Techniques de Classification

© LOTFI BEN ROMDHANE, *PH.D.*

ISITCOM/U. DE SOUSSE/ TN

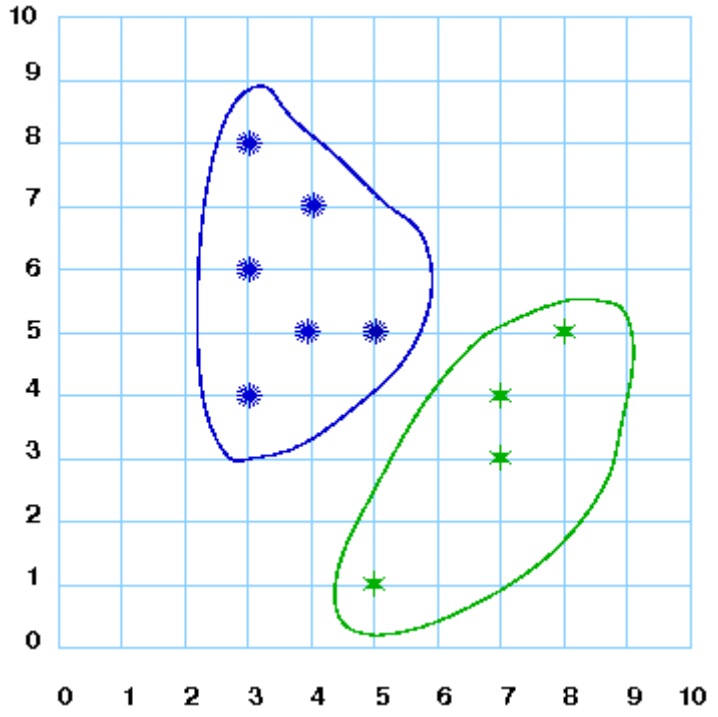
3DNI

Sommaire

- Concepts de base
- Mesures de Distance
- Classification hiérarchique
- Algorithme K-Moyenne
- Algorithme K-Medoides
- Discussion

Concepts de Base (1)

- Plusieurs noms: *clustering, classification, classement*, ...
- Un ensemble de méthodes permettant de créer des ***groupes*** à partir d'un ensemble de données tels que:
  - les données qui appartiennent au *même groupe* sont *similaires*
  - les données appartenant à des *groupes différents* sont *dissimilaires*

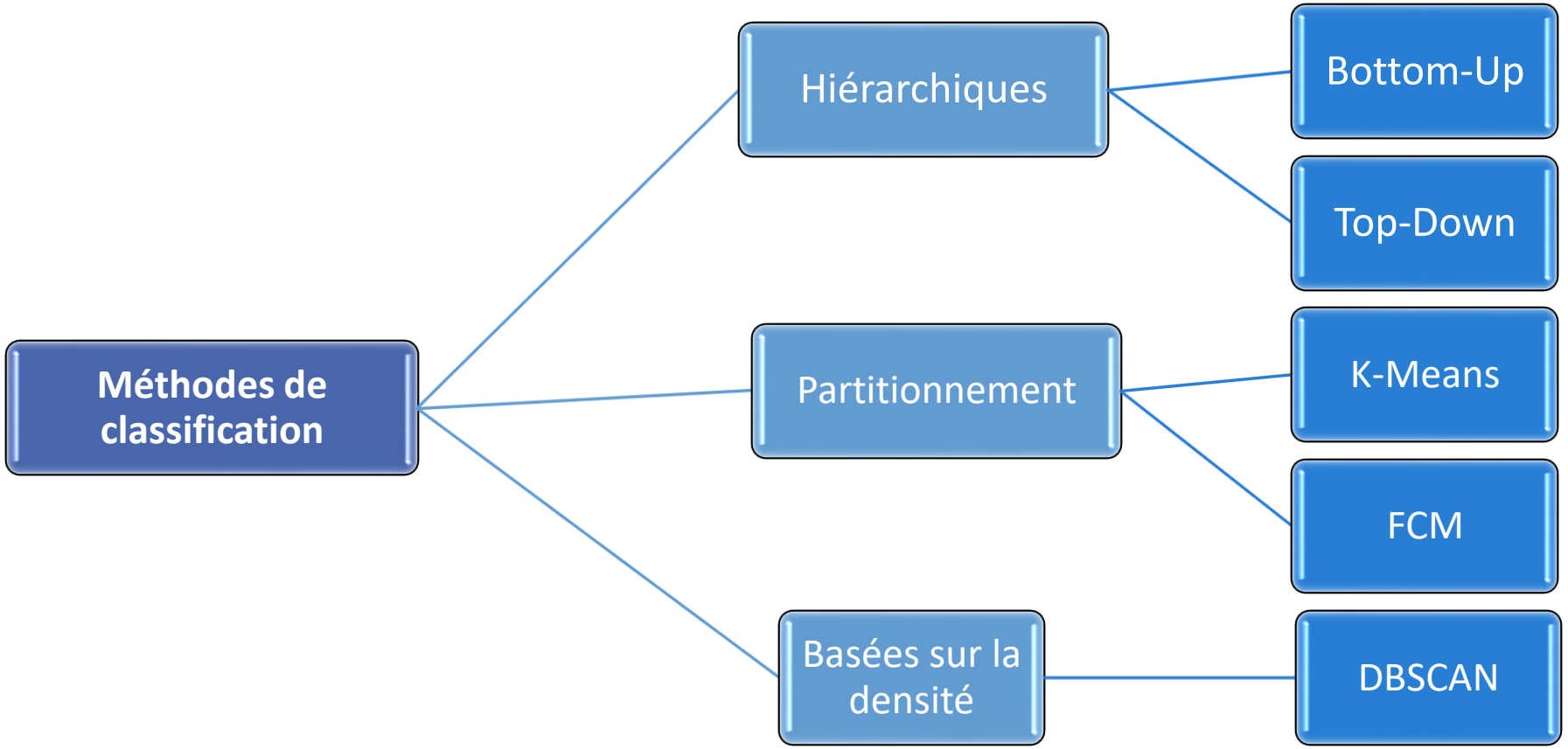


Concepts de Base (2)

- Plusieurs applications en pratique
- **Les données de puces ADN**
  - les gènes qui sont similaires admettent dans l'organisme des fonctions identiques
- **Compression des données**
  - Les données qui se trouvent dans un groupe sont similaires et peuvent être remplacées par une seule valeur
- **Segmentation des Images**: diviser une image en un ensemble de régions homogènes
  - *Compression des images pour la transmission*: représenter chaque région par un seul point (pixel)
  - *Reconnaissance des formes*: appariement d'une forme (par exemple le visage d'une personne) avec les régions de l'image (ensembles des visages dans l'image)

Concepts de Base (3)

Approches existantes



Formulation du problème

- On considère $D = \{X_1, X_2, \ldots, X_n\}$ un ensemble de n points (données)
- $Xi = \{x_{i,1}, x_{i,2}, \ldots, x_{i,l}\}$ une donnée vecteur de dimension ***l***, c'est-à-dire possédant l attributs participant à sa définition
- La classification consiste à trouver un ensemble de groupes {C1,C2, …, Cc }; dits aussi clusters; tel que la distance entre des données du même groupe est plus petite que la distance entre deux données appartenant à deux groupes différents.
- Ces groupes peuvent être
  - **disjointes**: chaque donnée appartient à un et un seul groupe
  - **chevauchantes** : chaque donnée peut appartenir à plus d'un groupe

Fonction de distance (1)

- Une **distance** défini sur un ensemble E est une **fonction** : d : E×E→ℜ+ possédant les propriétés suivantes
  - **Symétrie**

$$\forall x, y \in E; d(x, y) = d(y, x)$$

  - **Séparabilité**

$$\forall x, y \in E; d(x, y) = 0 \Leftrightarrow x = y$$

  - **Inégalité triangulaire**

$$\forall x, y, z \in E; d(x, z) \leq d(x, y) + d(y, z)$$

Fonction de distance (2)

- **Distance Euclidienne**

$$dist(X_i, X_j) = \sqrt{\sum_{k=1}^{l} (x_{i,k} - x_{j,k})^2}$$

- **Distance de Manhattan**

$$dist(X_i, X_j) = \sum_{k=1}^{l} |x_{i,k} - x_{j,k}|$$

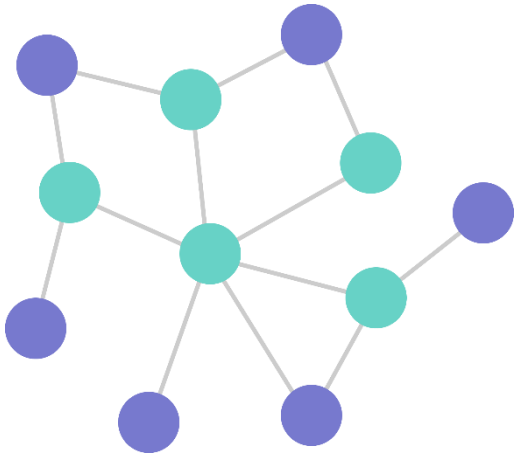
- **Distance de Jaccard (pour les ensembles)**

$$dist(A, B)^2 = \frac{|A \cup B| - |A \cap B|}{|A \cup B|}$$

- **Distance Edit**
  - *nombre d'opérations élémentaires* (insertions, suppression, …)

  pour changer un objet (chaîne de caractères, graphe) vers un autre

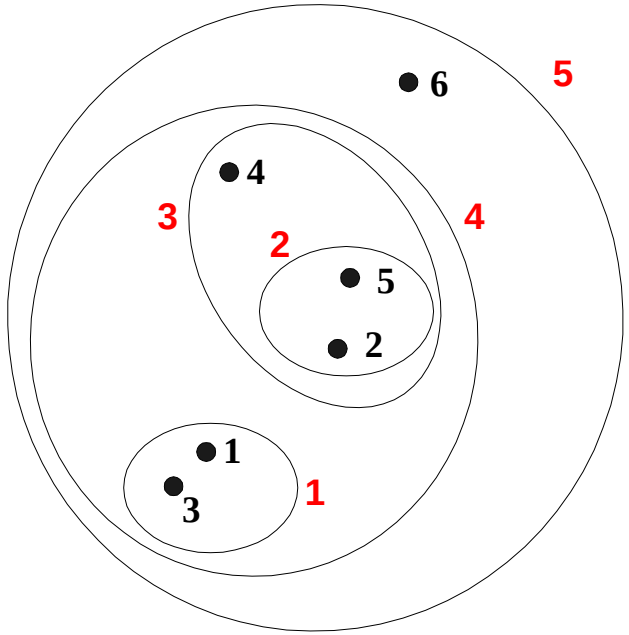
String1 = « ABCD »
String2= « ABDE »

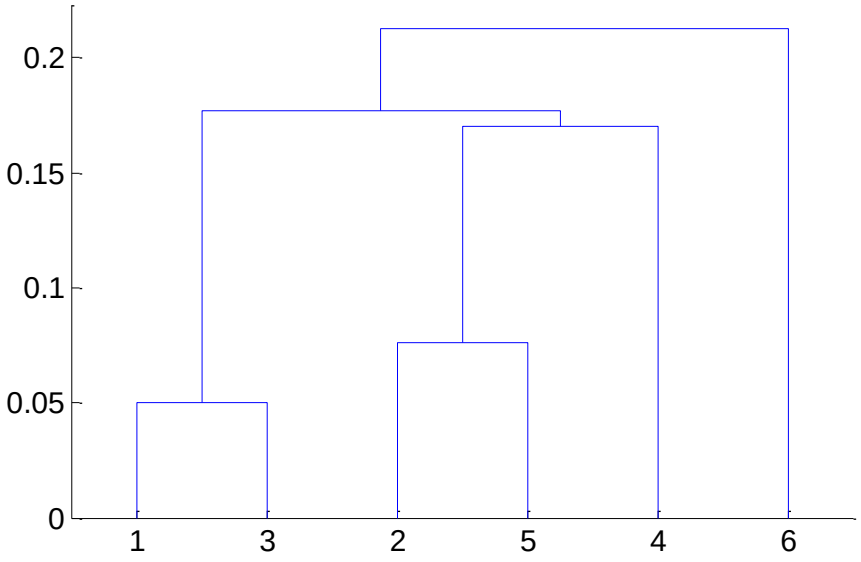


Classification hiérarchique

Notions de base (1)

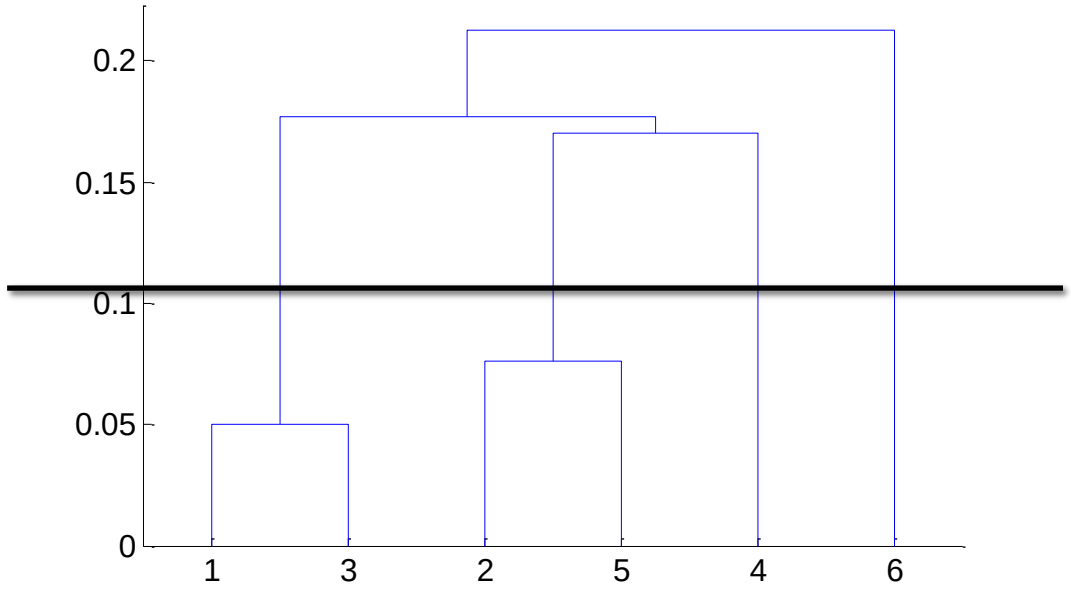
- La classification hiérarchique permet de produire une **hiérarchie** de groupes
  - chaque niveau de la hiérarchie représente un partitionnement différent
  - peut-être modélisé à l'aide d'un **arbre hiérarchique appelé dendrogramme**





Notions de base (2)

- La coupure du dendrogramme à un niveau particulier génère un partitionnement différent



Partitionnement = { {x1, x3} ; {x2, x5} ; {x4} ; {x6} }

Question 1: Comment générer cet arbre à partir d'un ensemble de données ?

Question 2: Comment déterminer la coupure optimale ?

Génération du dendrogramme (1)

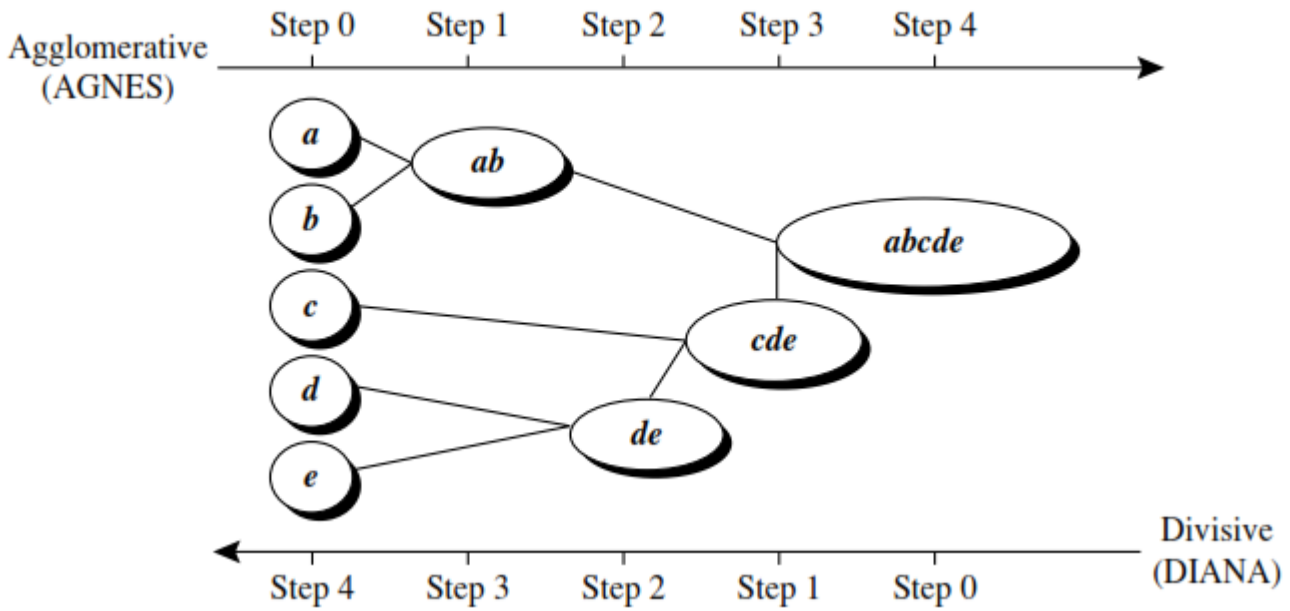
APPROCHE ASCENDANTE (BOTTOM-UP)

- dite aussi **agglomérative**
- partir d'un partitionnement dans lequel chaque donnée forme un groupe à part
- à chaque étape (niveau du dendrogramme), fusionner les deux groupes les plus proches
- s'arrêter lorsqu'on obtient un seul groupe

APPROCHE DESCENDANTE (TOP-DOWN)

- dite aussi **divisive**
- partir avec un seul groupe contenant toutes les données
- à chaque étape, diviser le groupe le plus mauvais
- s'arrêter lorsque chaque donnée est dans un groupe à part

Génération du dendrogramme (2)



Méthodes descendantes (1)

- Il y a plusieurs méthodes de classification qui diffèrent par le choix du groupe à diviser
  - DAA : Deterministic Annealing Algorithms
  - SPC : Super-Paramagnetic Clustering
  - SOTA : Self-Organizing Tree Algorithm
  - DIANA : Divisive ANAlysis

Algorithme de base

DEBUT

- mettre toutes les données dans un seul groupe
- **Répéter**
  - diviser en deux le groupe le plus mauvais
    - pour cela on peut utiliser un autre algorithme tel que K-moyenne
  - mettre à jour la matrice des distances entre les groupes
- **Jusqu'à (chaque donnée est dans un groupe à part)**

FIN.

Méthodes descendantes (3)

- On a besoin d'un critère pour choisir le groupe à diviser
  - **le plus large** : admettant les points les plus éloignés (plus distants)
  - **le plus dense** : contenant le maximum de points
- Pour diviser ***un groupe de n éléments*** en deux, il y a un grand nombre de possibilités à tester :

$$\left(2^{n-1}-1\right)$$

- On doit utiliser une **heuristique** pour réduire le nombre de possibilités

Méthodes descendantes (4)

DIANA (DIvisive ANAlysis)

- on divise le groupe admettant le ***plus grand rayon***

Méthode de division d'un groupe G

1. trouver P_0, le point admettant la plus *grande dissimilarité moyenne (distance)* avec le reste des points du groupe *G*
2. P_0 va initier la formation d'un nouveau groupe $G_{new} = \{P_0\}$
3. Pour chaque point $i \notin G_{new}$ calculer

$$Diff_i = \overline{d(i,j)}/j \notin G_{new} - \overline{d(i,j)}/j \in G_{new}$$

-

- Cette formule estime la différence entre la distance moyenne d'une donnée i par rapport à « l'ancien groupe » et la distance moyenne de la même donnée au « nouveau groupe »

- $Diff_i > 0$: la donnée est plus proche du nouveau groupe que de l'ancien et inversement

Méthodes descendantes (5)

4. Trouver un point h / D_h =max $\{D_i\}$ ET D_h >0
   - Le point h est plus proche du nouveau groupe G_{new} que du reste des points de G
   - $G_{new} \leftarrow G_{new} \cup \{h\}$; $G \leftarrow G \setminus \{h\}$
5. Répéter les étapes 3) et 4) jusqu'à ce que tous les points i de G admettent un $D_i < 0$

- on vient de terminer la division du groupe initial G en deux groupes
- Maintenant, on devra reprendre le même processus avec le plus mauvais groupe

Méthodes descendantes (6)

- Considérons un ensemble de villes italiennes et la matrice des distances entre elles

Initialement: G_{new} = {}; G_{old} = {BA, FI, MI, VO, RM, TO}

	BA	FI	MI	VO	RM	TO
BA	0	662	877	255	412	996
FI	662	0	295	468	268	400
MI	877	295	0	754	564	138
VO	255	468	754	0	219	869
RM	412	268	564	219	0	669
TO	996	400	138	869	669	0
Moyenne	533,6	348,8	438	427,5	355,3	512

G_{new} = {BA}
G_{old} = {FI, MI, VO, RM, TO}

Méthodes descendantes (7)

	FI	MI	VO	RM	TO
D(Gnew)	662	877	255	412	996
D(G\Gnew)	357,75	437,75	577,5	430	519
D_i	-304,25	-439,25	322,5	18	-477

G_{new} = {BA; VO}; G_{old} = {FI, MI, RM, TO}

	FI	MI	RM	TO
Gnew	565	815,5	315,5	932,5
G\Gnew	321	332,3	500,3	402,3
D_i	-244	-483,16	184,83	-530,16

G_{new} = {BA; VO; RM}; G_{old} = {FI, MI, TO}

	FI	MI	TO
Gnew	466	731,66	844,66
G\Gnew	347,5	216,5	269
D_i	-118,5	-515,16	-575,66

Tous les D_i sont négatifs : arrêt de la division et on obtient les deux groupes

- G_1 = { FI; MI; TO }
- G_2= { BA; VO; RM }

- *Rayon*(G_1)= distance(FI; TO)= 400
- *Rayon*(G_2) = distance(BA,RM) = 412

Prendre G2 et le diviser selon le même principe ...

Méthodes ascendantes (1)

- Le calcul des distances entre les groupes est un point central qui permet de distinguer la majorité des algorithmes existants
  - UPGMA (Unweighted Pair Group Method with Arithmetic Mean)
  - CHAMELEON
  - ROCK (RObust Clustering using linKs)
  - CURE (Clustering Using REpresentatives)

Algorithme de base

DEBUT

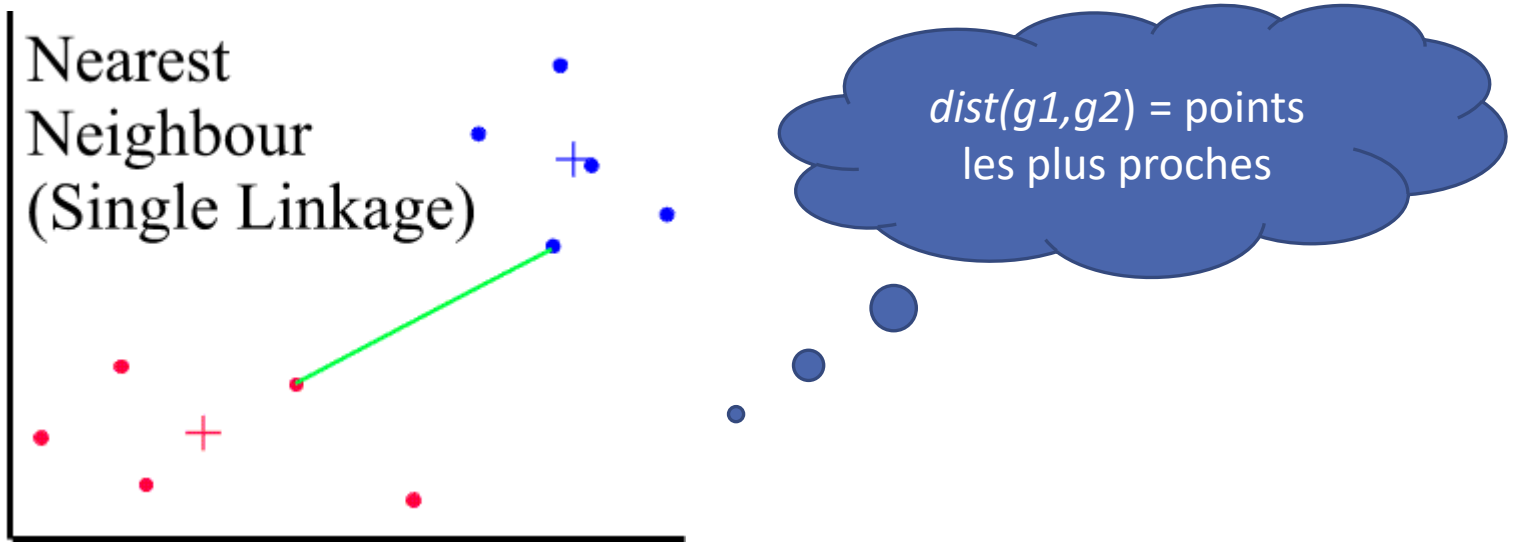
- mettre chaque donnée dans un groupe à part
- **Répéter**
- fusionner les deux groupes les plus proches
  - mettre à jour la matrice des distances entre les groupes
- **Jusqu'à (la formation d'un seul groupe)**

FIN.

Dans ces algorithme, on a besoin d'estimer la distance entre deux groupes (ensembles) de données ; en plus de la distance entre les données

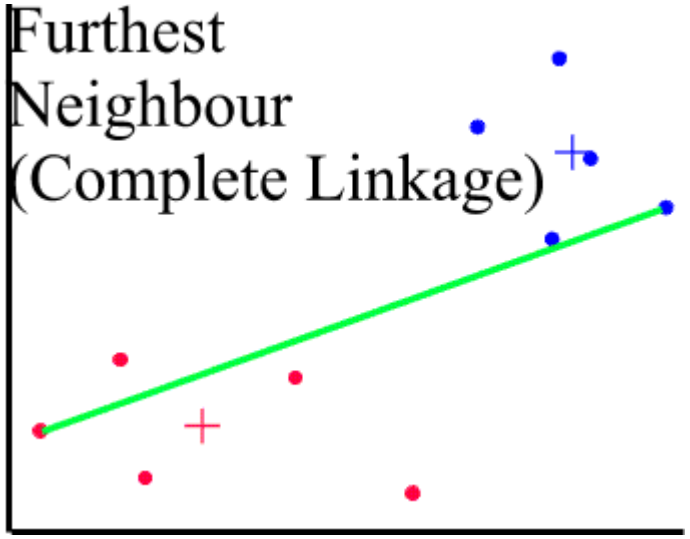
Distance (1)

- On rappelle que la distance entre les données est calculée en utilisant une fonction de distance dont la nature et la formulation dépend des données
- **single-linkage : MIN**



Distance (2)

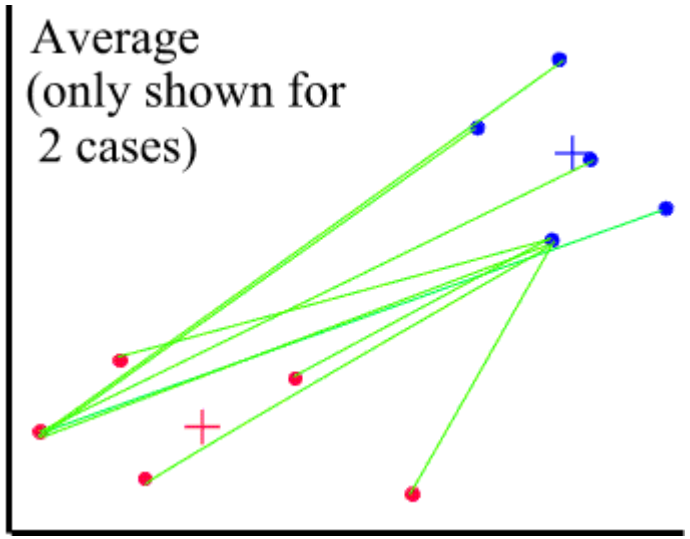
- **complete-linkage : MAX**



dist(g1,g2) = points les plus éloignés

Distance (3)

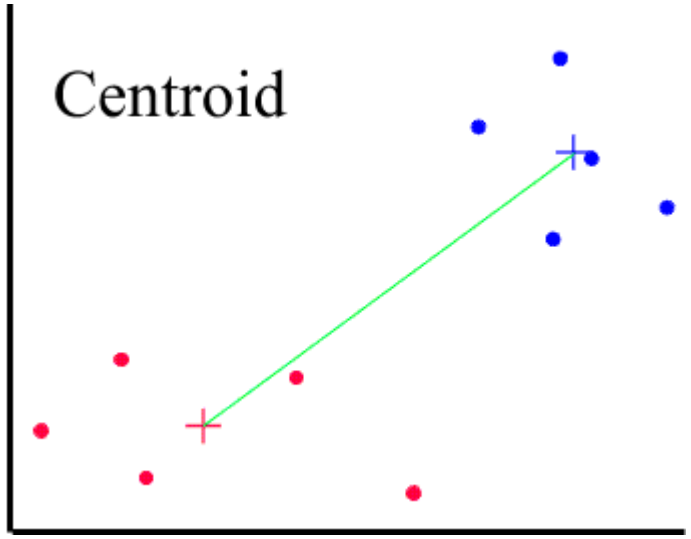
- **average-linkage : MOYENNE**



dist(g1,g2) = moyenne des distances

Distance (4)

- **centroid-linkage : centre des classes**
  - définir le centroide de chaque groupe (moyenne des données qu'elle contient)
  - prendre la distance entre les centroides

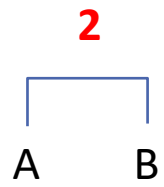


UPGMA (1)

- UPGMA : Unweighted pair group method with arithmetic mean
- Utilise la moyenne (average-linkage)
  - *dist(A,B),C = (distAC + distBC) / 2 = 4*
  - *dist(A,B),D = (distAD + distBD) / 2 = 6*
  - *dist(A,B),E = (distAE + distBE) / 2 = 6*
  - *dist(A,B),F = (distAF + distBF) / 2 = 8*

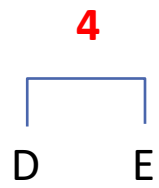
étape 1

	A	B	C	D	E	F
A	0					
B	2	0				
C	4	4	0			
D	6	6	6	0		
E	6	6	6	4	0	
F	8	8	8	8	8	0



étape 2

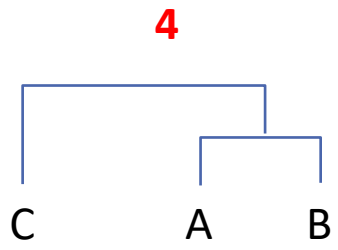
	AB	C	D	E	F
AB	0				
C	4	0			
D	6	6	0		
E	6	6	4	0	
F	8	8	8	8	0



UPGMA (2)

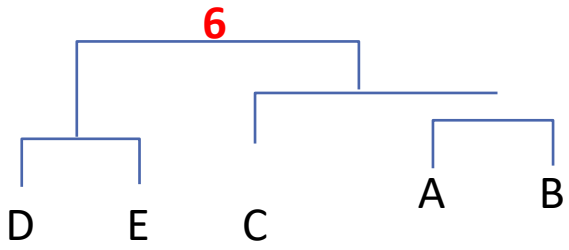
étape 3

	AB	C	DE	F
AB	0			
C	4	0		
DE	6	6	0	
F	8	8	8	0



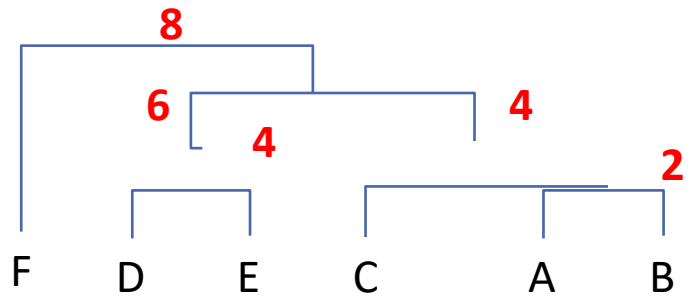
étape 4

	ABC	DE	F
ABC	0		
DE	6	0	
F	8	8	0



étape 5

	ABCDE	F
ABCDE	0	
F	8	0



Single-Linkage (Min)

Agrégation selon le lien minimum

	a	b	c	d	e
a	0	23	35	43	50
b	23	0	21	32	45
c	35	21	0	11	25
d	43	32	11	0	17
e	50	45	25	17	0

Tableau des dissimilarités

$G_1 = \{c, d\} \Rightarrow$

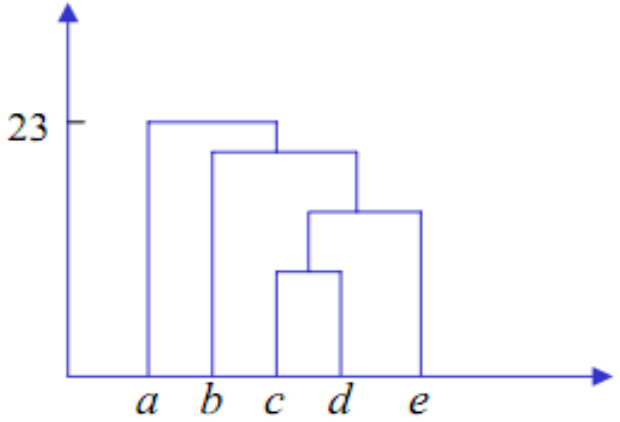
	a	b	e	G_1
a	0	23	50	35
b	23	0	45	21
e	50	45	0	17
G_1	35	21	17	0

$G_2 = \{e, G_1\} \Rightarrow$

	a	b	G_2
a	0	23	35
b	23	0	21
G_2	35	21	0

$G_3 = \{b, G_2\} \Rightarrow$

	a	G_3
a	0	23
G_3	23	0



Analyse de complexité

- La matrice des distances occupe un espace de $O(N^2)$
- La complexité temporelle est de $O(N^3)$
  - on a besoin de mettre à jour une matrice de $O(N^2)$ pendant N itérations
  - pour certains algorithmes, la complexité temporelle peut-être de $O(N^2 \log(N))$

Méthodes par Partitionnement

Concepts de base (1)

- Essayer de diviser *directement* l'ensemble des données en des groupes *homogènes*
- **K-Moyenne** est le premier algorithme qui a été proposé dans les méthodes par partitionnement
- Plusieurs autres algorithmes ont été dérivés
  - FCM, PCM, P3M, ...
- **Une difficulté intrinsèque: étant l'ensemble *D,* quel est *le nombre de groupes existants* dans D?**

Concepts de base (2)

(a) Original points.

(b) Two clusters.

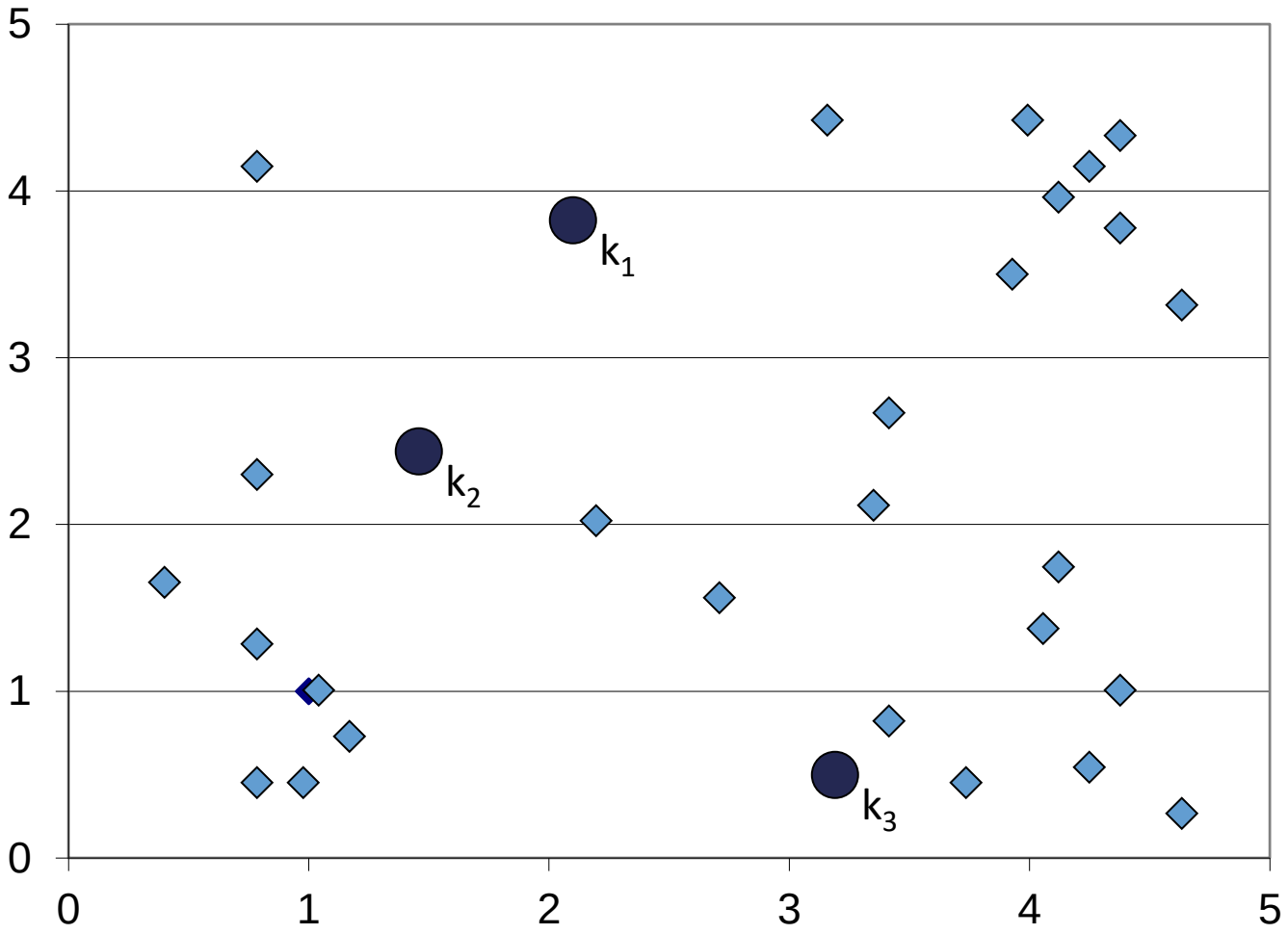
(c) Four clusters.

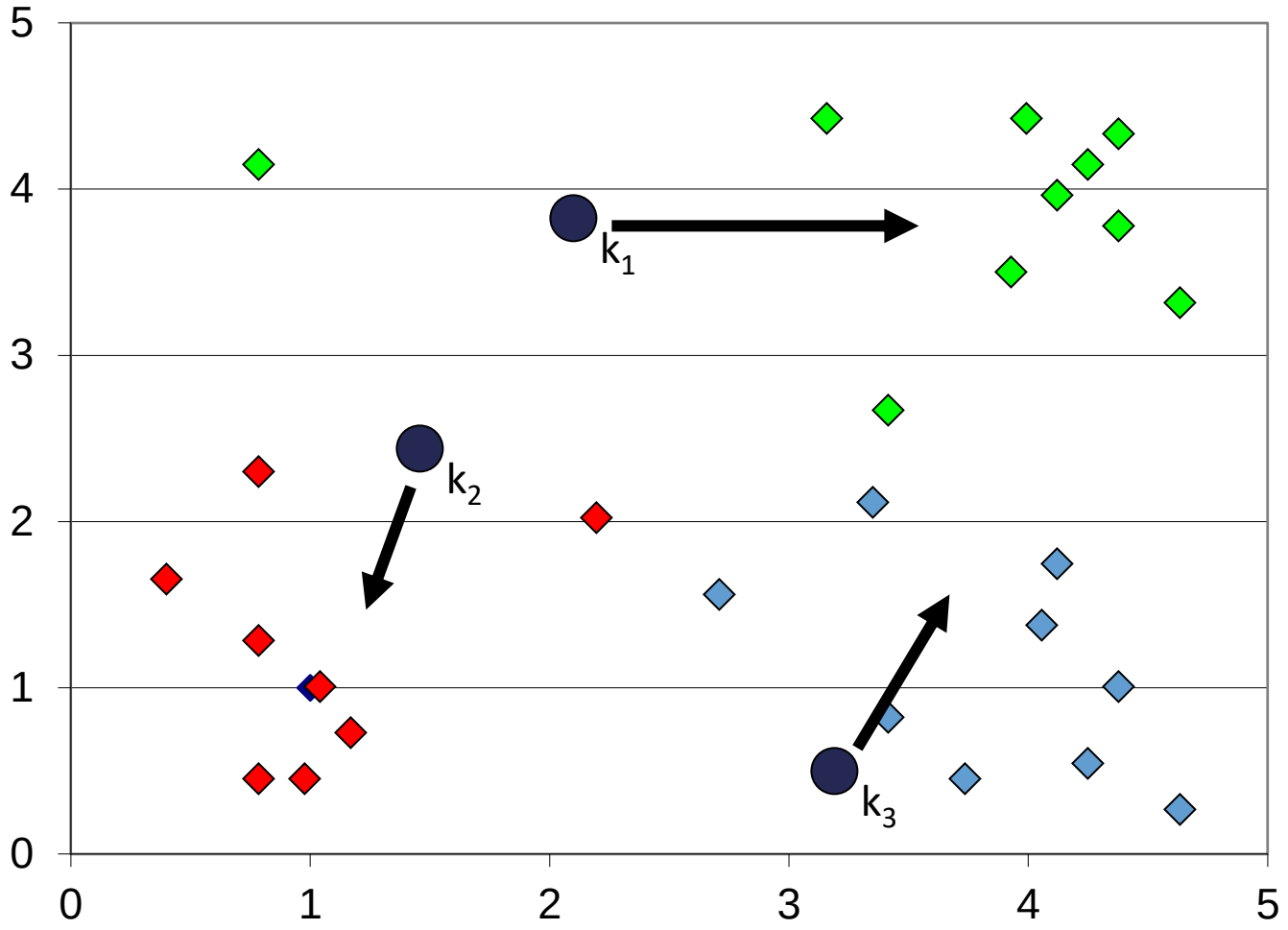
(d) Six clusters.

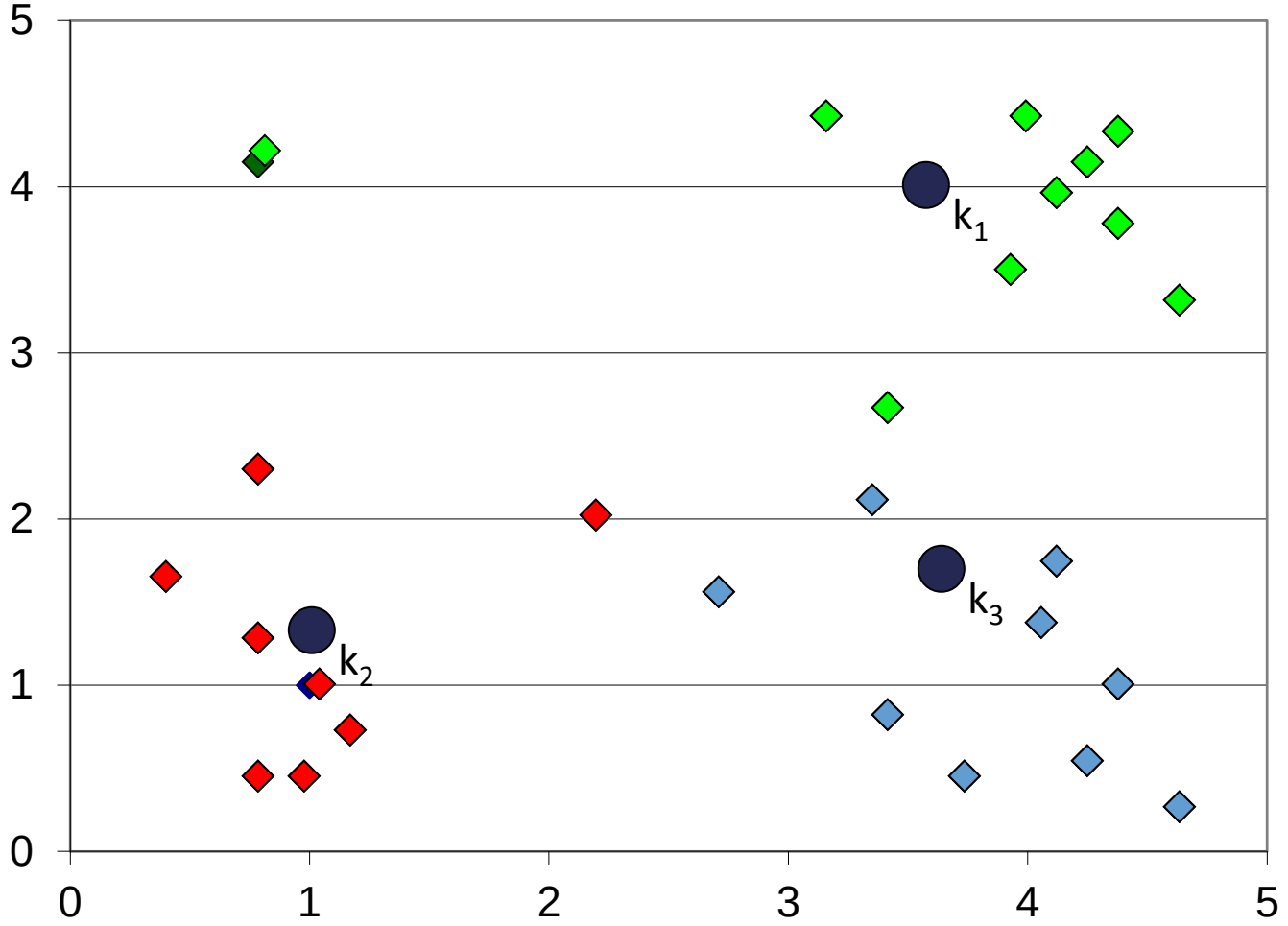
Concepts de base (3)

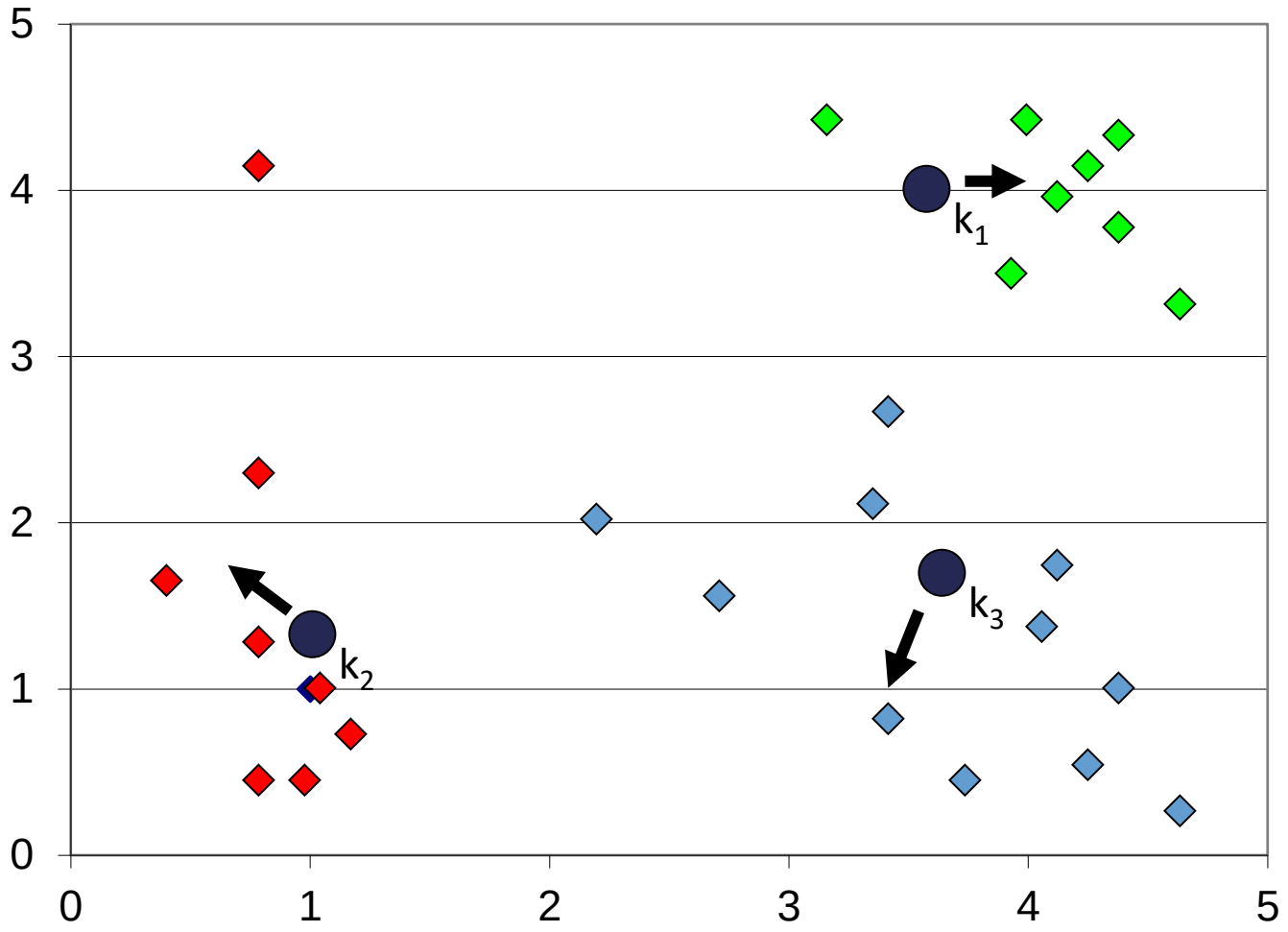
- La plupart des approches supposent un nombre prédéfini de classes
  - L'algorithme accepte comme paramètre le nombre de groupes à générer
- On peut utiliser un critère qui permet d'estimer le nombre optimal de groupe:
  - entropie, gain informationnel, ...
  - classer pour plusieurs valeurs du nombre de classes et choisir le partitionnement optimal selon le critère choisi
  - très coûteuse sur le plan temps d'exécution

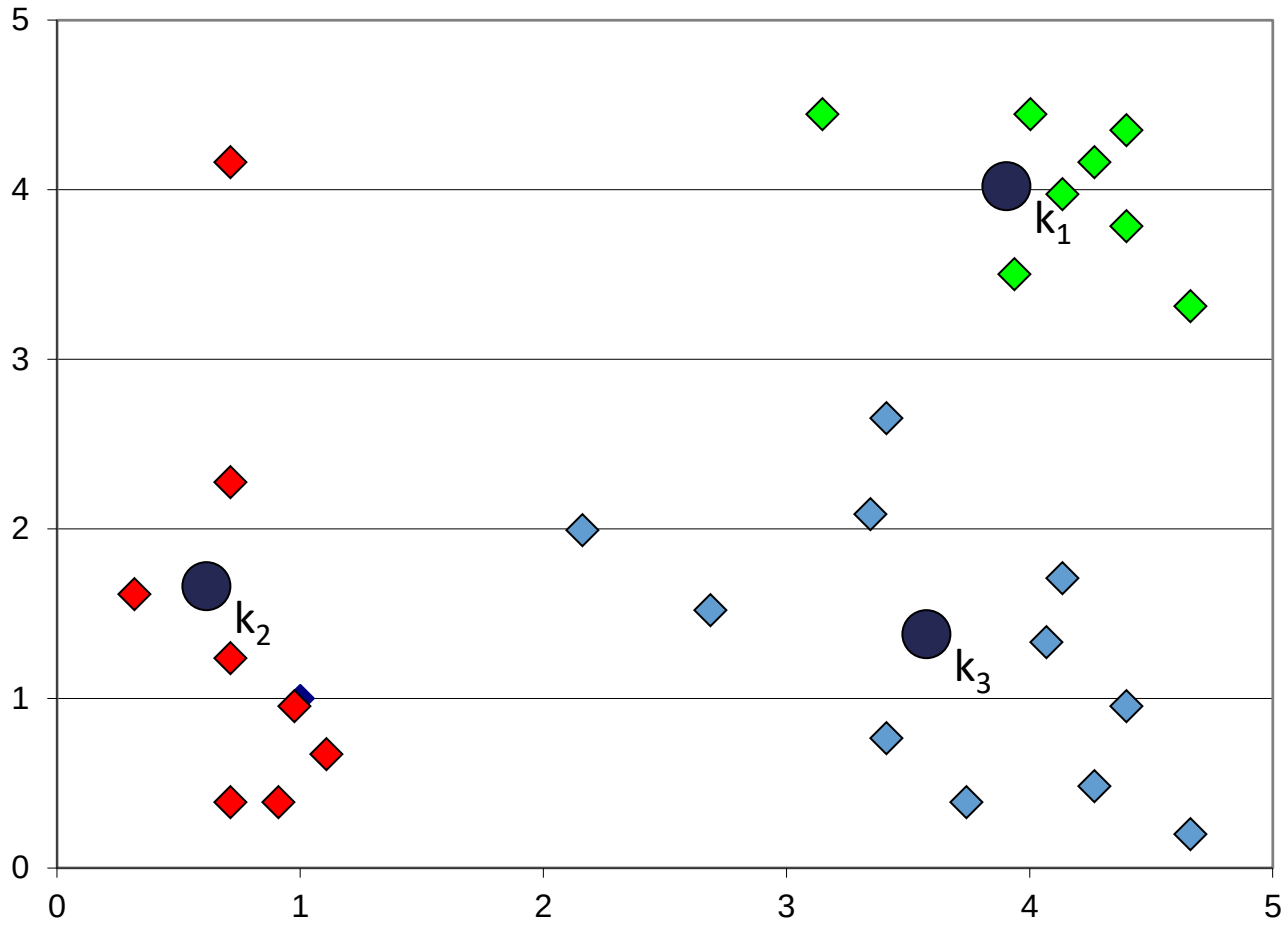
K-Moyenne


5
4
3
2
1
0
0
1
2
3
4
5
k1
k2
k3




5
4
3
2
1
0
0
1
2
3
4
5
k1
k2
k3


5
4
3
2
1
0
0
1
2
3
4
5
k1
k2
k3


5
4
3
2
1
0
0
1
2
3
4
5
k1
k2
k3

Source: K-Means Clustering in action

Algorithme (1)

Algorithm: *k*-means. The *k*-means algorithm for partitioning, where each cluster's center is represented by the mean value of the objects in the cluster.

Input:

- *k*: the number of clusters,
- *D*: a data set containing *n* objects.

Output: A set of *k* clusters.

Method:

(1) arbitrarily choose *k* objects from *D* as the initial cluster centers;
(2) **repeat**
(3) (re)assign each object to the cluster to which the object is the most similar, based on the mean value of the objects in the cluster;
(4) update the cluster means, that is, calculate the mean value of the objects for each cluster;
(5) **until** no change;

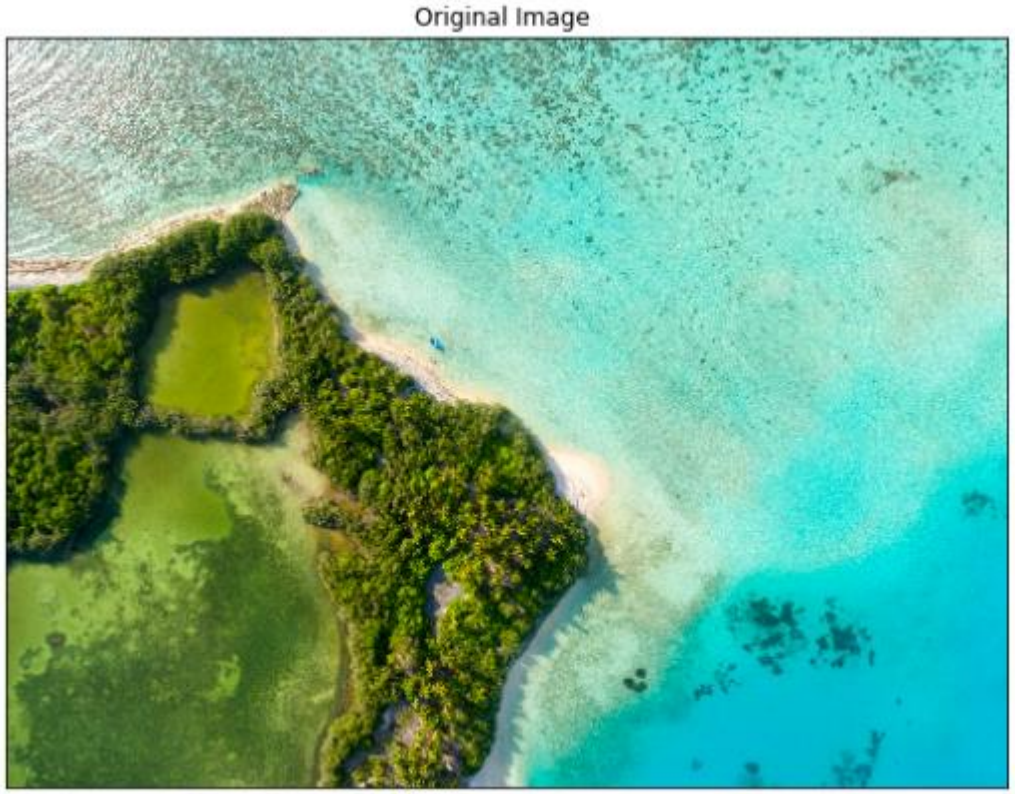
Algorithme (2)

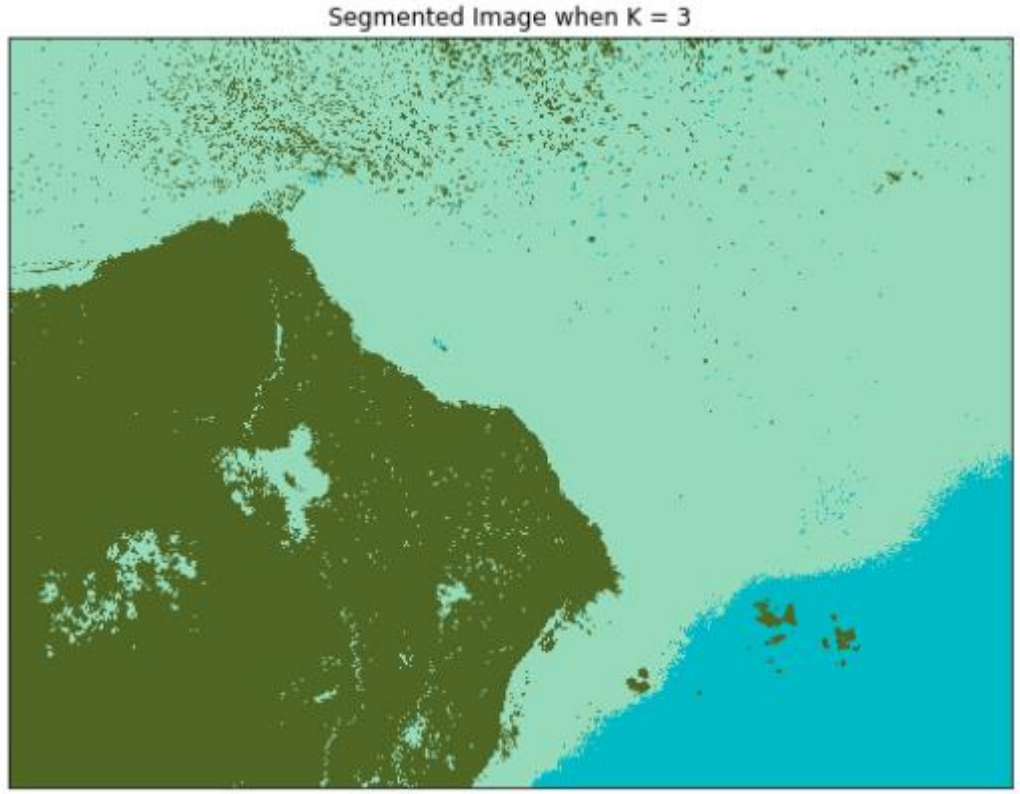
- L'algorithme K-moyenne est simple et efficace pour classer de grands volumes de données
- Il a une complexité temporelle de ***O(nkt)***
  - n : le nombre de données, k: nombre de classe, t: nombre de cycles d'exécution
  - généralement: k <<n, t << n
- Plusieurs critères d'arrêt sont utilisés en pratique et modélisent le même concept: « stabilité des classes »
  - **Critère 1:** les centres de classes deviennent stables d'une itération *T* à la suivante

$$\sum_{j=1}^{k} \left\| m_j(T+1) - m_j(T) \right\| \Big/ k < \varepsilon$$

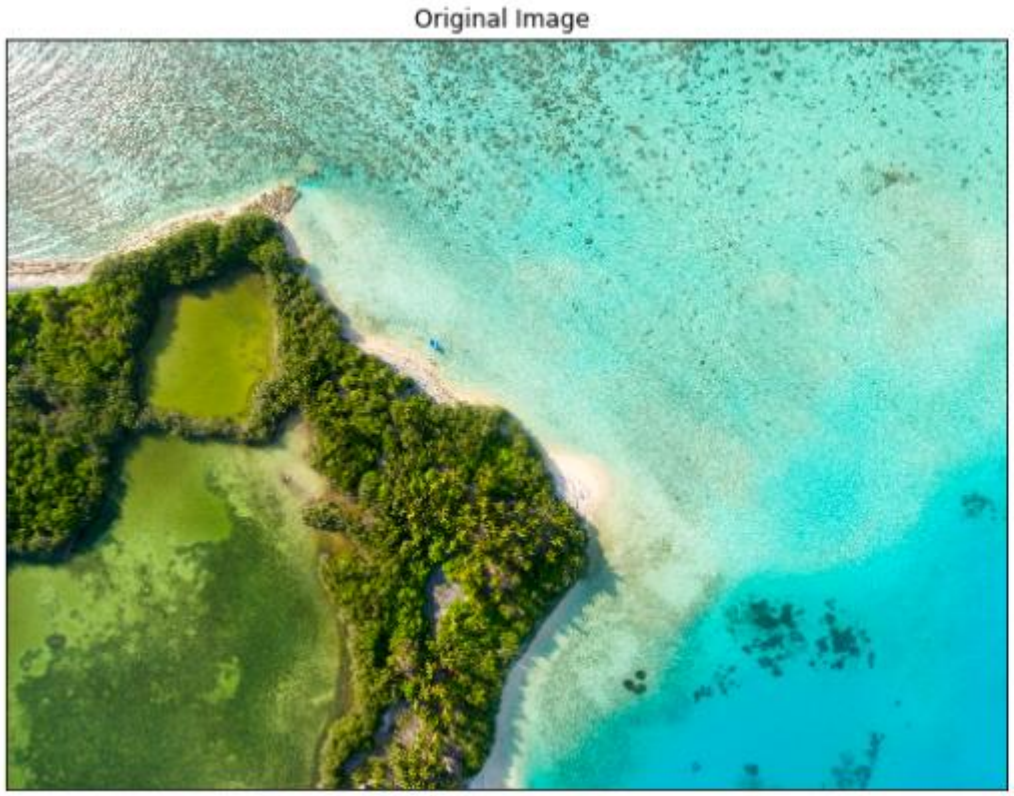
  - **Critère 2 :** le contenu des classes devient stable

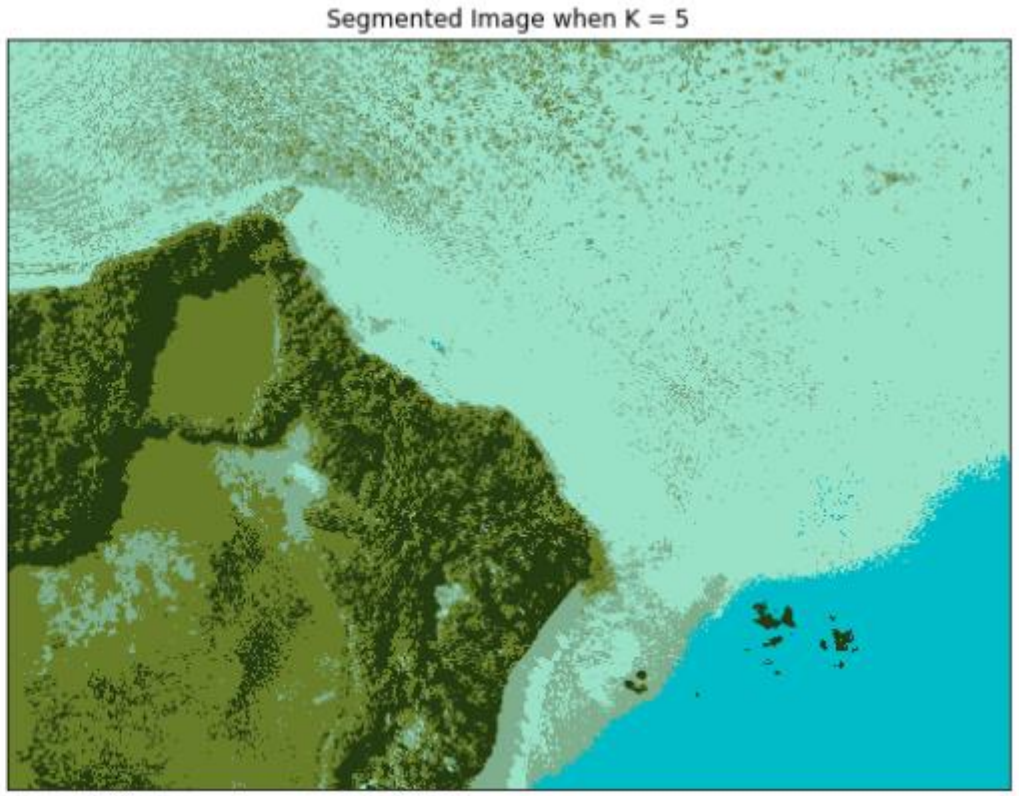
Valeur de K (1)?



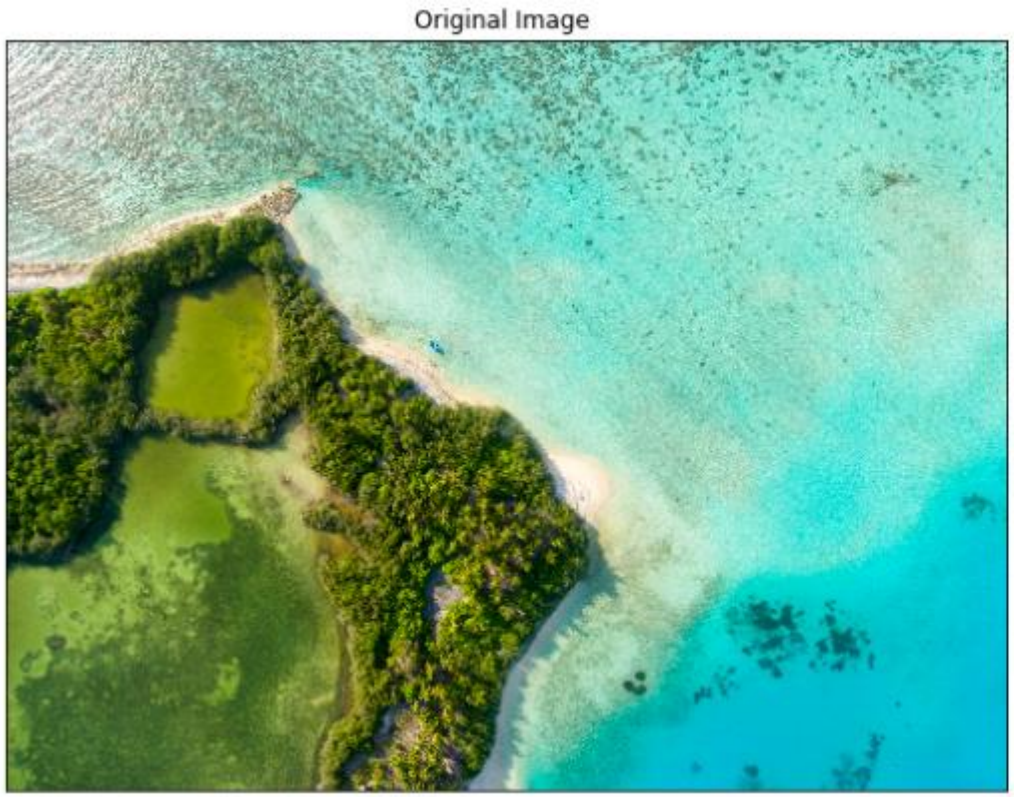


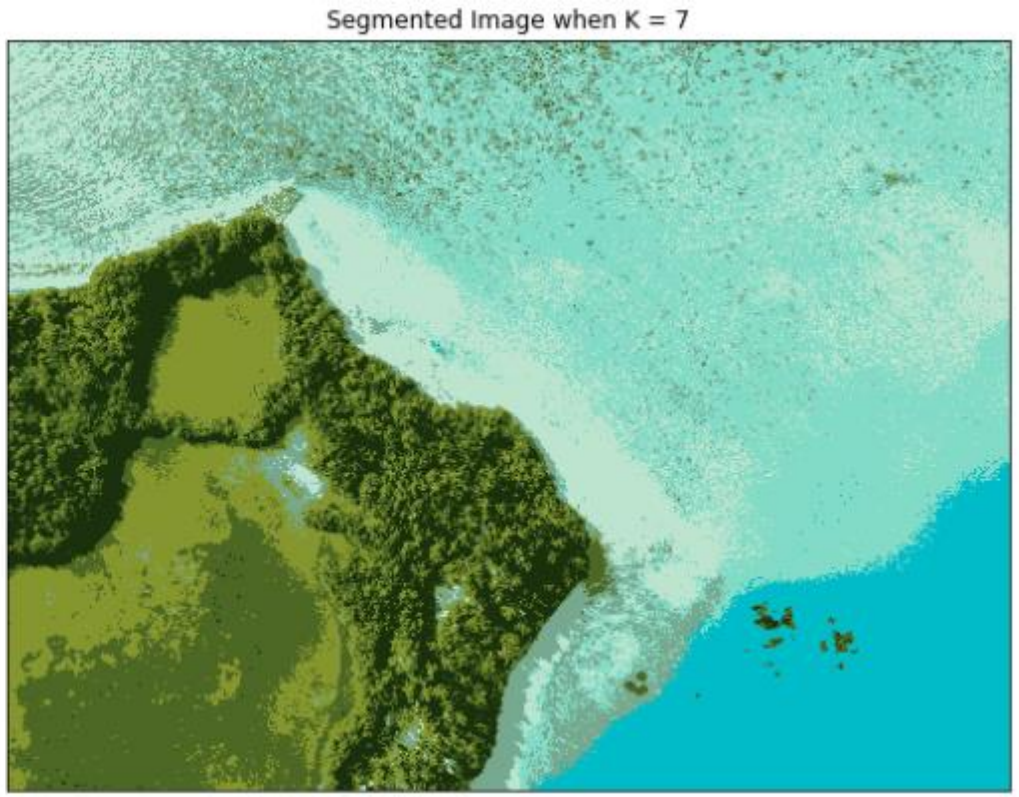
Valeur de K (2)?





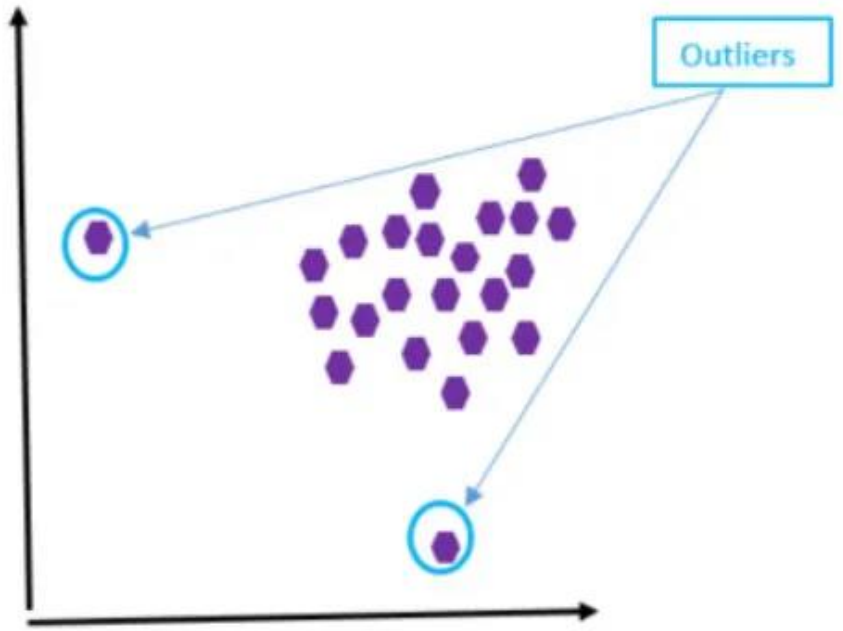
Valeur de K (3)?





Sensibilité au bruit ?

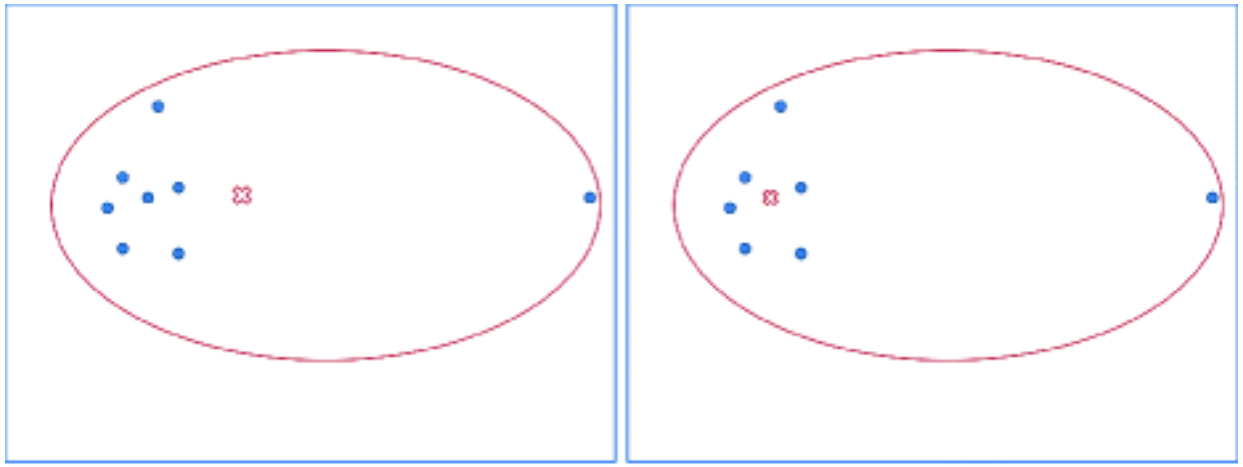
- Il est très sensible au **bruit (*outliers*)**
  - un bruit est un signal qui vient de s'ajouter aux données d'origine
  - Puisque toutes les points (données et bruit) participent également à la formation des classes



K-Medoides

Principe de Base

- Comment peut-on modifier le *K-Means* pour qu'ils deviennent moins sensible au bruit?
- Au lieu de prendre le **centroïde** (valeur moyenne) comme un point de référence dans une classe, **on peut choisir une donnée pour représenter la classe**
- Ce représentant de classe est appelé le **médoïde**



(a) Mean (b) Medoid

Un **médoïde** est le la **« donnée la plus centrale dans le cluster »**

K-Medoides

Algorithm: *k*-medoids. PAM, a *k*-medoids algorithm for partitioning based on medoid or central objects.

Input:

- *k*: the number of clusters,
- *D*: a data set containing *n* objects.

Output: A set of *k* clusters.

Method:

(1) arbitrarily choose *k* objects in *D* as the initial representative objects or seeds;
(2) **repeat**
(3) assign each remaining object to the cluster with the nearest representative object;
(4) randomly select a nonrepresentative object, o_{random};
(5) compute the total cost, *S*, of swapping representative object, o_j, with o_{random};
(6) **if** $S < 0$ **then** swap o_j with o_{random} to form the new set of *k* representative objects;
(7) **until** no change;

Total Cost :

$$S = \sum_{i=1}^{k} \sum_{p \in C_i} dist((p, o_i)$$

C_i : *un cluster*

o_i : *le médoide de* C_i

Exemple (1)

- Cet exemple se trouve sur le lien suivant : https://en.wikipedia.org/wiki/K-medoids

X_1	2	6
X_2	3	4
X_3	3	8
X_4	4	7
X_5	6	2
X_6	6	4
X_7	7	3
X_8	7	4
X_9	8	5
X_{10}	7	6

- On désire classer les données suivantes en **deux classes**
- Deux médoïdes: **m1**=x2=(3,4)} et **m2**=x8=(7,4)} sont choisis d'une manière arbitraire.
- On utilise la **distance de Manhattan**

$$X = (x_1, x_2, \dots, x_n); Y = (y_1, y_2, \dots, y_n), alors$$

$$Manhatten(X, Y) = \sum_{i=1}^{n} |x_i - y_i|$$

Exemple (2)

Données		m1={3,4}	m2=(7,4)
1	(2, 6)	**3**	7
2	**(3, 4)**	**0**	4
3	(3, 8)	**4**	8
4	(4, 7)	**4**	6
5	(6, 2)	5	**3**
6	(6, 4)	3	**1**
7	(7, 3)	5	**1**
8	**(7, 4)**	4	**0**
9	(8, 5)	6	**2**
10	(7, 6)	6	**2**
Cost		11	9

Cluster1 = {(2,6); (3,4); (3,8); (4,7)}

Cluster2 = {(7,4);(6,2);(6,4);(7,3);(8,5);(7,6)}

Total Cost S = (3 + 0 +4 + 4) + (3 +1 +1 +0 +2 +2)=11+9=20

Etape 2: choisissons un point non-médoide

Supposons d=(7,3)

Exemple (3)

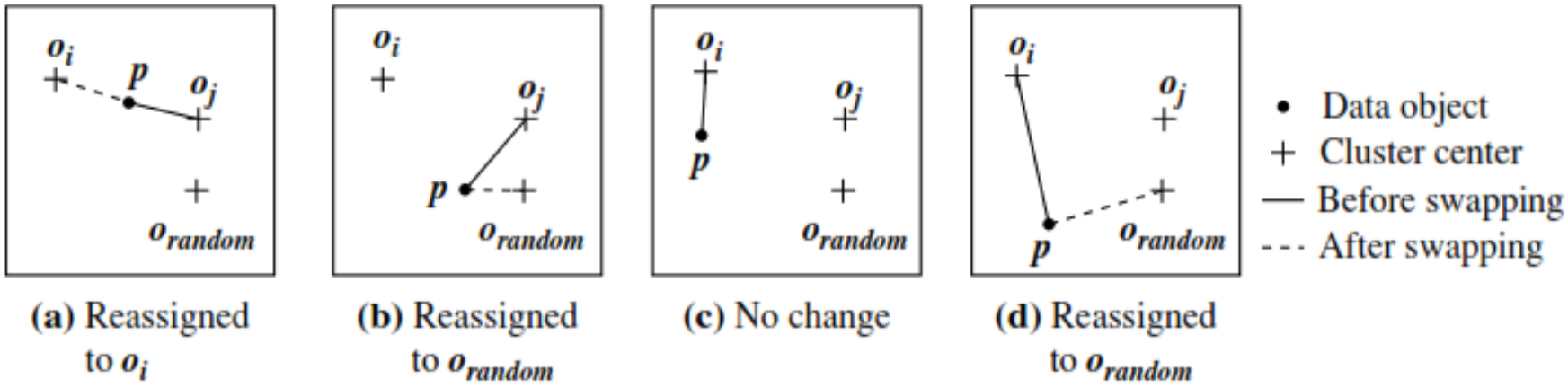
i	m1		Data objects (X_i)		Cost (distance)
1	3	4	2	6	**3**
3	3	4	3	8	**4**
4	3	4	4	7	**4**
5	3	4	6	2	5
6	3	4	6	4	3
8	3	4	7	4	4
9	3	4	8	5	6
10	3	4	7	6	6

i	d		Data objects (X_i)		Cost (distance)
1	7	3	2	6	8
3	7	3	3	8	9
4	7	3	4	7	7
5	7	3	6	2	**2**
6	7	3	6	4	**2**
8	7	3	7	4	**1**
9	7	3	8	5	**3**
10	7	3	7	6	**3**

- Total Cost = (3+4+4) + (2+2+1+3+3)
- =11 + 11 = 22
- Coût d'échange = 22 -20 >0
- Donc, **on ne doit pas effectuer cet échange et essayer un autre**

Propriétés

- La complexité temporelle pour le K-Médoids est :
- A chaque itération on : $O(k(n-k)^2) \approx O(n^2)$ puisque en général k est négligeable devant n
- Ce qui donne une complexité globale pour t itérations: $O(tn^2)$
- Dans le cas ou un remplacement est effectué, on doit réaffecter certaines données à d'autres clusters, c'est-à-dire que les données peuvent bouger d'un cluster vers un autre.



(a) Reassigned to o_i **(b)** Reassigned to o_{random} **(c)** No change **(d)** Reassigned to o_{random}