Curs 8:

Gruparea datelor (II)

Structura

- Algoritmi bazați pe estimarea densității datelor
  - DBSCAN (Density Based Spatial Clustering of Applications with Noise)
  - DENCLUE (DENsity based CLUstEring)

- Algoritmi bazați pe modele probabiliste
  - EM (Expectation Maximization)

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?

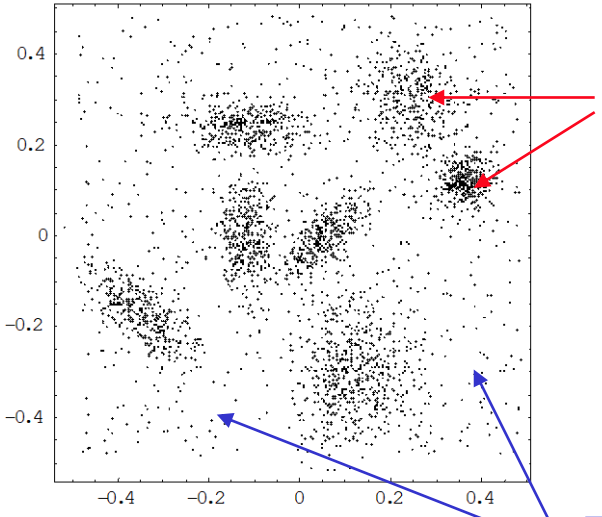
- Să se poată verifica dacă două date aparţin aceluiaşi cluster
- Să se determine clusterul de care aparţine o dată

Tehnici de grupare (reminder)

- Partiționale
  - Conduce la o partiție a datelor în clustere
  - Fiecare cluster este reprezentat de un prototip (centroid, medoid etc)
  - Favorizează clusterele de formă circulară
  - Algoritm reprezentativ: kMeans
    - Necesită specificarea numărului de clustere
    - Ordin de complexitate liniar în raport cu numărul de date
- Ierarhice
  - Conduce la o structură ierarhică (dendrogramă) construită folosind similaritățile/disimilaritățile dintre date/clustere
  - Generează partiții prin secționarea dendrogramei la un anumit nivel
  - Algoritm reprezentativ: aglomerativ
    - Există diferite variante de calcul a similarității între clustere
    - Ordin de complexitate cubic/pătratic în raport cu numărul de date

Metode bazate pe densitate

- Cluster = grup dens de date similare separate de regiuni cu densitate mai mică de date
- Problema principală:
  - Cum se estimează densitatea?
- 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)



DBSCAN

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

- Analiza densității locale (într-un punct) – estimare bazată pe numărarea elementelor din vecinătatea acelui punct

- Analiza densității la nivel de regiune (cluster) – estimarea gradului în care densitatea locală se "propagă" prin faptul că sunt conectate "puncte" cu densitate locală "mare" (zone cu densitate mare separate de o zonă cu densitate mică ar trebui să corespundă unor clustere diferite deci să fie considerate deconectate)

- Probleme:
  - Cum definim vecinătatea unui punct?
  - Cum stabilim că un punct face parte din zonă densă sau din zonă care nu e densă? Ce înseamnă densitate mare/mică?
  - Cum stabilim că două puncte sunt conectate (fac partea din aceeași regiune densă, adică din același cluster)

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 punctului respectiv definită de o anumită rază (Eps)
- Reminder: vecinătate de raza Eps(ε): $V_\varepsilon(p) = \{\mathrm{q}|\mathrm{d}(\mathrm{p},\mathrm{q}) \leq \varepsilon\}$

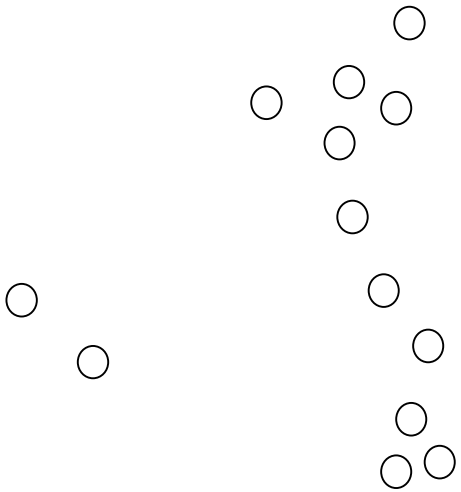
- Un punct este considerat că face parte dintr-o zonă densă, adică e un punct nucleu (core point) dacă numărul de puncte din vecinătatea sa depăşeşte un prag (MinPts); acestea sunt puncte considerate a fi în interiorul clusterului

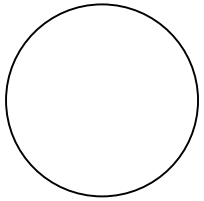
- Un punct frontieră (border point) are un număr de vecini mai mic decât MinPts dar este în vecinătatea unui punct nucleu; punctele frontieră sunt considerată ca făcând parte din cluster (doar că nu sunt în zona densă a clusterului

- Punctele care nu sunt nici puncte nucleu, nici puncte frontieră sunt considerate zgomot (nu aparțin nici unui cluster)

DBSCAN

Exemplu (simplu)





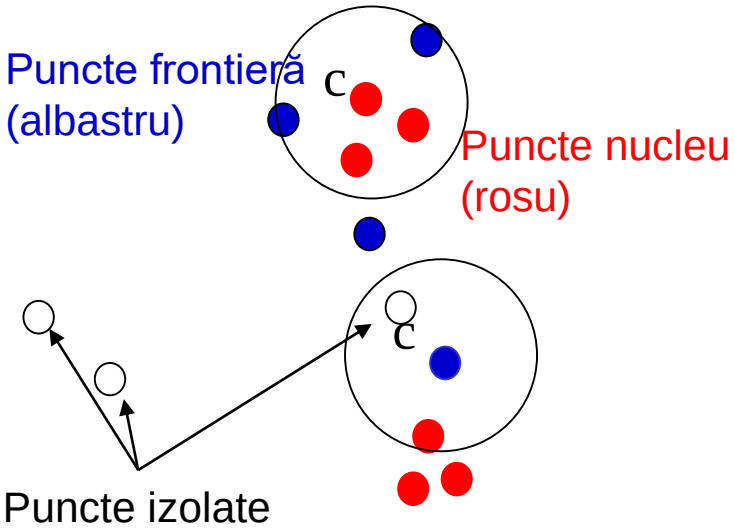
Vecinătate

Raza vecinătate (eps)

Număr minim puncte: 3

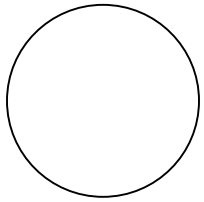
DBSCAN

Exemplu (simplu)



Raza vecinătate (eps)

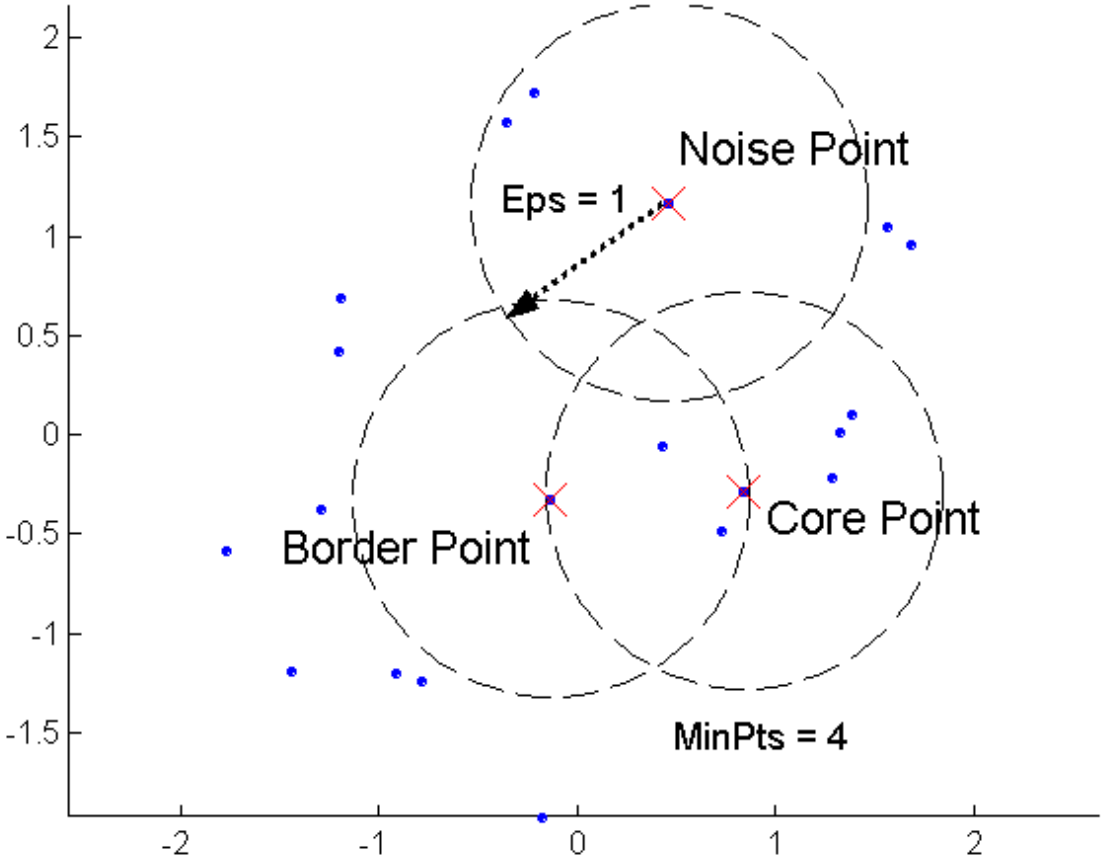
Vecinătate



Număr minim puncte: 3

DBSCAN

Alt exemplu (număr minim de puncte în vecinătatea unui punct nucleu: 4)



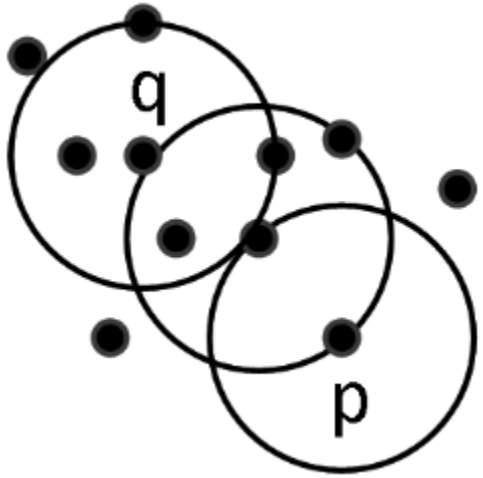
DBSCAN

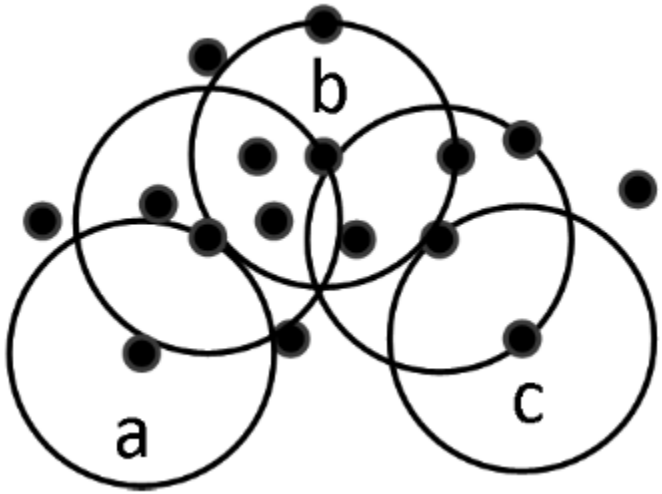
Propagarea densității - accesibilitate

- Un punct p este accesibil direct (directly density reachable) dintr-un punct nucleu q dacă este în vecinătatea lui q (d(p,q)<=Eps);
  - Obs: definiția accesibilității directe se bazează pe condiția ca punctul origine (q) să fie punct nucleu

- Un punct p este accesibil (density reachable) dintr-un punct nucleu q dacă există o secvență de puncte $q=p_1,p_2,...,p_n=p$ cu proprietatea că p_{i+1} este accesibil direct din p_i
  - Obs: toate punctele din secvență (cu excepția lui p) trebuie să fie puncte nucleu
  - Obs: relația de accesibilitate poate fi interpretată ca fiind închiderea tranzitivă a relației de accesibilitate directă
- Obs: relația de accesibilitate nu e simetrică (e posibil ca p să fie accesibil din q, dar q să nu fie accesibil din p, dacă, de exemplu p nu este punct nucleu)

DBSCAN





p este accesibil din q

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

Conectivitate:

- Două puncte, a şi c, sunt considerate conectate dacă există un punct (nucleu) b astfel încât a este accesibil din b și c este accesibil din b

- Două puncte conectate ar trebui să aparţină aceluiaşi cluster => un cluster definit pe baza densităţii este un set maximal de date conectate

DBSCAN

Ideea generală a algoritmului:

- Se pornește de la un punct arbitrar p
- Dacă p este punct nucleu atunci se identifică toate punctele accesibile din p și se marchează ca făcând parte din același cluster (se asignează eticheta specifică clusterului
- Se trece la un alt punct și se continuă analiza

Complexitate (în raport cu numărul, N, de elemente din setul de date):

- $O(N^2)$ – implementare directă
- O(NlogN) – dacă se utilizează o structură de indexare care asigură acces eficient la datele aflate în vecinătate

DBSCAN

ALGORITHM 1: Pseudocode of Original Sequential DBSCAN

```
Input: DB: Database
Input: ε: Radius
Input: minPts: Density threshold
Input: dist: Distance function
Data: label: Point labels, initially undefined
1  foreach point p in database DB do
2      if label(p) ≠ undefined then continue
3      Neighbors N ← RangeQuery(DB, dist, p, ε)
4      if |N| < minPts then
5          label(p) ← Noise
6          continue
7      c ← next cluster label
8      label(p) ← c
9      Seed set S ← N \ {p}
10     foreach q in S do
11         if label(q) = Noise then label(q) ← c
12         if label(q) ≠ undefined then continue
13         Neighbors N ← RangeQuery(DB, dist, q, ε)
14         label(q) ← c
15         if |N| < minPts then continue
16         S ← S ∪ N
```

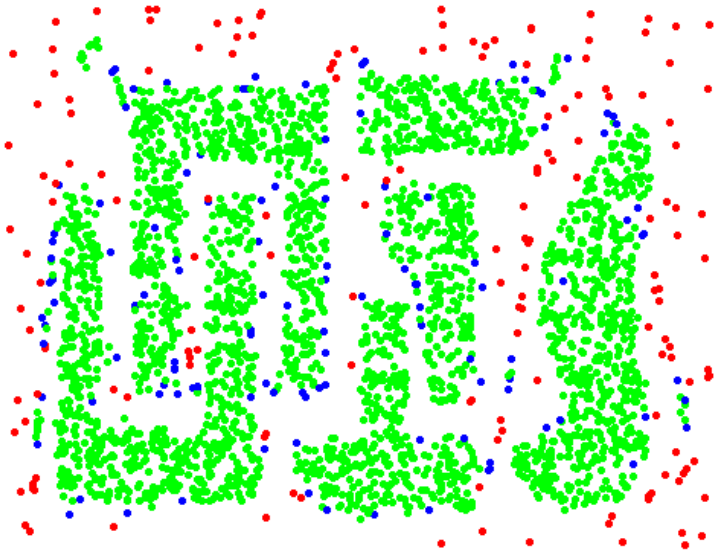
Ideea de bază:

- Etichetarea datelor se bazează pe parcurgerea setului de date și asignarea unei etichete
- Date care initial au fost etichetate ca nois pot să fie ulterior asignate unui cluster
- RangeQuery returnează toate datele care sunt in vecinătatea datei procesate la etapa curentă (modul de implementare a acestei funcții influențează eficiența algoritmului)

E. Schubert, J. Sander, M. Ester, H-P Kriegel, and X. Xu. - DBSCAN Revisited, Revisited: Why and How You Should (Still) Use DBSCAN. *ACM Trans. Database Syst.* 42, 3, Article 19, 2017

DBSCAN

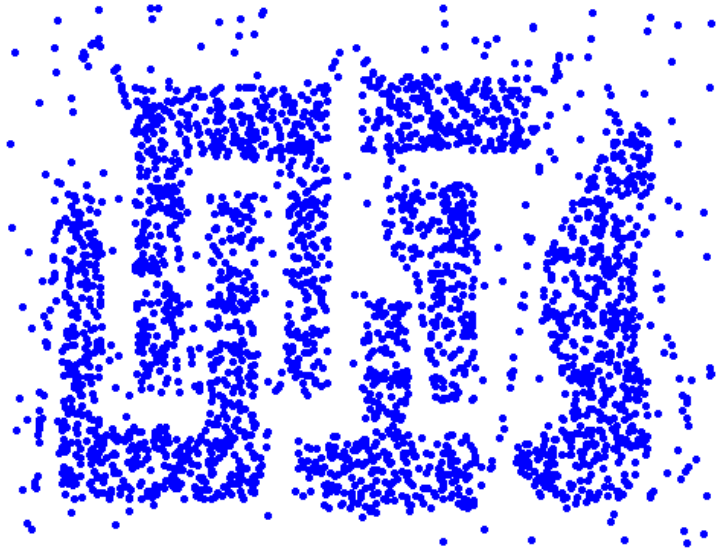
Date (puncte) de prelucrat



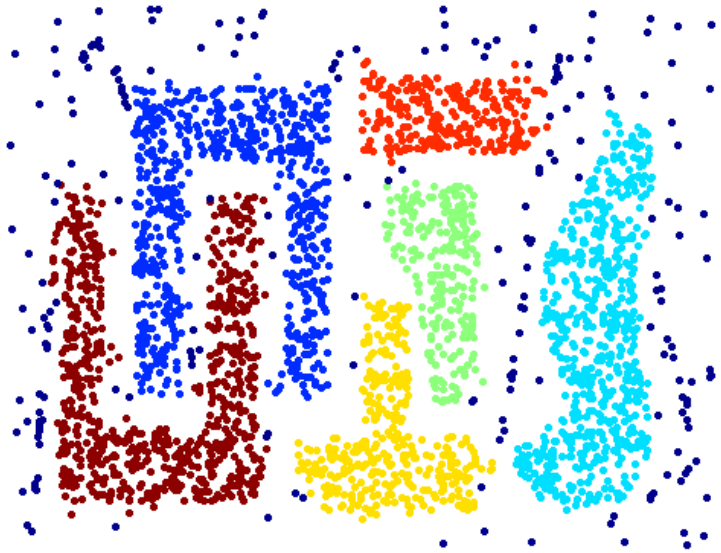
Tipuri de puncte : core, border şi noise

Eps = 10, MinPts = 4

DBSCAN



Date

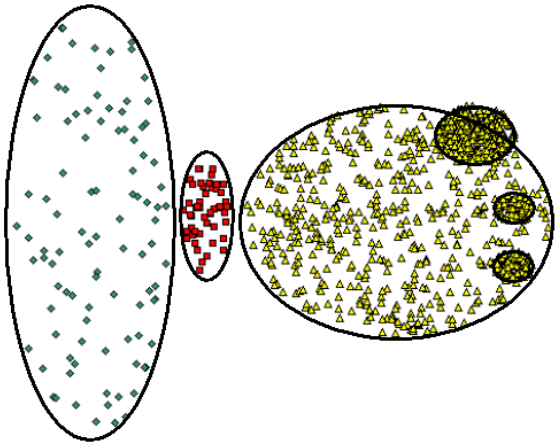


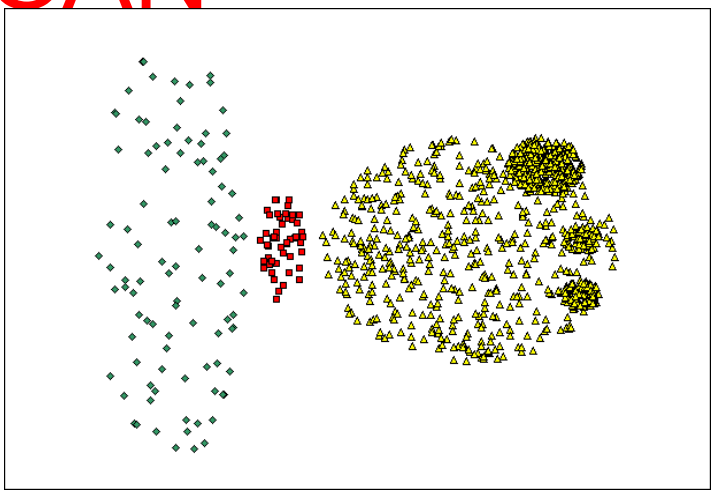
Clustere

Avantaj DBSCAN:

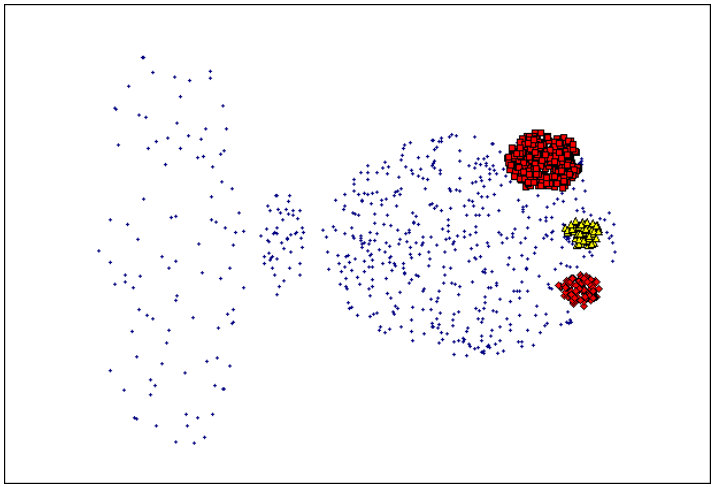
- Permite identificarea clusterelor de diferite forme
- Nu necesită specificarea numărului de clustere

DBSCAN





(MinPts=4, Eps=9.75).



Dezavantaje DBSCAN:

- este sensibil la variaţii în densitatea datelor
- nu este adecvat pentru date cu multe atribute
- este sensibil la valorile parametrilor de control (MinPts, Eps)

DBSCAN

Alte variante:

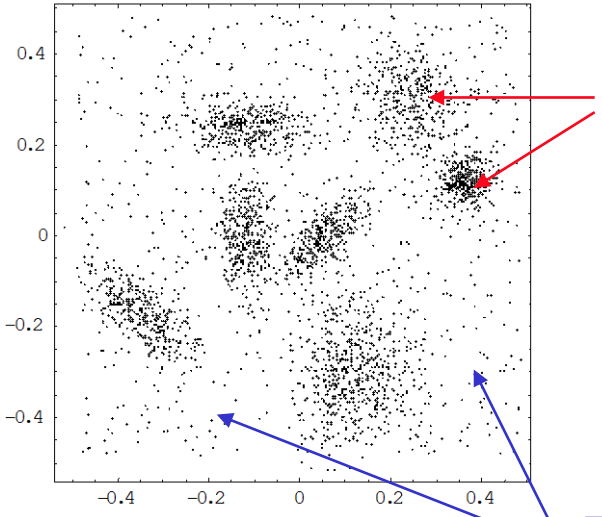
- OPTICS – Ordering Points to Identify Clustering Structure [Ankerst, Breuning, Kriegel, Sanders, 1999]
  - Idee: punctele care au densitate mai mare sunt procesate primele
  - Permite identificarea unei structuri ierarhice de clustere
  - Mai puțin sensitiv la alegerea parametrilor (MinPts si Eps)

- Grid-based Clustering:
  - Idee: partitionarea spațiului datelor în celule prin definirea unei grile de discretizare pe fiecare dimensiune
  - Exemple:
    - STING = Statistical Information Grid Approach (Wang et al. 1997) – se bazează pe o ierarhie de grile și pe calculul unor indicatori statistici pentru fiecare celulă din grilă
    - CLIQUE = Clustering in QUEst (IBM data structure) (Agrawal et al, 1998) - se bazează pe identificarea regiunilor dense in manieră incrementală – folosind subseturi de atribute)

DENCLUE

[Hinnenburg, Keim - An efficient approach to clustering in large multimedia databases with noise- 1998)

- 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
- Funcții de influență
  - $I_y(x)$ reprezintă influența lui x asupra lui y (cu cât y este mai aproape de x influența este mai mare) – exemplu: fcție gaussiană
  - f(x) = influența medie a lui x



Funcţie de densitate

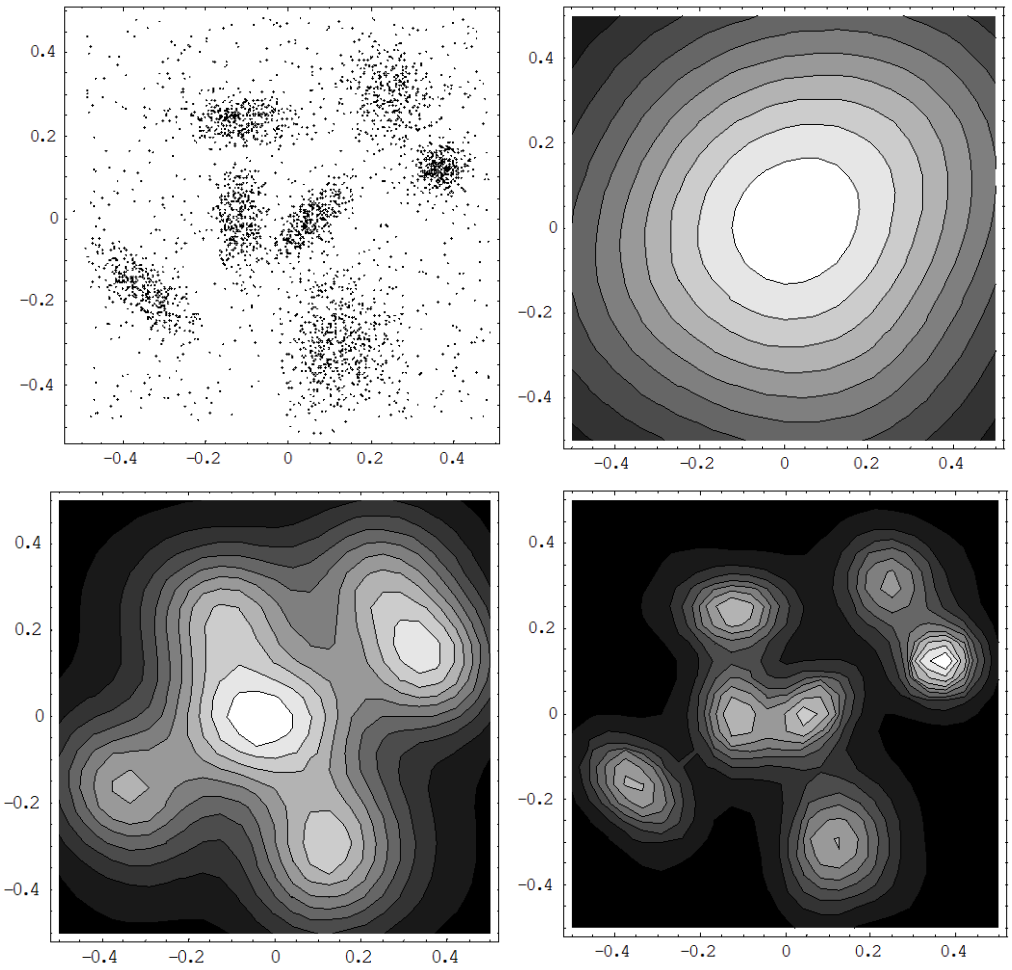
$$I_y(x) = \frac{1}{\sigma^{n/2}} \exp\left(-\frac{\sum_{j=1}^{n}(x_j - y_j)^2}{2\sigma^2}\right)$$

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

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 funcția de densitate are un maxim unic (σ=0.5)
- Pt valori prea mici ale lui σ maximele locale corespund unor vârfuri izolate şi pot fi dificil de detectat (σ=0.05)

σ=0.5



σ=0.1

σ=0.05

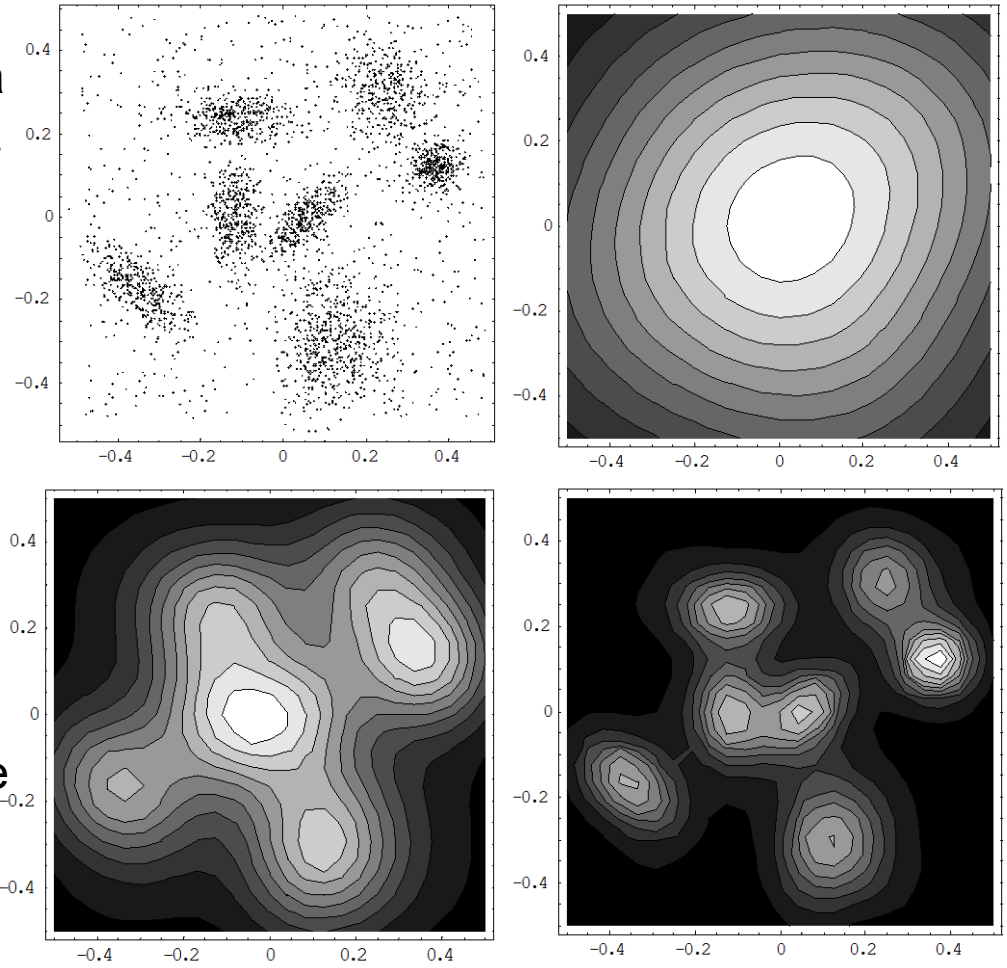
DENCLUE

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 (pentru $I_y(x)$ de tip gaussian clusterele detectate vor fi sferice sau elipsoidale)

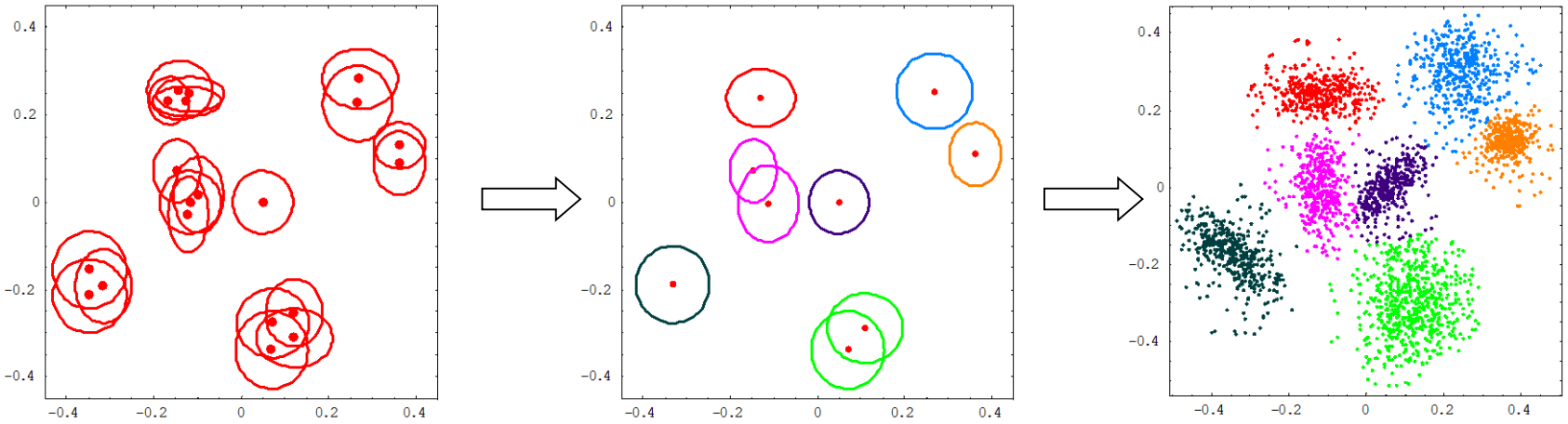
- Un cluster corespunde unui set de maxime locale "învecinate" (se pot identifica clustere de forma arbitrară)



σ=0.1 σ=0.05

DENCLUE

Exemple de rezultate



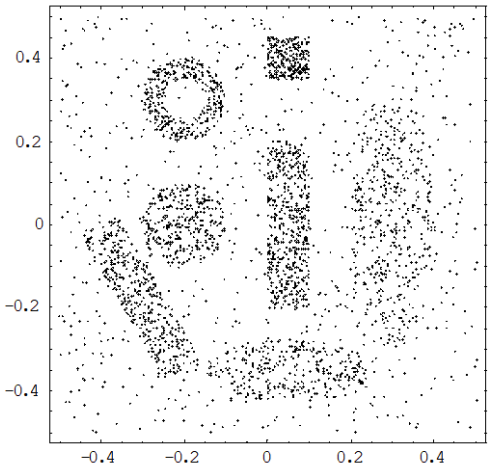
Punctele marcate = puncte de maxim ale funcției de densitate

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

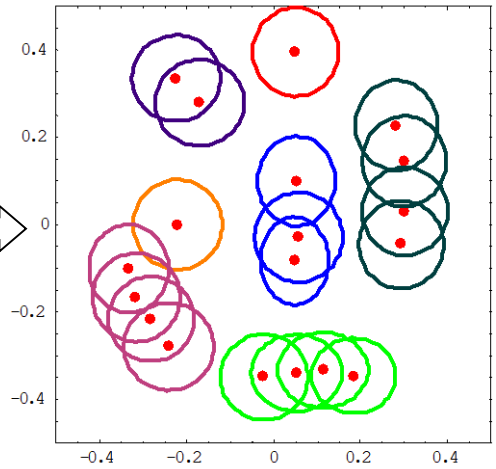
Grupare = distribuirea datelor în clustere se bazează pe valorile asociate funcțiilor de influență

DENCLUE

Exemple de rezultate

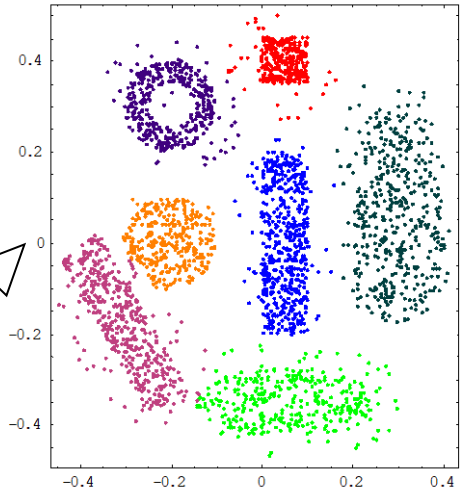


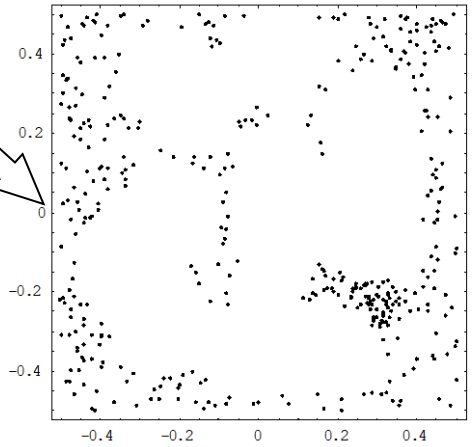
Date inițiale



Descriptori
ai clusterelor

Clustere identificate





Date considerate zgomot

Metode probabiliste

Idee de bază:

- Se presupune că datele sunt generate de un proces stohastic (o mixtură de distribuţii de probabilitate, fiecare dintre distribuții corespunzând unui cluster)
- Scopul algoritmului de grupare este de a descoperi modelul probabilist, adică de a identifica distribuţiile de probabilitate și parametrii mixturii

Exemplu: să presupunem că setul de date este generat în felul următor

- Se aruncă o monedă (despre care nu știm dacă este corectă, deci pentru care nu cunoaștem valorile pentru p_1 = probabilitatea de a obține cap, p_2 = probabilitatea de a obține pajură)
- Dacă la aruncarea monedei s-a obținut
  - Cap: se generează o dată folosind sursa 1 de date (de exemplu o distribuție normală cu media m_1 și matricea de covarianță C_1)
  - Pajură: se generează o dată folosind sursa 2 de date (de exemplu o distribuție normală cu media m_2 și matricea de covarianță C_2)

Metode probabiliste

Exemplu:

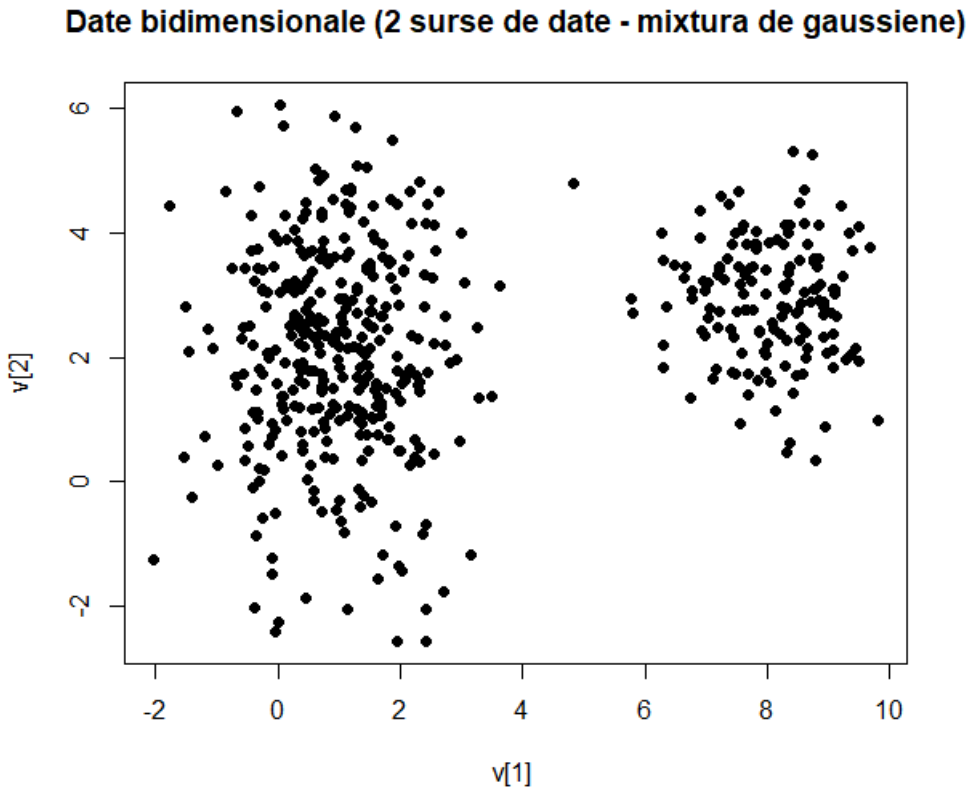
p1=0.7 (probabilitate cap),

p2=0.3 (probabilitate pajură),

N= 500 (nr date), n=2 (date bidimensionale)

Sursa 1: gaussiana cu m1=[1,2], C1=[[1,0],[0,3]]

Sursa 2: gaussiana cu m2=[8,3], C2=[[1,0],[0,1]]



Metode probabiliste

Problema:

- Fiecare dată x_i a fost generată folosind una dintre surse dar nu știm care dintre ele
- Dacă am cunoaște parametrii fiecărei surse (m_1 și C_1, respectiv m_2 și C_2) și care este probabilitatea fiecăreia de a fi fost folosită (p_1 și p_2) atunci ar fi suficient să calculăm pentru fiecare dată x_i și fiecare dintre cele două surse care este șansa ca data x_i să fi fost generată de sursa k:

$$r_{ik} = \frac{p_k \cdot Prob(x_i; m_k, C_k)}{\sum_{l=1}^{K} p_l \cdot Prob(x_i; m_l, C_l)} \qquad \text{(Rel. 1)}$$

Obs: Valorile r sunt denumite responsabilități (cuantifică în ce măsură este responsabilă sursa k de generarea datei i) și sunt considerate variabile latente (în sensul că nu intervin în mod explicit în generarea datelor și nici nu sunt observabile)

In ipoteza că sursele au distribuție normală, probabilitățile care intervin în relația de mai sus (Rel. 1) sunt:

$$Prob(x_i; m_k, C_k) = \frac{1}{(2\pi)^{\frac{n}{2}} \det(C_k)^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x - m_k)^T C_k^{-1}(x - m_k)\right)$$

Metode probabiliste

Problema:

- Pe de altă parte ...
- Dacă am cunoaște responsabilitățile r_{ik} atunci s-ar putea estima probabilitățile de selecție ale sursei:

$$p_k = \frac{1}{N} \sum_{i=1}^{N} r_{ik} \qquad \text{(Rel. 2)}$$

și parametrii surselor – în cazul distribuției normale este vorba de medie și matricea de covarianță – prin maximizarea verosimilității de a observa datele din set (metoda verosimilității maxime folosită la estimarea parametrilor modelelor statistice) care conduce la:

$$m_k = \frac{\sum_{i=1}^{N} r_{ik} x_i}{\sum_{i=1}^{N} r_{ik}} \qquad \text{(Rel. 3)}$$

$$C_k = \frac{\sum_{i=1}^{N} r_{ik} x_i x_i^T}{\sum_{i=1}^{N} r_{ik}} - m_k m_k^T \qquad \text{(Rel. 4)}$$

Algoritmul EM

Algoritmul Expectation-Maximization (EM) permite estimarea iterativă a valorilor responsabilităților și a parametrilor surselor

- 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 (inclusiv estimări ale parametrilor distribuțiilor de probabilitate care generează datele)

Structura generală algoritm EM:

Iniţializarea parametrilor modelelor și a probabilităţilor de selecţie

REPEAT

- (E-Step) Se determină valorile responsabilităţilor folosind valorile curente ale parametrilor modelelor (Rel. 1)
- (M-Step) Se determină probabilitățile de selecție ale surselor (Rel. 2) și parametrii acestora (Rel. 3 și 4) folosind valorile curente ale responsabilităților

UNTIL <conditie de oprire>

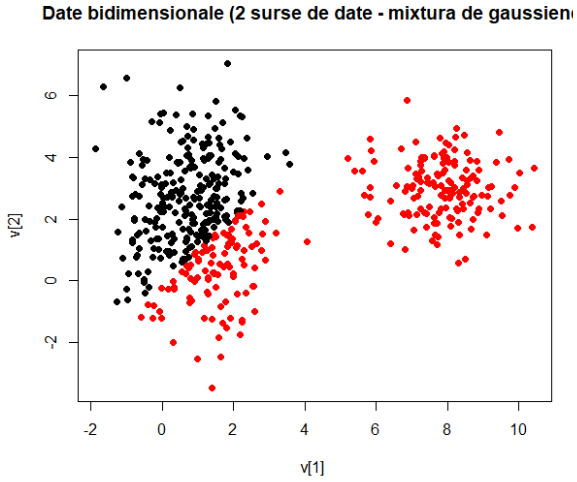
Algoritmul EM

- 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 (inclusiv estimări ale parametrilor distribuțiilor de probabilitate care generează datele)

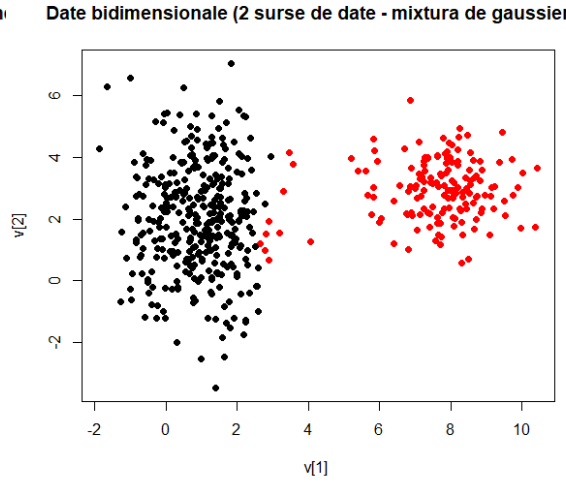
Construirea partiției:

- După estimarea tuturor parametrilor, partiția se obține folosind valorile responsabilităților: pentru o dată x_i se determină clusterul k pentru care valoarea responsabilității r_{ik} este maximă.

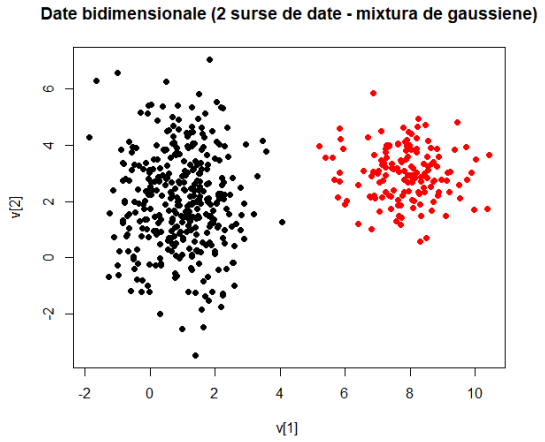
Iter=10



Iter=20



Iter=50



Sumar

- Algoritmi bazați pe densitate
  - Nu necesita specificarea numărului de clustere insă necesită specificarea unor parametri corelați cu informații privind măsurarea densității:
    - DBSCAN: Eps si Nmin
    - DENCLUE: Parametri sigma pt funcțiile de densitate (nucleu)
  - Permit identificarea unor clustere de forme arbitrare si separarea datelor de tip zgomot

- Algoritmul Expectation Maximization (EM)
  - Similar cu kMeans însă se bazează pe modelarea probabilistă a surselor de date
  - Permit identificarea clusterelor elipsoidale cu orientare arbitrară (matricea de covarianță nu e neapărat diagonală)

Cursul următor

- Reguli de asociere
- Algoritmul Apriori