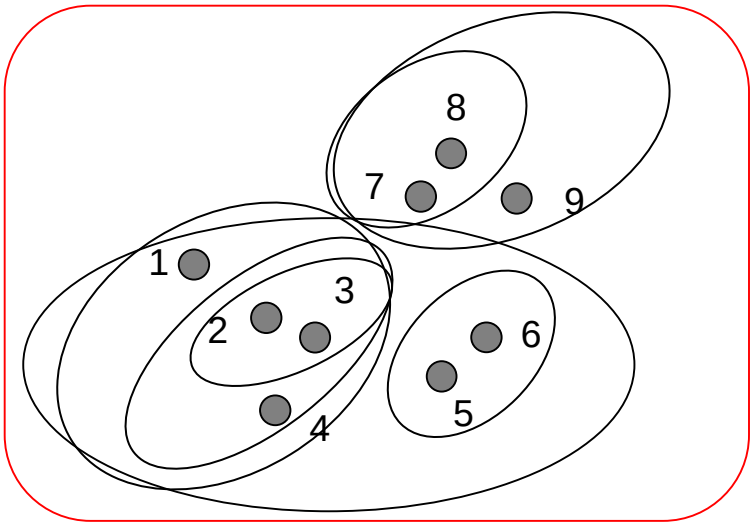
Idee: se identifică la fiecare etapă care sunt cele mai similare clustere şi se reunesc



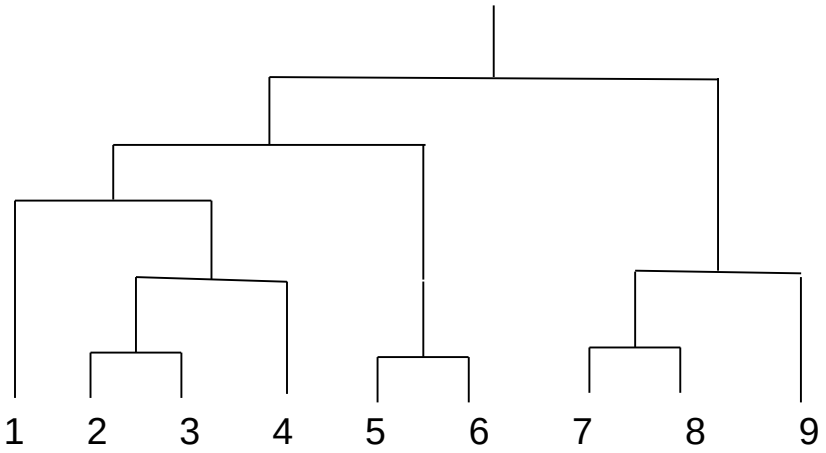


Metoda aglomerativă

Idee: se identifică la fiecare etapă care sunt cele mai similare clustere şi se reunesc



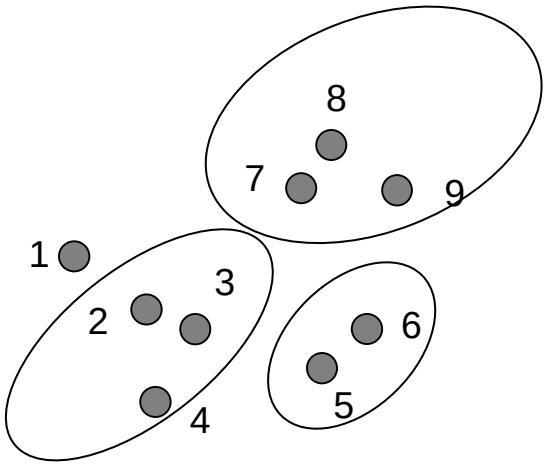
- Dendrograma rezultată



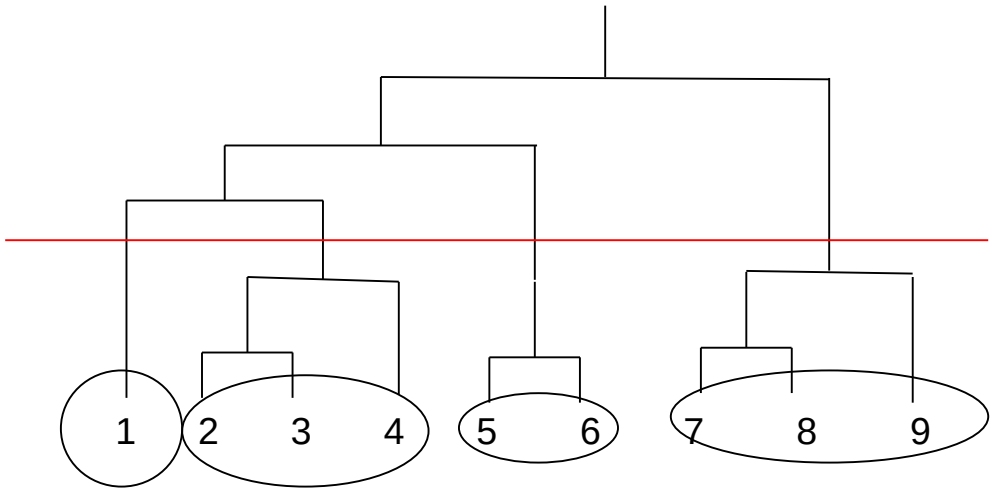
- Reprezentarea unei dendrograme: ca set de triplete ordonate (nivel, nr de clustere, clustere)

{(0,9,{{1},{2},…,{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



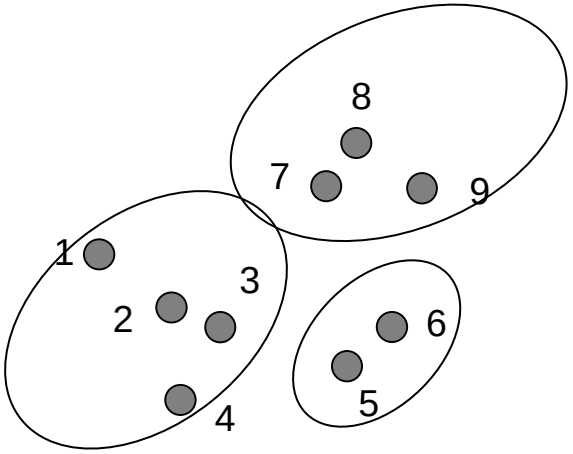
- Pentru a obţine o partiţie dendrograma trebuie secţionată



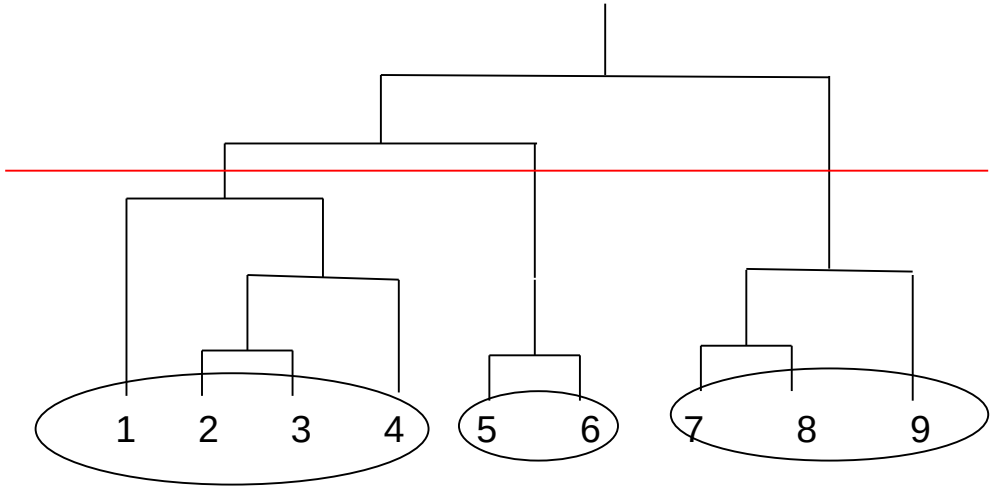
Partiţie:

C1={1}
C2={2,3,4}
C3={5,6}
C4={7,8,9}

Metoda aglomerativă



- Schimbând nivelul de secţionare se obţine o altă partiţie



Partition:

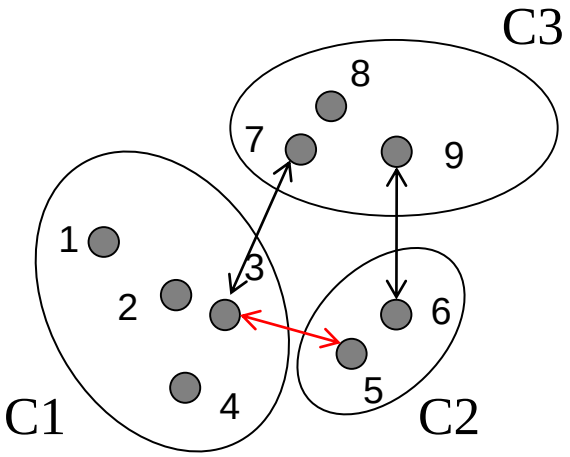
C1={1,2,3,4}
C2={5,6}
C3={7,8,9}

Metoda aglomerativă

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

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

- Single-linkage: cea mai mică disimilaritate (distanţă) între datele aparţinând unor clustere diferite



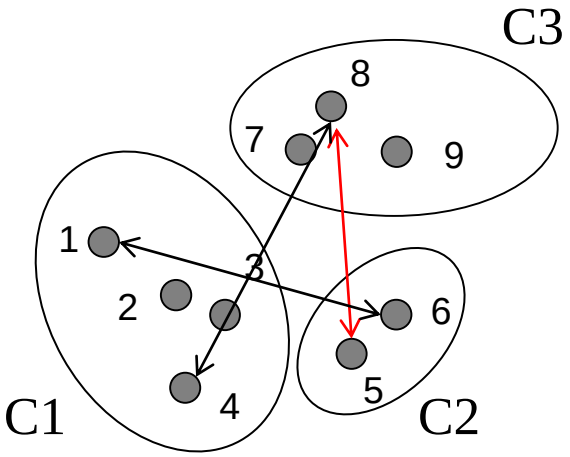
$$D_{SL}(C_i, C_j) = \min_{x \in C_i, y \in C_j} d(x, y)$$

Metoda aglomerativă

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

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

- Complete-linkage: cea mai mare disimilaritate (distanţă) între datele aparţinând unor clustere diferite



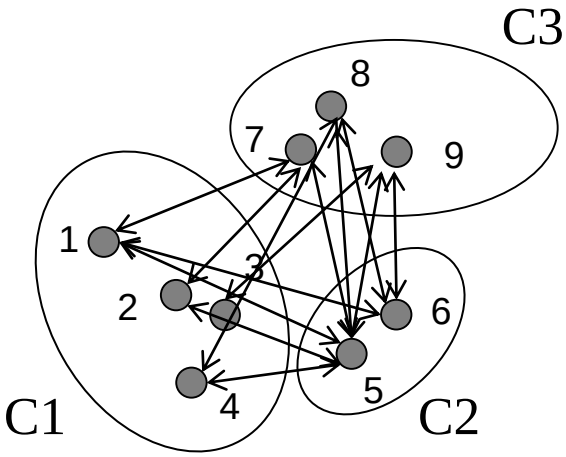
$$D_{CL}(C_i, C_j) = \max_{x \in C_i, y \in C_j} d(x, y)$$

Metoda aglomerativă

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

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

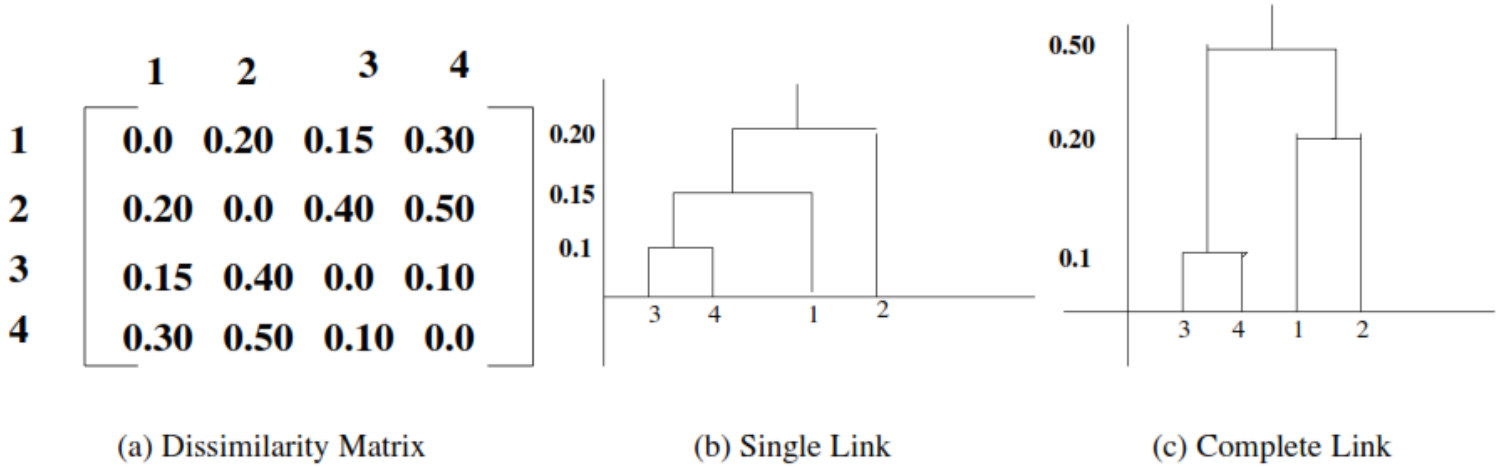
- Average-linkage: media distanţelor dintre datele aparţinând unor clustere diferite



$$D_{AL}(C_i, C_j) = \frac{1}{card(C_i)\,card(C_j)} \sum_{x \in C_i, y \in C_j} d(x, y)$$

Metoda aglomerativă

Măsura de disimilaritate folosită influenţează rezultatul grupării:



(a) Dissimilarity Matrix (b) Single Link (c) Complete Link

[Data Clustering: Algorithms and Applications, 2014]

Metoda aglomerativă

Algoritm

Input : set de date cu N instanţe

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

Output: dendrograma (set de triplete)

agglomerative(X,D)
level=0; k=N
$C=\{\{x_1\},\{x_2\},\ldots,\{x_N\}\}$; 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

- Funcţia mergeClusters identifică cele mai apropiate clustere şi le reuneşte
- Algoritmul are complexitate pătratică în raport cu numărul de date din set ($O(N^2)$)
- Este senzitiv la erorile din date

Metoda divizivă

Structura generică

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

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 $L_1,L_2,\ldots L_k$

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

until <a stopping criterion>

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

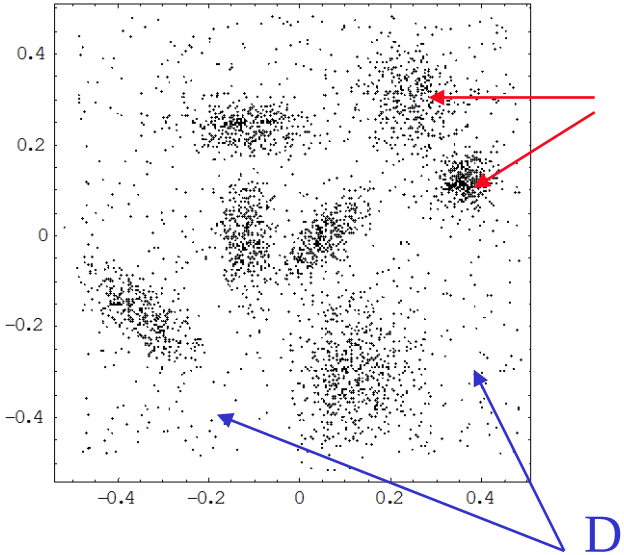
Bisecting Kmeans

- Varianta de algoritm de bisecţie 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 densităţii locale a datelor
  - se determină numărul de date din vecinătatea punctului analizat (DBSCAN)
  - se utilizează funcţii de influenţă pt estimarea densităţii (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 depăşeşte 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

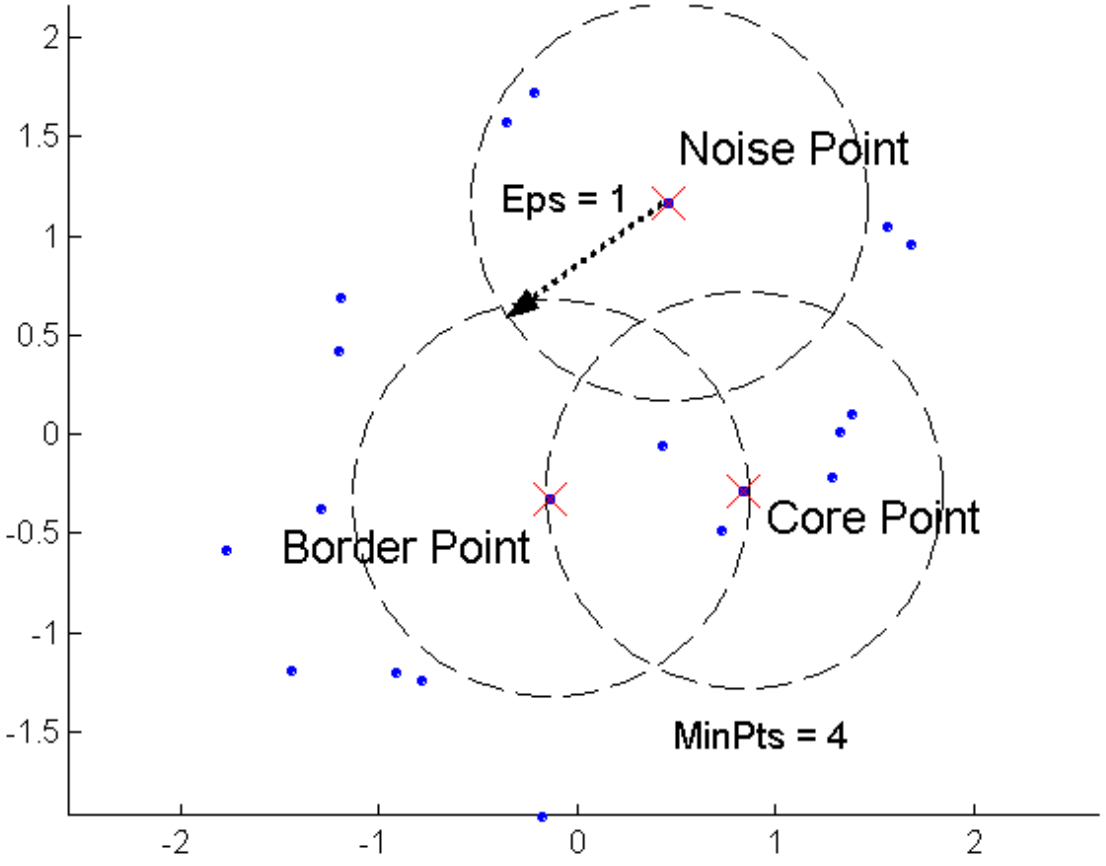
DBSCAN

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

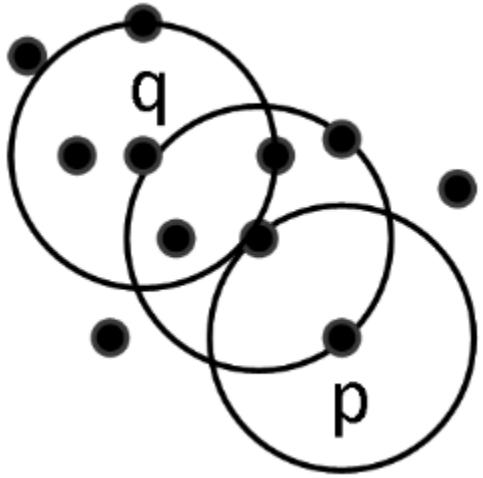
- Un punct q est accesibil direct (directly density reachable) dintr-un punct nucleu p dacă e în vecinătatea lui p; accesibilitatea în sens general este definită ca fiind închiderea tranzitivă a relaţiei de accesibilitate directă (există un lanţ de puncte nucleu ce începe cu p şi se termină cu q cu proprietatea 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

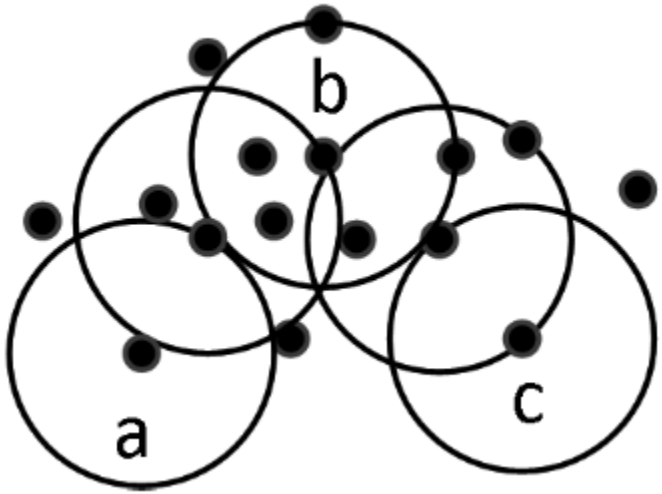
DBSCAN


Noise Point
Eps = 1
Border Point
Core Point
MinPts = 4
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-1
0
1
2

DBSCAN



p este accesibil din q



b este accesibil din a
c este accesibil din b
=> a şi c sunt conectate

Obs:

- Două pcte, a şi c, sunt conectate dacă există un punct b astfel încât b este accesibil atât din a cât şi din c
- Două puncte conectate ar trebui să aparţină aceluiaşi cluster => un cluster definit pe baza densităţii 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
```

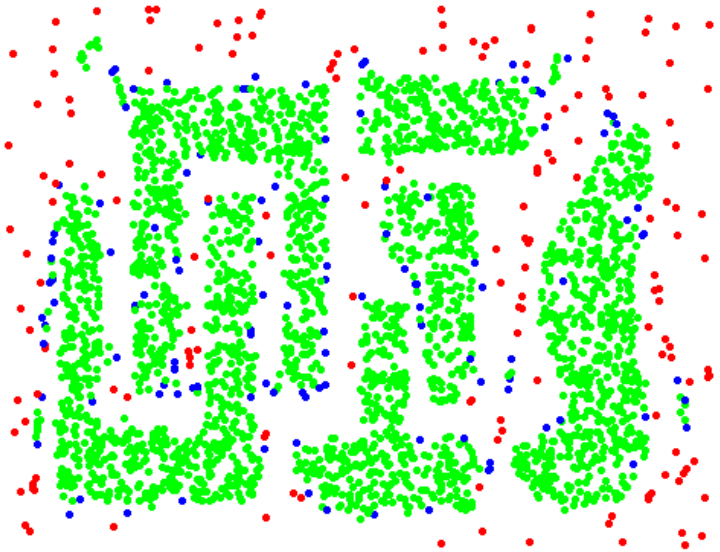
[M.Ester, H Kriegel et al, A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise, 1996]

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

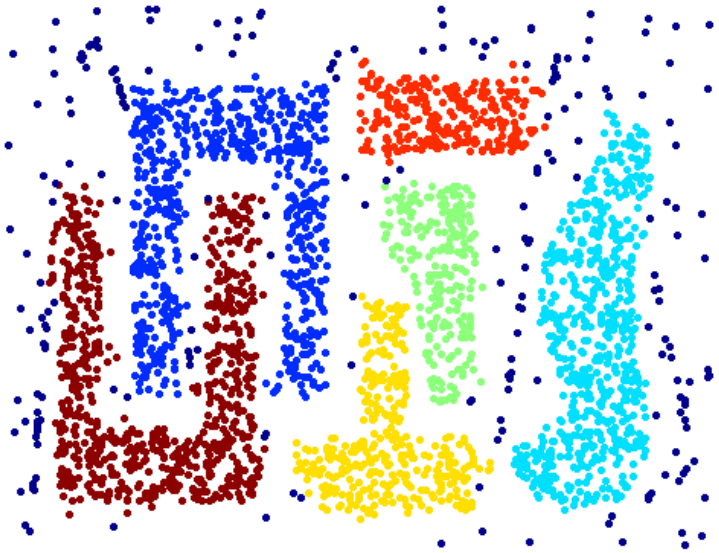


Tipuri de puncte : core, border şi noise

Eps = 10, MinPts = 4

DBSCAN

Date

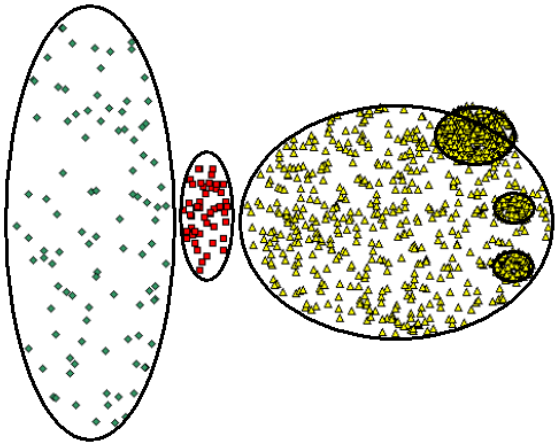


Clustere

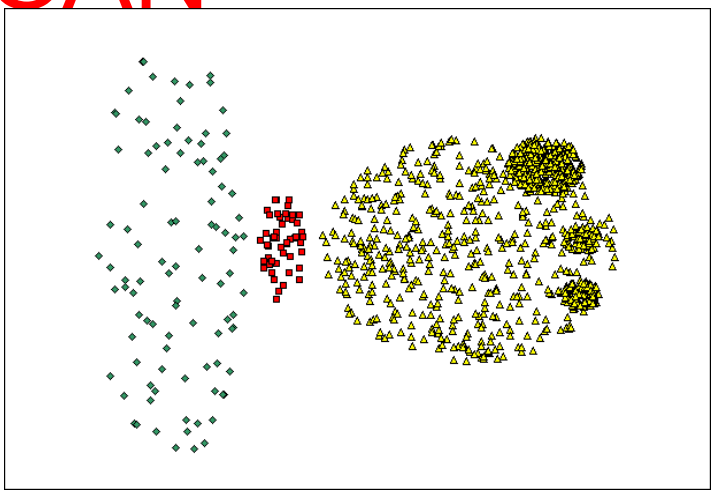
Specific DBSCAN:

- Permite identificarea clusterelor de diferite forme
- Robust

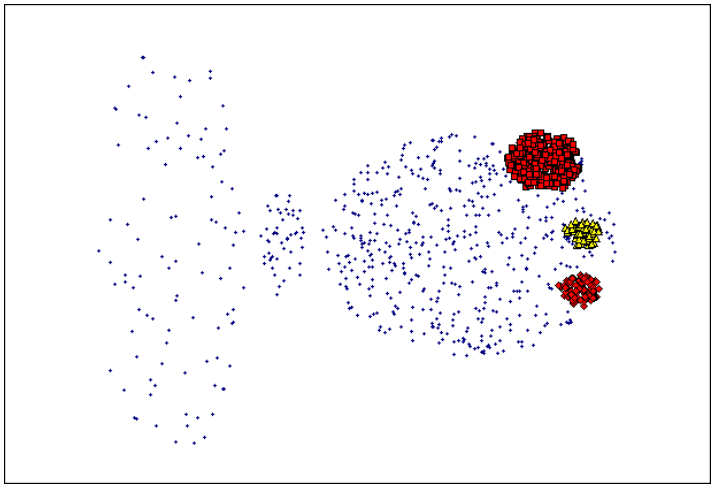
DBSCAN



Original Points



(MinPts=4, Eps=9.75).



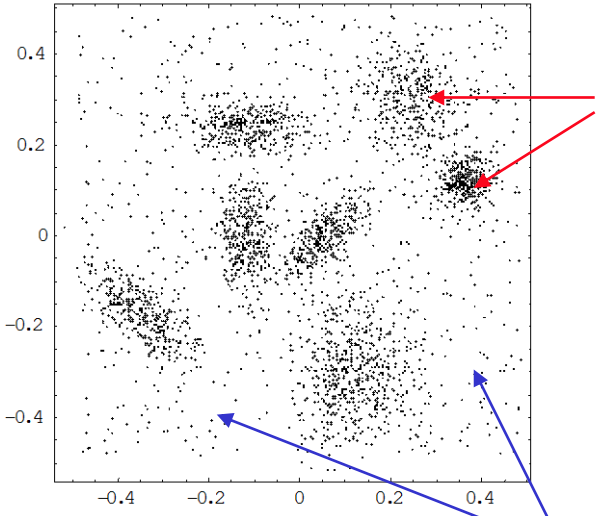
Nu este adecvat pt:

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

DENCLUE

- Cluster = grup dens de date similare separate de regiuni cu densitate mai mică de date

- Idee de bază: estimarea densităţii locale a datelor
  - se utilizează funcţii de influenţă pt estimarea densităţii



Funcţie de influenţă

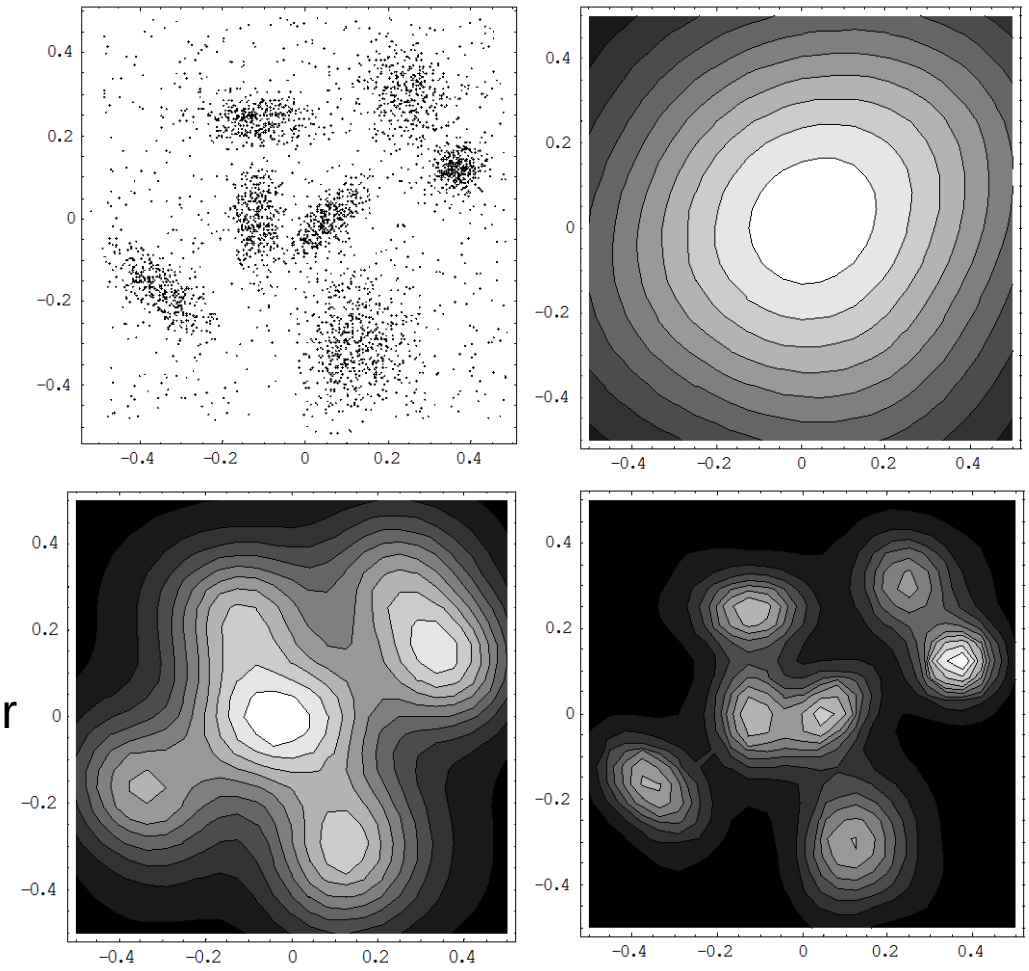
$$I_y(x) = \frac{1}{\sigma^{n/2}} \exp\left(-\frac{\sum_{j=1}^{n} (x_j - y_j)^2}{2\sigma^2} \right)$$

Funcţie de densitate

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

DENCLUE

- Forma funcţiei de densitate depinde de valoarea lui σ
- Dacă valoarea lui σ este adecvată, maximele locale ale funcţiei de densitate corespund reprezentanţilor clusterilor
- Pt valori mari ale lui σ fctia de densitate are un maxim unic
- Pt valori prea mici ale lui σ maximele locale corespund unor vârfuri izolate şi pot fi dificil de detectat



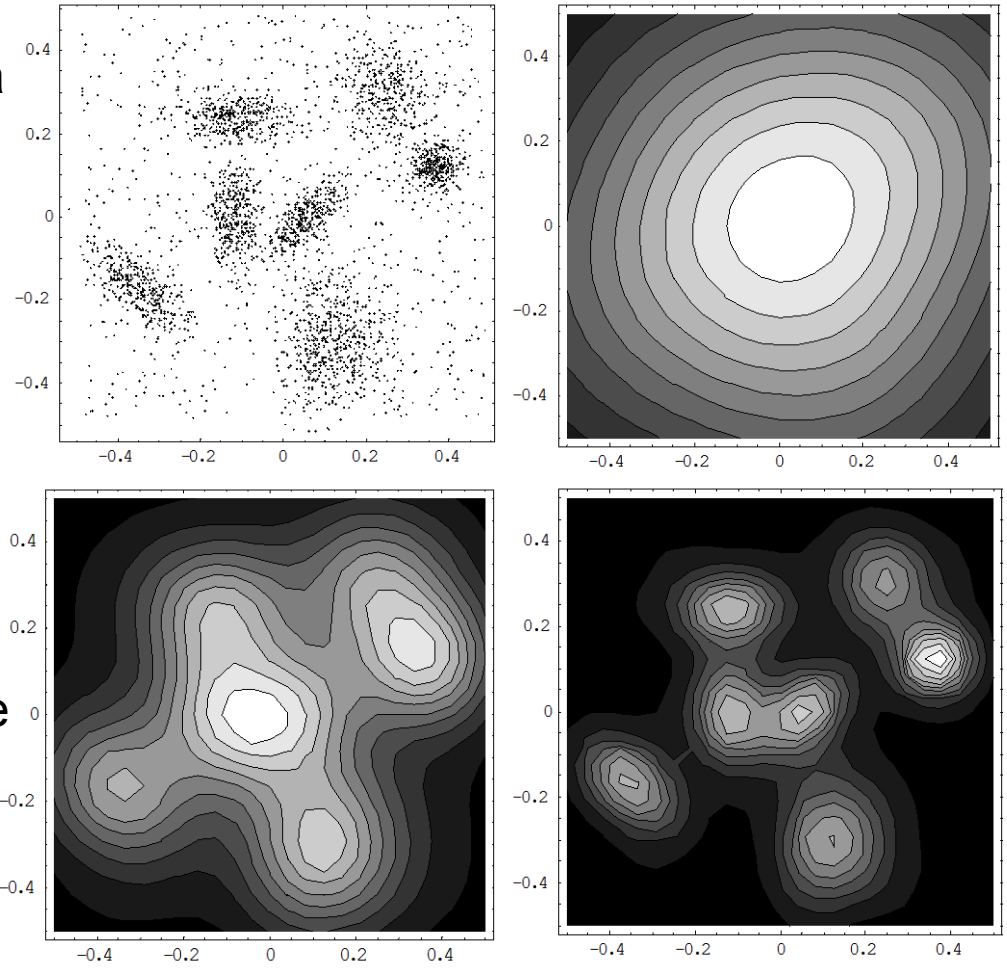
σ=0.1 σ=0.05

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:

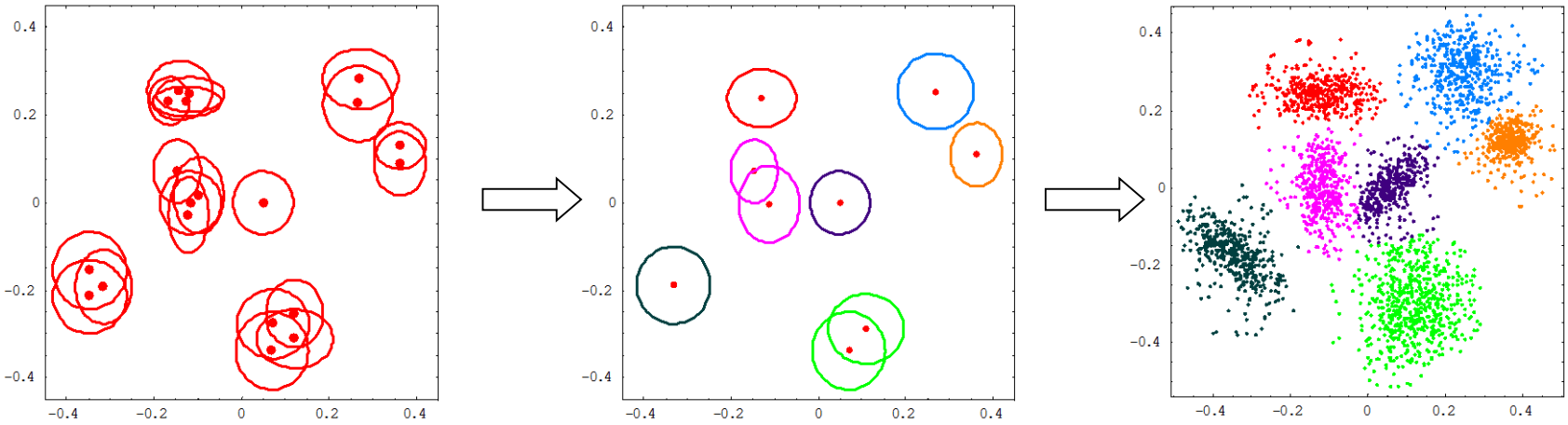
- Fiecare maxim local corespunde unui cluster (clusterele detectate vor fi sferice sau elipsoidale)
- Un cluster corespunde unui set de maxime locale "învecinate" (se pot identifica clustere de forma arbitrara)



σ=0.1 σ=0.05

DENCLUE

Exemple de rezultate



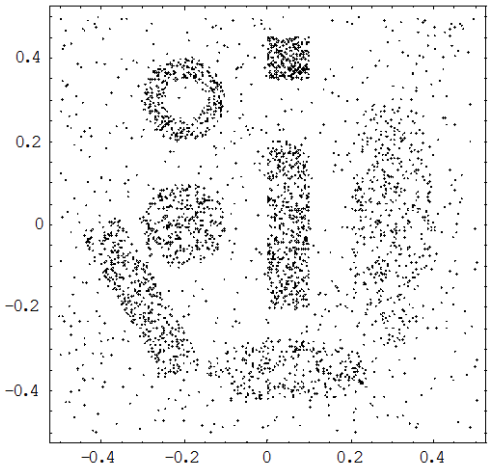
Punctele marcate = puncte de maxim ale funcției de densitate

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

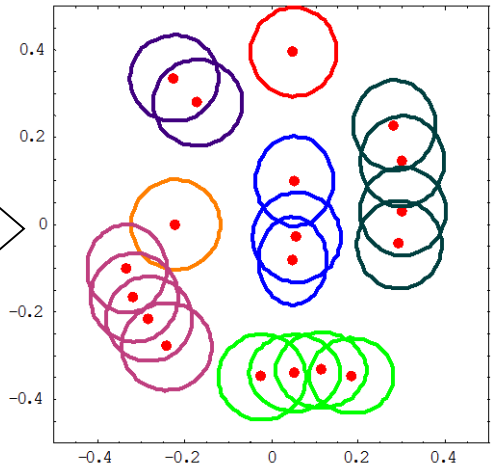
Grupare = distribuirea datelor în clustere se bazează pe valorile asociate funcțiilor de influență

DENCLUE

Exemple de rezultate

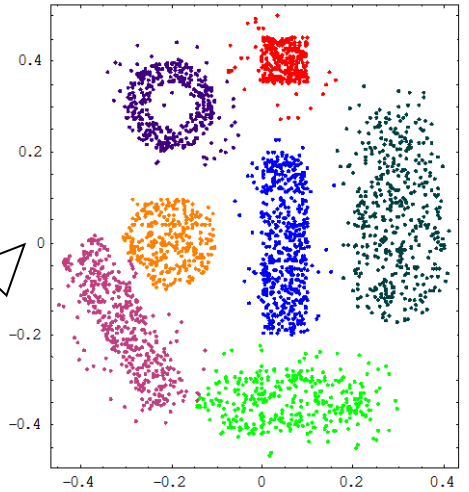


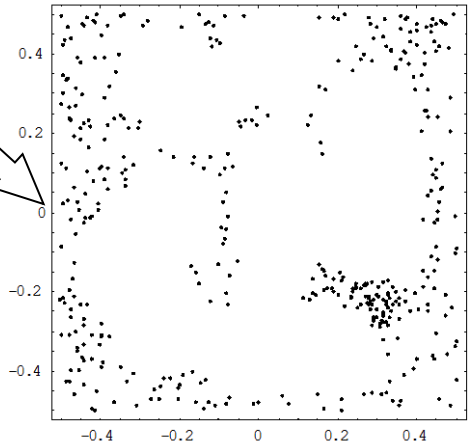
Date inițiale



Descriptori
ai clusterelor

Clustere identificate





Date considerate zgomot

Metode probabiliste

Idee de bază:

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

Exemple:

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

Algoritmul EM

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

Iniţializarea parametrilor modelelor şi a probabilităţilor de selecţie

- (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 probabilităţilor de asignare a datelor la clustere

Obs:

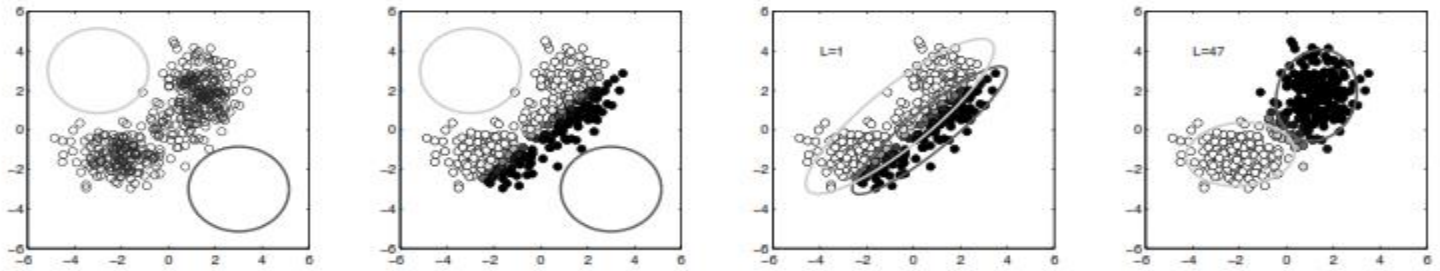
- In cazul datelor numerice se poate folosi distribuţia normală multi-dimensională pentru a modela datele aparţinând unui cluster (modelul de generare a datelor este o mixtură de gaussiene). Parametrii care se estimează în acest caz sunt media şi matricea de covarianţă.

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

Algoritmul EM

Ecuaţii care intervin in algoritmul EM

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

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

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

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

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

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

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

Cursul următor

- Reguli de asociere
- Algoritmul Apriori