Regroupement

(clustering)

Qu'est ce qu'un bon regroupement ?

- Une bonne méthode de regroupement permet de garantir
  - Une grande similarité intra-groupe
  - Une faible similarité inter-groupe
- La qualité d'un regroupement dépend donc de la mesure de similarité utilisée par la méthode et de son implémentation

Structures de données

- Matrice de données

$$\begin{bmatrix} x_{11} & \cdots & x_{1f} & \cdots & x_{1p} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{i1} & \cdots & x_{if} & \cdots & x_{ip} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{n1} & \cdots & x_{nf} & \cdots & x_{np} \end{bmatrix}$$

- Matrice de similarité

$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & & \\ d(n,1) & d(n,2) & \cdots & \cdots & 0 \end{bmatrix}$$

Mesurer la qualité d'un clustering

- Métrique pour la similarité: La similarité est exprimée par le biais d'une mesure de distance
- Une autre fonction est utilisée pour la mesure de la qualité
- Les définitions de distance sont très différentes que les variables soient des intervalles (continues), catégories, booléennes ou ordinales
- En pratique, on utilise souvent une pondération des variables

Types des variables

- Intervalles:
- Binaires:
- catégories, ordinales, ratio:
- Différents types:

Intervalle (discrètes)

- Standardiser les données
  - Calculer l'écart absolu moyen:

où

  - Calculer la mesure standardisée (*z-score*)

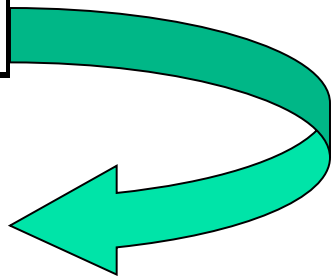
$$z_{if} = \frac{x_{if} - m_f}{s_f}$$

Exemple

	Age	Salaire
Personne1	50	11000
Personne2	70	11100
Personne3	60	11122
Personne4	60	11074

$M_{Age}=6$ $S_{Age}=5$

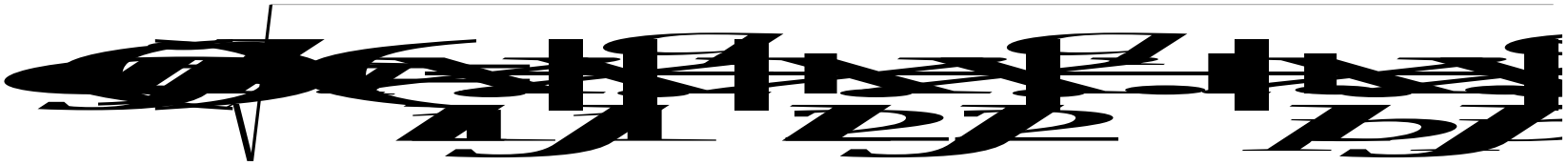
$M_{Salaire}=11074$



	Age	Salaire
Personne1	-2	-0,5
Personne2	2	0,175
Personne3	0	0,324
Personne4	0	2

Similarité entre objets

- Les distances expriment une similarité
- Ex: la *distance de Minkowski* :

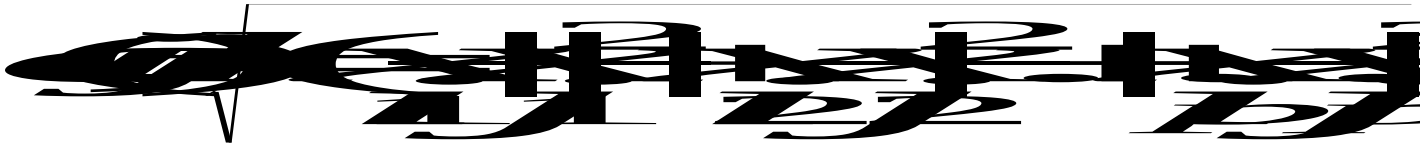


où $i = (x_{i1}, x_{i2}, \ldots, x_{ip})$ et $j = (x_{j1}, x_{j2}, \ldots, x_{jp})$ sont deux objets p-dimensionnels et q un entier positif

- Si $q = 1$, d est la distance de Manhattan

Similarité entre objets(I)

- *Si q = 2, d est la distance Euclidienne* :



  - Propriétés
    - $d(i,j) \geq 0$
    - $d(i,i) = 0$
    - $d(i,j) = d(j,i)$
    - $d(i,j) \leq d(i,k) + d(k,j)$

Exemple: distance de Manhattan

	Age	Salaire
Personne1	50	11000
Personne2	70	11100
Personne3	60	11122
Personne4	60	11074

→ d(p1,p2)=120

d(p1,p3)=132

Conclusion: p1 ressemble plus à p2 qu'à p3 ☹

	Age	Salaire
Personne1	-2	-0,5
Personne2	2	0,175
Personne3	0	0,324
Personne4	0	0

→ d(p1,p2)=4,675

d(p1,p3)=2,324

Conclusion: p1 ressemble plus à p3 qu'à p2 ☺

Variables binaires

- Une table de contingence pour données binaires

		Objet *j*		
		1	0	*sum*
Objet *i*	1	*a*	*b*	*a+b*
	0	*c*	*d*	*c+d*
	sum	*a+c*	*b+d*	*p*

a= nombre de positions où i a 1 et j a 1

- Exemple o_i=(1,1,0,1,0) et o_j=(1,0,0,0,1)

a=1, b=2, c=1, d=2

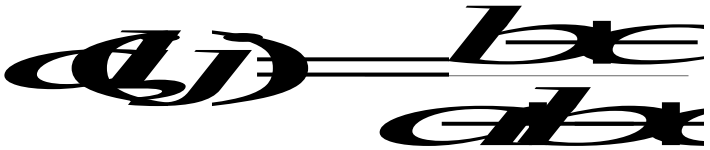
Mesures de distances

- Coefficient d'appariement (matching) simple (invariant pour variables symétriques):

Exemple $o_i=(1,1,0,1,0)$ et $o_j=(1,0,0,0,1)$

$d(o_i, o_j)=3/5$

- Coefficient de Jaccard



$d(o_i, o_j)=3/4$

Variables binaires (I)

- Variable symétrique: Ex. le sexe d'une personne, i.e coder masculin par 1 et féminin par 0 c'est pareil que le codage inverse

- Variable asymétrique: Ex. Test HIV. Le test peut être positif ou négatif (0 ou 1) mais il y a une valeur qui sera plus présente que l'autre. Généralement, on code par 1 la modalité la moins fréquente
  - 2 personnes ayant la valeur 1 pour le test sont *plus similaires* que 2 personnes ayant 0 pour le test

Variables binaires(II)

- Exemple

Nom	Sexe	Fièvre	Toux	Test-1	Test-2	Test-3	Test-4
Jack	M	Y	N	P	N	N	N
Mary	F	Y	N	P	N	P	N
Jim	M	Y	P	N	N	N	N

- Sexe est un attribut symétrique
- Les autres attributs sont asymétriques
- Y et P ≡ 1, N ≡ 0, la distance n'est mesurée que sur les asymétriques

$$d(jack, mary) = \frac{0+1}{2+0+1} = 0.3$$

$$d(jack, jim) = \frac{1+1}{1+1+1} = 0.67$$

$$d(jim, mary) = \frac{1+2}{1+1+2} = 0.75$$

Les plus similaires sont Jack et Mary⇒atteints du même mal

Variables Nominales

- Une généralisation des variables binaires, ex: rouge, vert et bleu
- Méthode 1: Matching simple
  - *m*: # d'appariements, *p*: # total de variables

- Méthode 2: utiliser un grand nombre de variables binaires
  - Créer une variable binaire pour chaque modalité (ex: variable rouge qui prend les valeurs vrai ou faux)

Variables Ordinales

- Une variable ordinale peut être discrète ou continue
- L'ordre peut être important, ex: classement
- Peuvent être traitées comme les variables intervalles
  - remplacer x_{if} par son rang $r_{if} \in \{1, \ldots, M_f\}$
  - Remplacer le rang de chaque variable par une valeur dans [0, 1] en remplaçant la variable f dans l'objet I par

$$z_{if} = \frac{r_{if} - 1}{M_f - 1}$$

  - Utiliser une distance pour calculer la similarité

En Présence de Variables de différents Types

- Pour chaque type de variables utiliser une mesure adéquate. Problèmes: les clusters obtenus peuvent être différents
- On utilise une formule pondérée pour faire la combinaison

  - *f* est binaire ou nominale:

    $d_{ij}^{(f)} = 0$ si $x_{if} = x_{jf}$, sinon $d_{ij}^{(f)} = 1$
  - *f* est de type intervalle: utiliser une distance normalisée
  - *f* est ordinale
    - calculer les rangs r_{if} et

    - Ensuite traiter z_{if} comme une variable de type intervalle

Approches de Clustering

- Algorithmes de Partitionnement: Construire plusieurs partitions puis les évaluer selon certains critères
- Algorithmes hiérarchiques: Créer une décomposition hiérarchique des objets selon certains critères
- Algorithmes basés sur la densité: basés sur des notions de connectivité et de densité
- Algorithmes de grille: basés sur un structure à multi-niveaux de granularité
- Algorithmes à modèles: Un modèle est supposé pour chaque cluster ensuite vérifier chaque modèle sur chaque groupe pour choisir le meilleur

Algorithmes à partionnement

- Construire une partition à ***k*** clusters d'une base ***D*** de ***n*** objets
- Les *k clusters doivent* optimiser le critère choisi
  - Global optimal: Considérer toutes les k-partitions
  - Heuristic methods: Algorithmes *k-means* et *k-medoids*
  - *k-means* (MacQueen'67): Chaque cluster est représenté par son centre
  - *k-medoids* or PAM (Partition around medoids) (Kaufman & Rousseeuw'87): Chaque cluster est représenté par un de ses objets

La méthode des k-moyennes (*K-Means)*

- L'algorithme *k-means* est en 4 étapes :

1. Choisir k objets formant ainsi k clusters

1. (Ré)affecter chaque objet O au cluster C_i de centre M_i tel que dist(O,M_i) est minimal

1. Recalculer M_i de chaque cluster (le barycentre)

1. Aller à l'étape 2 si on vient de faire une affectation

K-Means :Exemple

A={1,2,3,6,7,8,13,15,17}. Créer 3 clusters à partir de A

On prend 3 objets au hasard. Supposons que c'est 1, 2 et 3. Ca donne C_1={1}, M_1=1, C_2={2}, M_2=2, C_3={3} et M_3=3

Chaque objet O est affecté au cluster au milieu duquel, O est le plus proche. 6 est affecté à C_3 car dist(M_3,6)<dist(M_2,6) et dist(M_3,6)<dist(M_1,6)

On a C_1={1}, M_1=1,

C_2={2}, M_2=2

C_3={3, 6,7,8,13,15,17}, M3=69/7=9.86

K-Means :Exemple (suite)

- dist(3,M2)<dist(3,M3)➔3 passe dans C_2. Tous les autres objets ne bougent pas. C_1={1}, M_1=1, C_2={2,3}, M_2=2.5,C_3={6,7,8,13,15,17} et M_3= 66/6=11

- dist(6,M_2)<dist(6,M_3)➔6 passe dans C2. Tous les autres objets ne bougent pas. C_1={1}, M_1=1, C_2={2,3,6}, M_2=11/3=3.67, C3={7,8,13,15,17}, M_3= 12

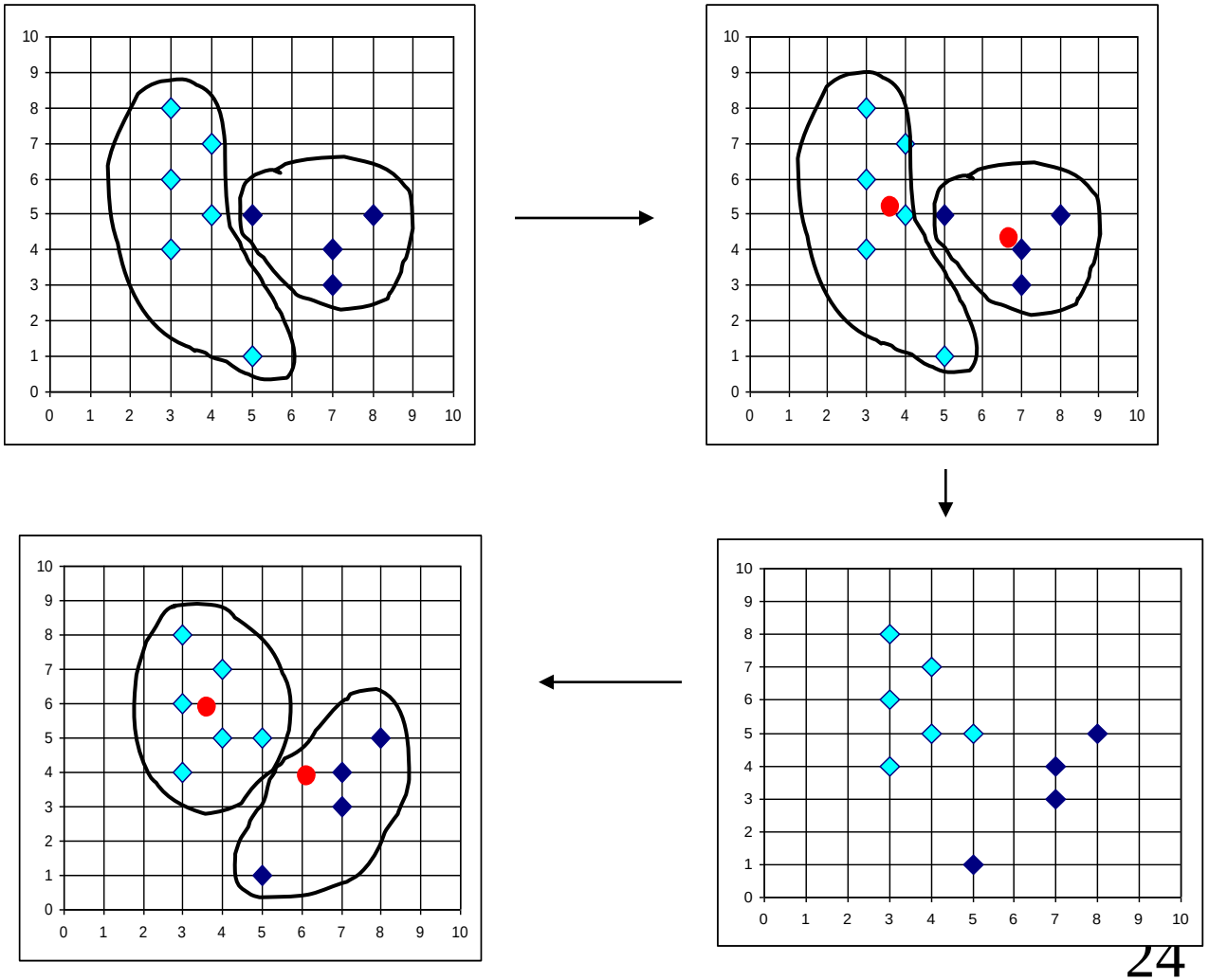
- dist(2,M1)<dist(2,M2)➔2 passe en C1. dist(7,M2)<dist(7,M3)➔ 7 passe en C2. Les autres ne bougent pas. C1={1,2}, M1=1.5, C2={3,6,7}, M2=5.34, C3= {8,13,15,17}, M3=13.25

- dist(3,M1)<dist(3,M2)➔3 passe en 1. dist(8,M2)<dist(8,M3)➔8 passe en 2 C1={1,2,3}, M1=2, C2={6,7,8}, M2=7, C3={13,15,17}, M3=15

Plus rien ne bouge

Algorithme *K-Means*

- Exemple



Commentaires sur la méthode des *K-Means*

- Force
  - *Relativement efficace*: $O(tkn)$, où n est # objets, k est # clusters, et t *est* # itérations. Normalement, k, $t << n$.
  - Tend à réduire

- Faiblesses
  - N'est pas applicable en présence d'attributs qui ne sont pas du type intervalle (moyenne=?)
  - On doit spécifier k (nombre de clusters)
  - Les clusters sont construits par rapports à des objets inexistants (les milieux)
  - Ne peut pas découvrir les groupes *non-convexes*

La méthode des *K-Medoids (PAM)*

- Trouver des objets représentatifs (medoïdes) dans les clusters (au lieu de la moyenne)
- Principe
  - Commencer avec un ensemble de medoïdes puis itérativement remplacer un par un autre si ça permet de réduire la distance globale
  - Efficace pour des données de petite taille

Algorithme des k-Medoides

Choisir arbitrairement k medoides

Répéter

affecter chaque objet restant au medoide le plus proche

Choisir aléatoirement un non-medoide O_r

Pour chaque medoide O_j

Calculer le coût TC du remplacement de O_j par O_r

Si TC < 0 alors

Remplacer O_j par O_r

Calculer les nouveaux clusters

Finsi

FinPour

Jusqu'à ce ce qu'il n'y ait plus de changement

PAM (Partitioning Around Medoids) (1987)

Choisir arbitrairement ***k*** objets représentatifs

- Pour toute paire (h,j) d'objets t.q h est choisi et j non, calculer le coût $\boldsymbol{TC_{jh}}$ du remplacement de j par h
  - Si $TC_{ih} < 0$, ***j*** est remplacé par ***h***
  - Puis affecter chaque objet non sélectionné au medoïde qui lui est le plus similaire
- Répéter jusqu'à ne plus avoir de changements

La méthode des *K-Medoids*

- TC_{jh} représente le gain en distance globale que l'on va avoir en remplaçant h par j

- Si TC_{jh} est négatif alors on va perdre en distance. Ca veut dire que les clusters seront plus compacts.

- $TC_{jh}=\Sigma_i$ dist(j,h)-dist(j,i)= $\Sigma_i C_{ijh}$

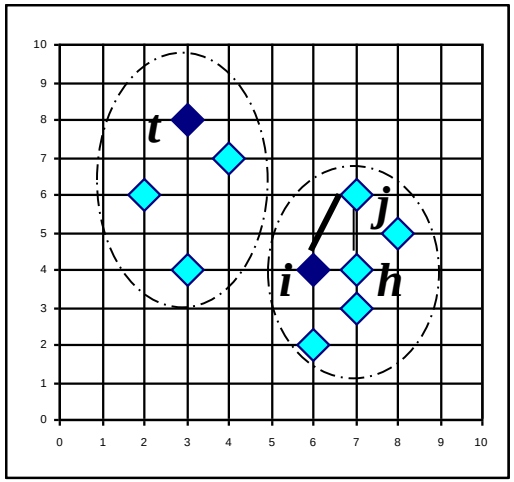
La méthode des *K-Medoids:* Exemple

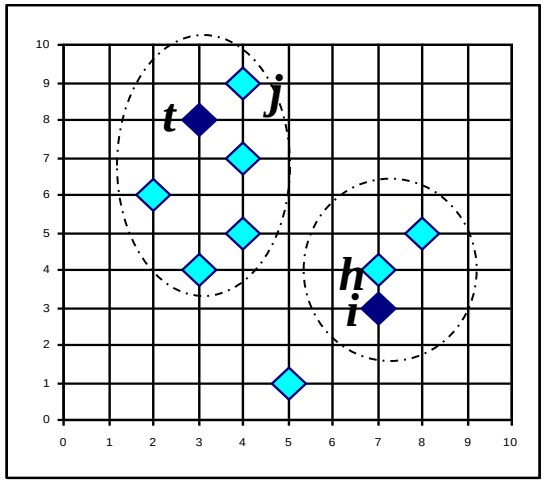
- Soit A={1,3,4,5,8,9}, k=2 et M={1,8} ensemble des medoides
  ➔C1={1,3,4} et C2={5,8,9}
  $E_{\{1,8\}}$=dist(3,1)^2+dist(4,1)^2+dist(5,8)^2+dist(5,9)^2+dist(9,8)^2=39

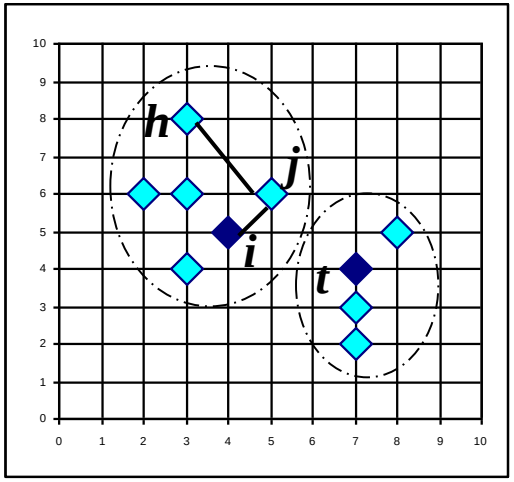
- Comparons 1 et 3➔M={3,8}➔C1={1,3,4,5} et C2={8,9}
  $E_{\{3,8\}}$ =dist(1,3)^2+dist(4,3)^2+dist(5,3)^2+dist(9,8)^2=10
  $E_{\{3,8\}}$ - $E_{\{1,8\}}$= -29 <0 donc le remplacement est fait.

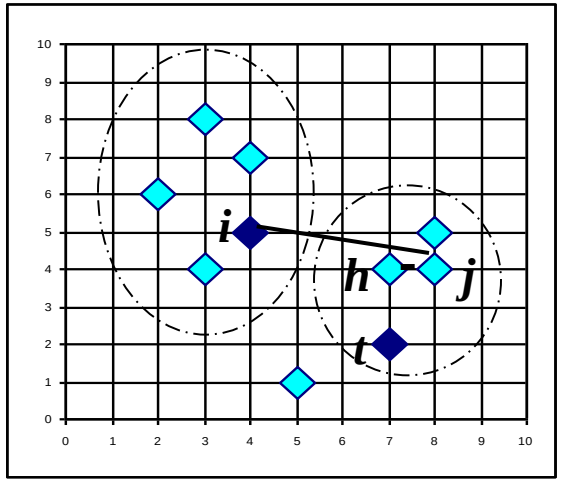
- Comparons 3 et 4➔ M={4,8}➔ C1 et C2 inchangés et $E_{\{4,8\}}$=dist(1,4)^2+dist(3,4)^2+dist(5,4)^2+dist(8,9)^2= 12➔ 3 n'est pas remplacé par 4
- Comparons 3 et 5➔M={5,8}➔ C1 et C2 inchangés et E{5,8}>E{3,8}

PAM Clustering: $TC_{ih}=\sum_j C_{jih}$



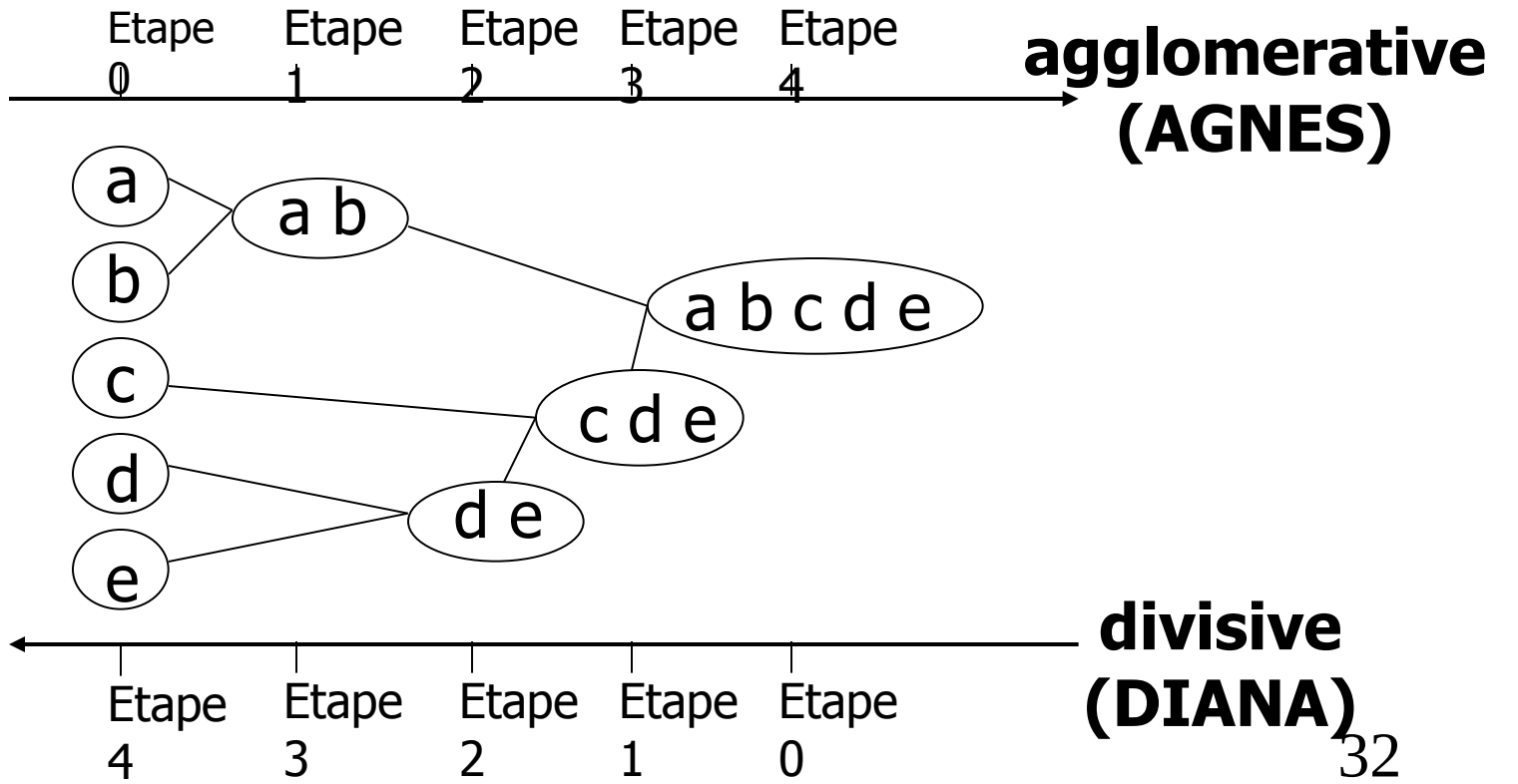






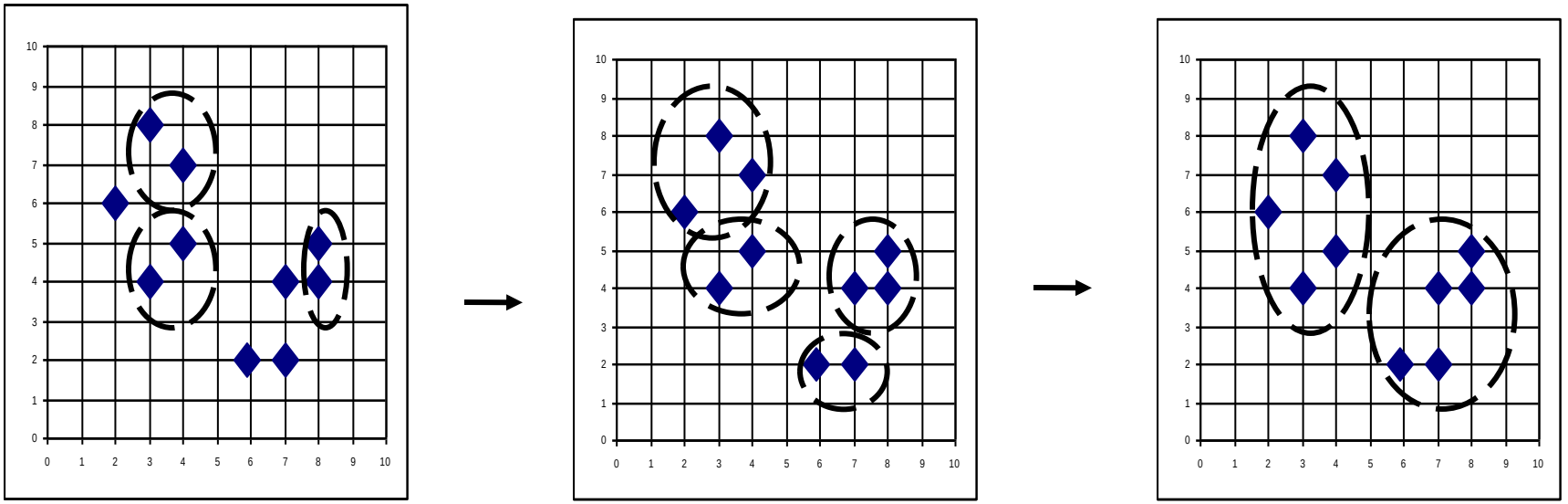
Clustering Hiérarchique

- Utiliser la matrice de distances comme critère de regroupement. ***k*** n'a pas à être précisé, mais a besoin d'une condition d'arrêt



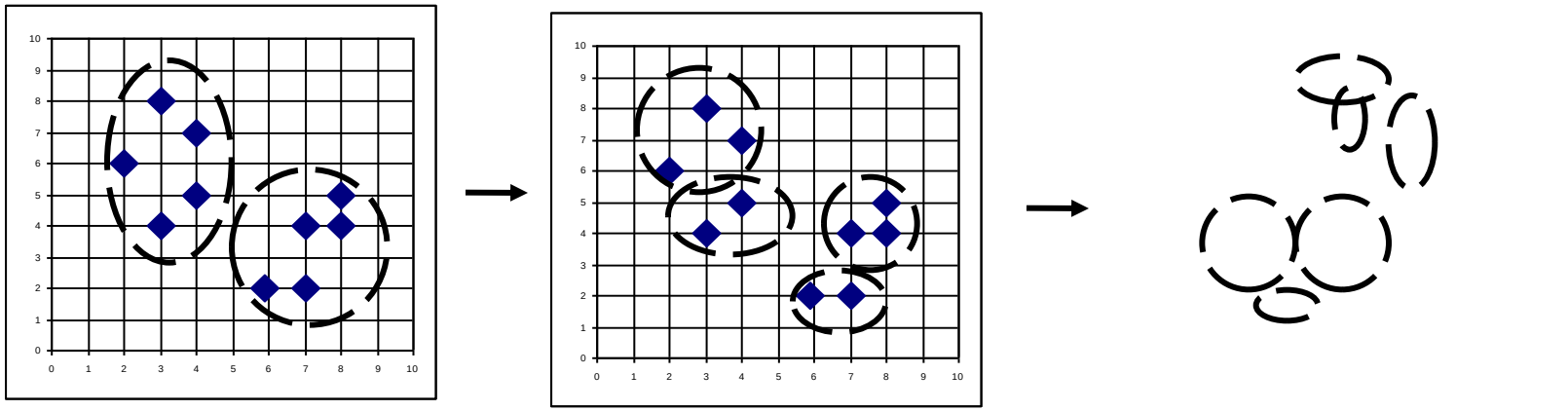
AGNES (Agglomerative Nesting)

- Utilise la matrice de dissimilarité.
- Fusionne les nœuds qui ont la plus faible dissimilarité
- On peut se retrouver dans la situation où tous les nœuds sont dans le même groupe



DIANA (Divisive Analysis)

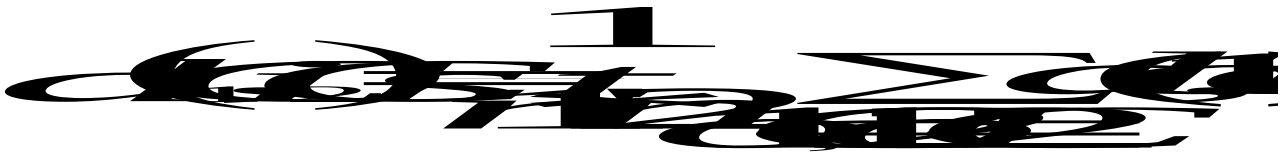
- L'ordre inverse de celui d'AGNES
- Il se peut que chaque objet forme à lui seul un groupe



Critères de fusion-éclatement

- Exemple: pour les méthodes agglomératives, C1 et C2 sont fusionnés si
  - il existe o1 $\in$ C1 et o2$\in$ C2 tels que dist(o1,o2) $\leq$ seuil, ou
  - il n'existe pas o1 $\in$ C1 et o2$\in$ C2 tels que dist(o1,o2) $\geq$ seuil, ou
  - distance entre C1 et C2 $\leq$ seuil avec

Lien unique



et n1=|C1|.

- Ces techniques peuvent être adaptées pour les méthodes divisives

BIRCH (1996)

- Birch: Balanced Iterative Reducing and Clustering using Hierarchies
- Construit incrémentalement un arbre (CF-tree : Clustering Feature), une structure hiérarchique où chaque niveau représente une phase de clustering
  - Phase 1: scanner la base pour construire le CF-tree dans la mémoire
  - Phase 2: utiliser n'importe quel algorithme de clustering sur les feuilles du CF-tree
- *Avantage*: trouve les clusters en une seule passe sur la BD
- *Inconvénient:* ne considère que les données numériques et est sensible à l'ordre des enregistrements

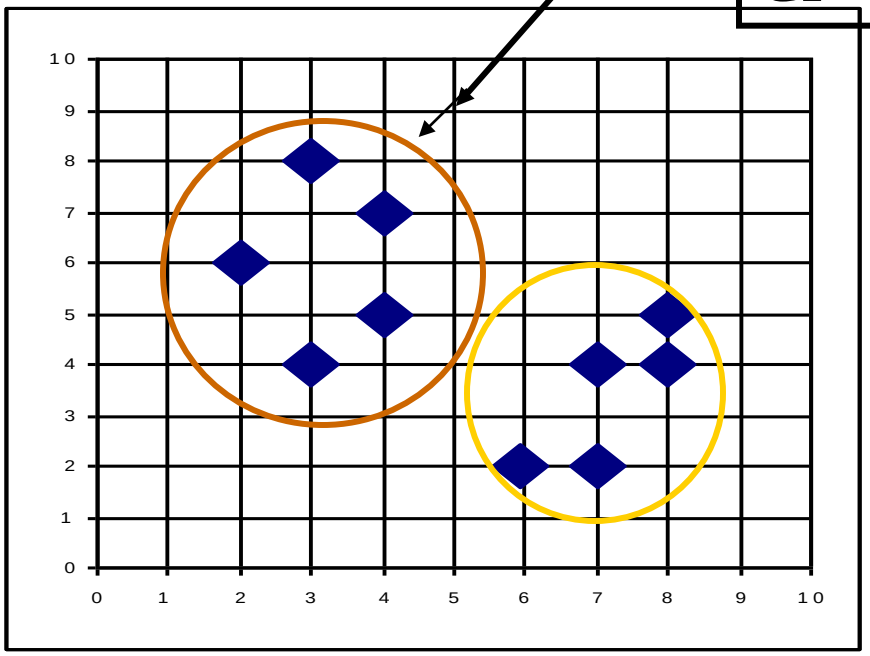
Clustering Feature Vector

Clustering Feature: $CF = (N, \overrightarrow{LS}, SS)$

N: **Number of data points**

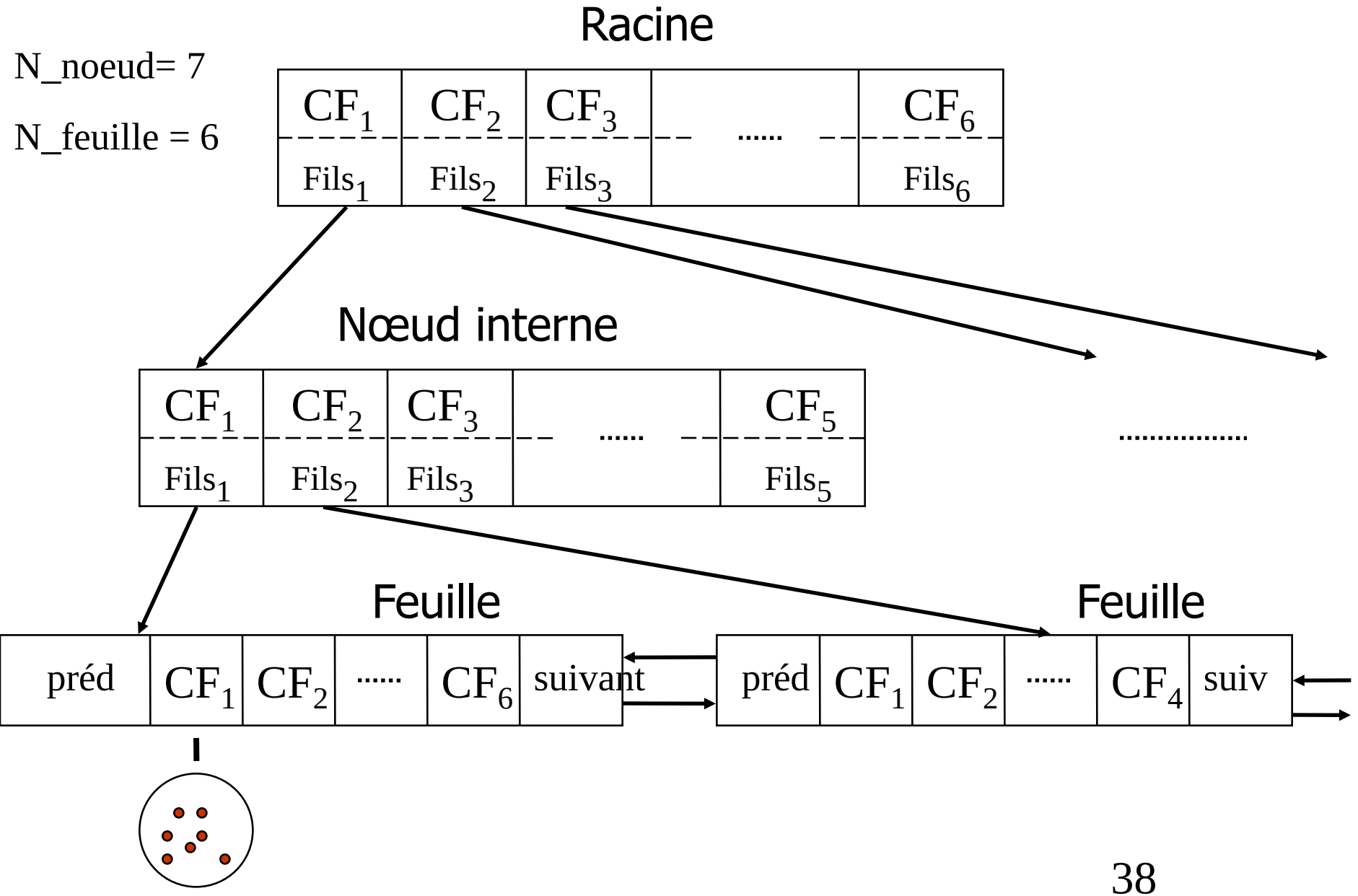
$LS: \sum_{i=1}^{N} \overrightarrow{X_i}$

$SS: \sum_{i=1}^{N} \overrightarrow{X_i^2}$

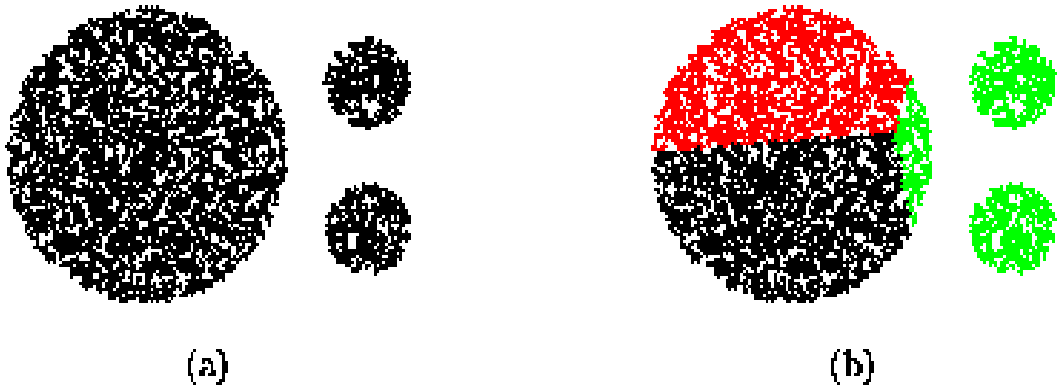


(3,4)
(2,6)
(4,5)
(4,7)
(3,8)

CF Tree


Racine
N_noeud= 7
N_feuille = 6
CF1 CF2 CF3 ..... CF6
Fils1 Fils2 Fils3 Fils6
Nœud interne
CF1 CF2 CF3 ..... CF5
Fils1 Fils2 Fils3 Fils5
Feuille
Feuille
préd CF1 CF2 ..... CF6 suivant
préd CF1 CF2 ..... CF4 suiv

CURE (Clustering Using REpresentatives)



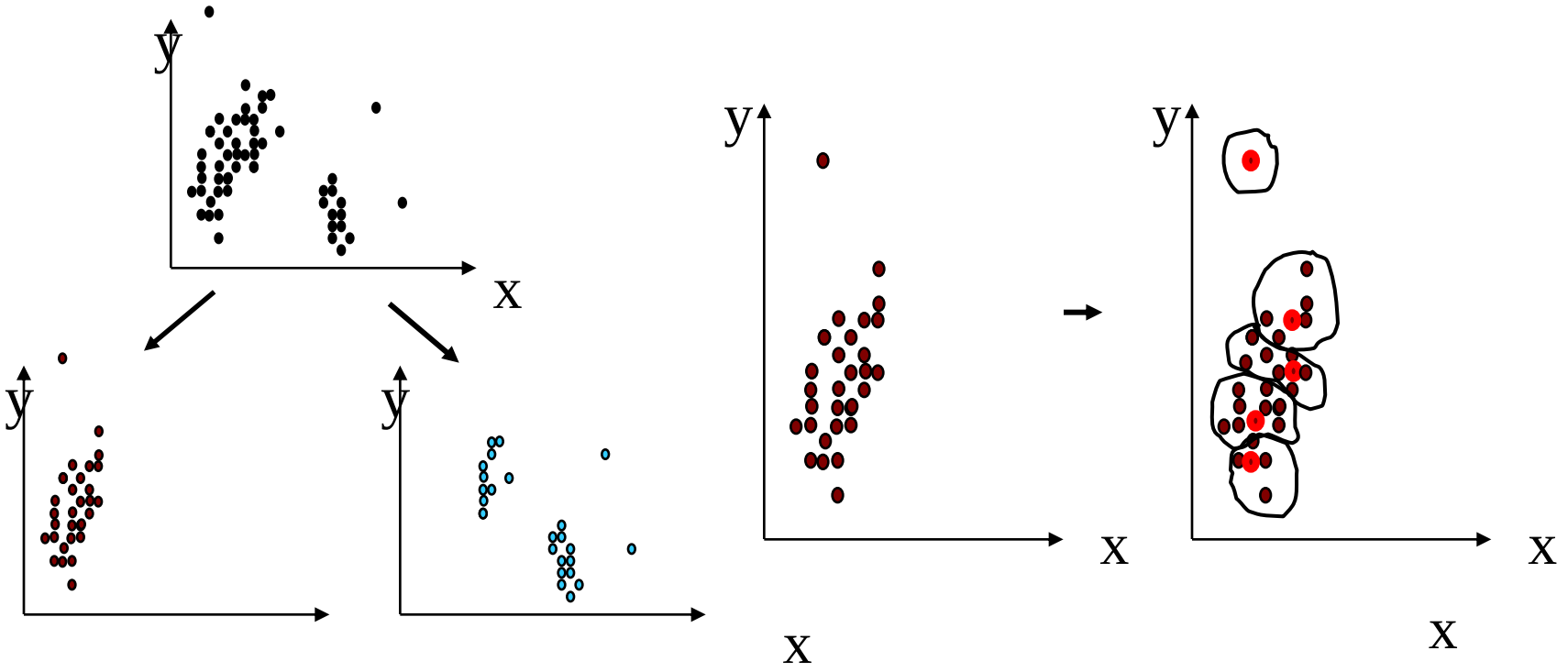
- Les méthodes précédentes donnent les groupes (b)
- CURE: (1998)
  - Arrête la création de clusters dès qu'on en a *k*
  - Utilise plusieurs points représentatifs clusters

Cure: l'algorithme

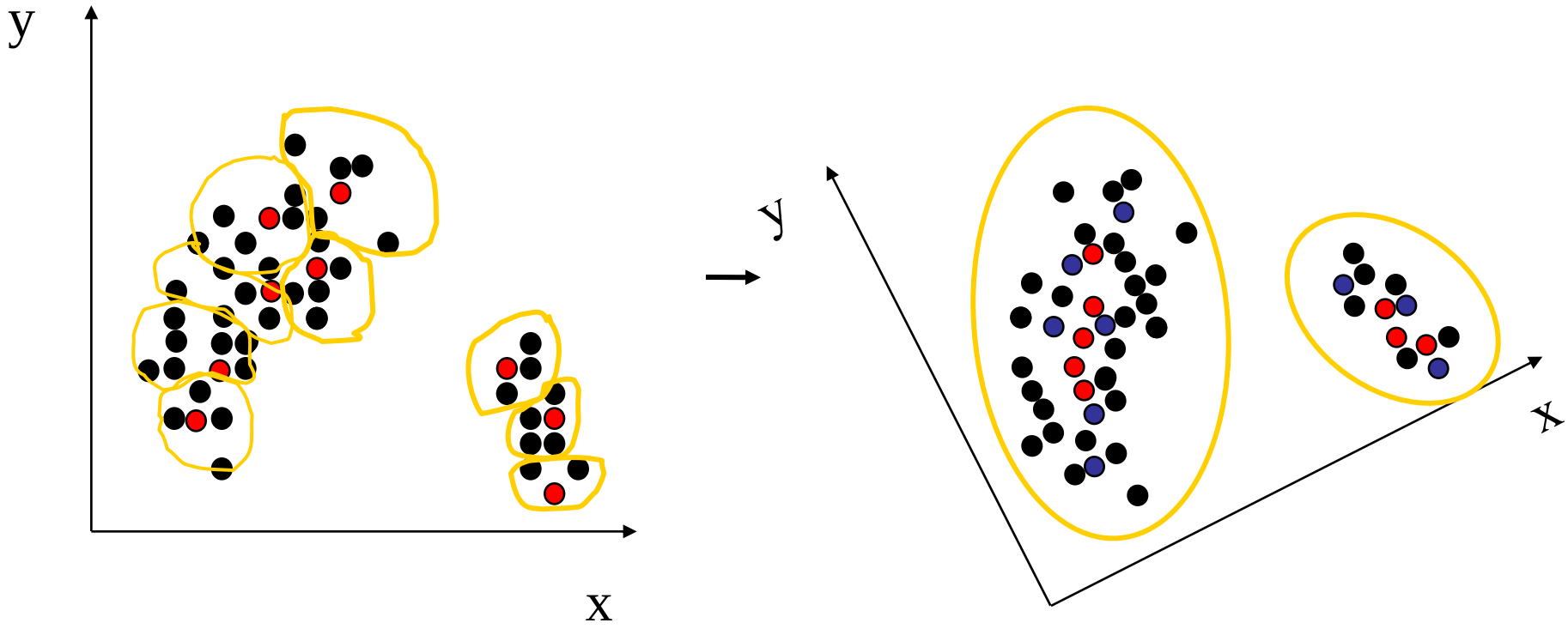
- Prendre un sous-ensemble *s*
- Partitionner s en *p* partitions de taille *s/p*
- Dans chaque partition, créer *s/pq* clusters
- Eliminer les exceptions (points aberrants)
- Regrouper les clusters partiels

Partitionnment et Clustering

- s = 50
- p = 2
- s/p = 25
- s/pq = 5



Cure: Rapprochement des points représentatifs

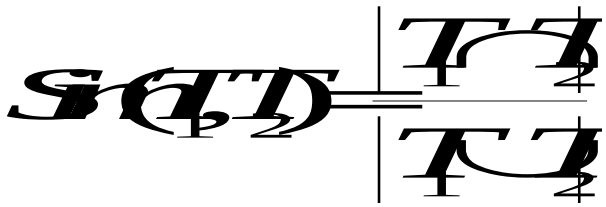


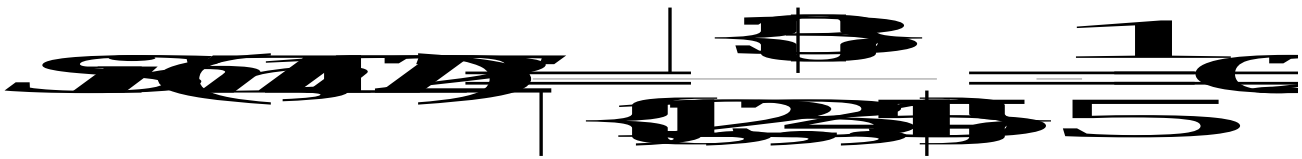
- Rapprocher les points représentatifs vers le centre de gravité par un facteur α.
- Plusieurs points représentatifs permettent de figurer la forme du cluster

Clustering de données Catégorielles : ROCK

- ROCK: Robust Clustering using linKs
  - Utilise les liens pour mesurer la similarité/proximité
  - N'est pas basé sur la notion de distance
- Idée :
  - Fonction de similarité et voisins:

    Let T_1 = {1,2,3}, T_2={3,4,5}





Rock

- Considérons 4 transactions et 6 produits t.q
  T1={1,2,3,5} T2={2,3,4,5}
  T3={1,4} et T4={6}

- T1 peut être représentée par {1,1,1,0,1,0}

dist(T1,T2)=2 qui est la plus petite distance entre 2 transactions ➔ T1 et T2 dans même cluster. La moyenne de C1=(0.5,1,1,0.5,1,0).

C2={T3,T4} car dist(T3,T4)=3. Or T3 et T4 n'ont aucun produit en commun !

Idée : se baser sur le nombre d'éléments en commun

Ce n'est pas suffisant {1,2} est plus proche de {1,2,3} que de {1,2,3,4,5,6}

Rock: l'algorithme

- Liens: Le nombre de voisins communs de 2 points

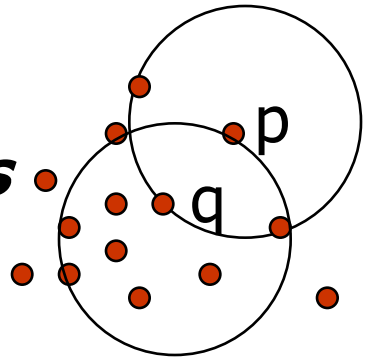
{1,2,3}, {1,2,4}, {1,2,5}, {1,3,4}, {1,3,5}
{1,4,5}, {2,3,4}, {2,3,5}, {2,4,5}, {3,4,5}

{1,2,3} ⟷ (3) {1,2,4}

- Algorithme
  - Prendre un sous ensemble
  - Regrouper avec les liens

Clustering basé sur la densité

- Voit les clusters comme des régions denses séparées par des régions qui le sont moins (bruit)
- Deux paramètres***:***
  - ***Eps***: Rayon maximum du voisinage
  - ***MinPts***: Nombre minimum de points dans le voisinage-Eps d'un point
- ***Voisinage : $V_{Eps}(p)$***: ***$\{q \in D \mid dist(p,q) <= Eps\}$***
- Un point ***p*** est directement densité-accessible à partir de ***q*** resp. à ***Eps***, ***MinPts*** si
  - 1) ***$p \in V_{Eps}(q)$***
  - 2) $|V_{Eps}(q)| >= MinPts$

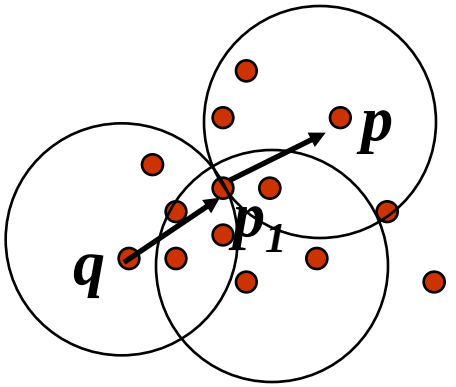


MinPts = 5

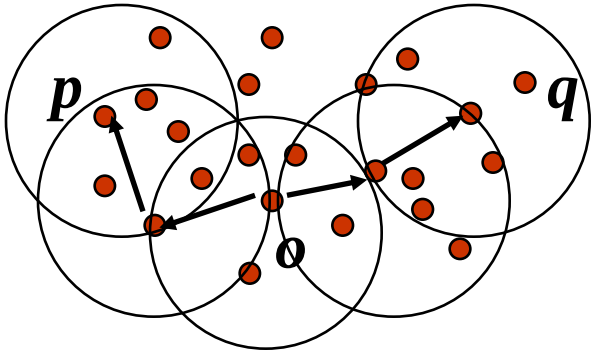
Eps = 1 cm

Clustering basé sur la densité

- Accessibilité:
  - *p* est accessible à partir de *q* resp. à *Eps*, *MinPts* si il existe p_1, …, p_n, $p_1 = q$, $p_n = p$ t.q p_{i+1} est directement densité accessible à partir de p_i

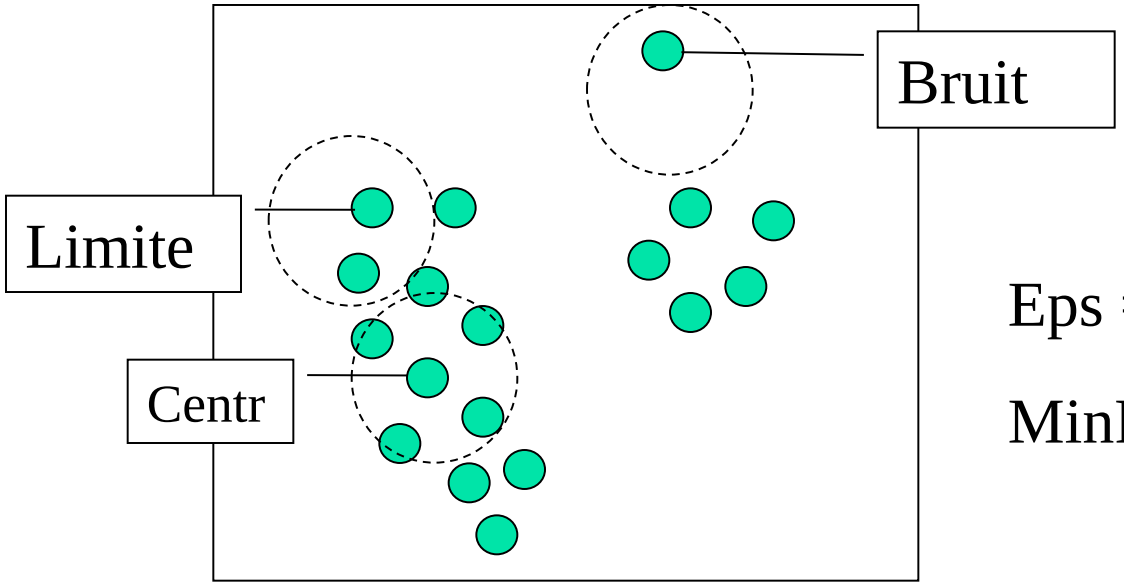


- Connexité
  - *p* est connecté à *q* resp. à *Eps*, *MinPts* si il existe un point *o* t.q *p* et *q* accessibles à partir de *o* resp. à *Eps* et *MinPts*.



DBSCAN: Density Based Spatial Clustering of Applications with Noise

- Un *cluster* est l'ensemble maximal de points connectés
- Découvre des clusters non nécessairement convexes



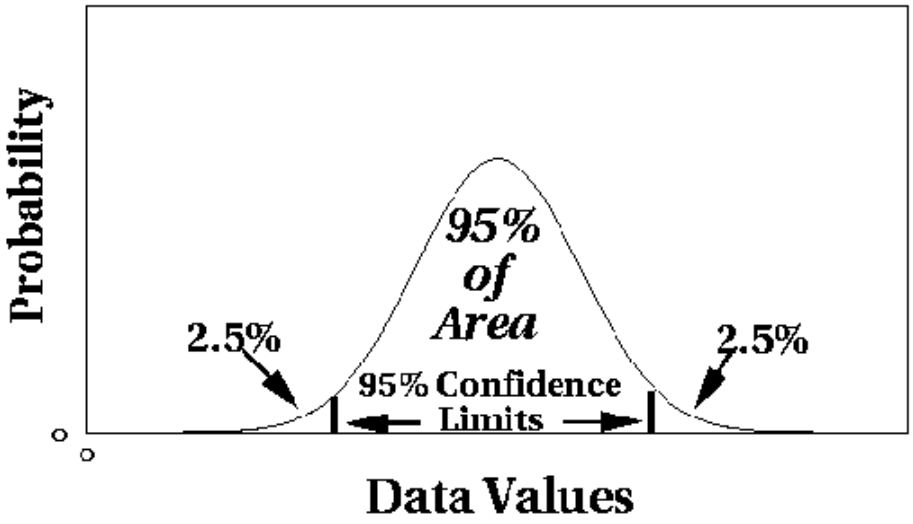
DBSCAN: l'algorithme

- Choisir ***p***
- Récupérer tous les points accessibles à partir de ***p*** resp. ***Eps*** et ***MinPts***.
- Si ***p*** est un centre, un cluster est formé.
- si ***p*** est une limite, alors il n'y a pas de points accessibles de ***p*** : passer à un autre point
- Répéter le processus jusqu'à épuiser tous les points.

Découverte d'exceptions

- Ce sont les objets qui sont considérablement différents du reste, exemple: ornithorynque, kiwi
- Problème
  - Trouver n objets qui sont les plus éloignés du reste
- Applications:
  - fraude
  - Analyse médicale
  - …

Approache statistique



- On suppose que les données suivent une loi de distribution statistique (ex: loi normale)
- Utiliser les tests de discordance
  - Proba(Xi=val)< β alors X est une exception
- Problèmes
  - La plupart des tests sont sur un attribut
  - Dans beaucoup de cas, la loi de distribution est inconnue

Approche Basée sur la Distance

- Une (α, β)-exception est un object O dans T tel qu'il y a au moins α objets O' de T avec dist(O,O')> β