Apprentissage non supervisé

Clustering

Mme Hamdad Leila

ESI, LCSI.

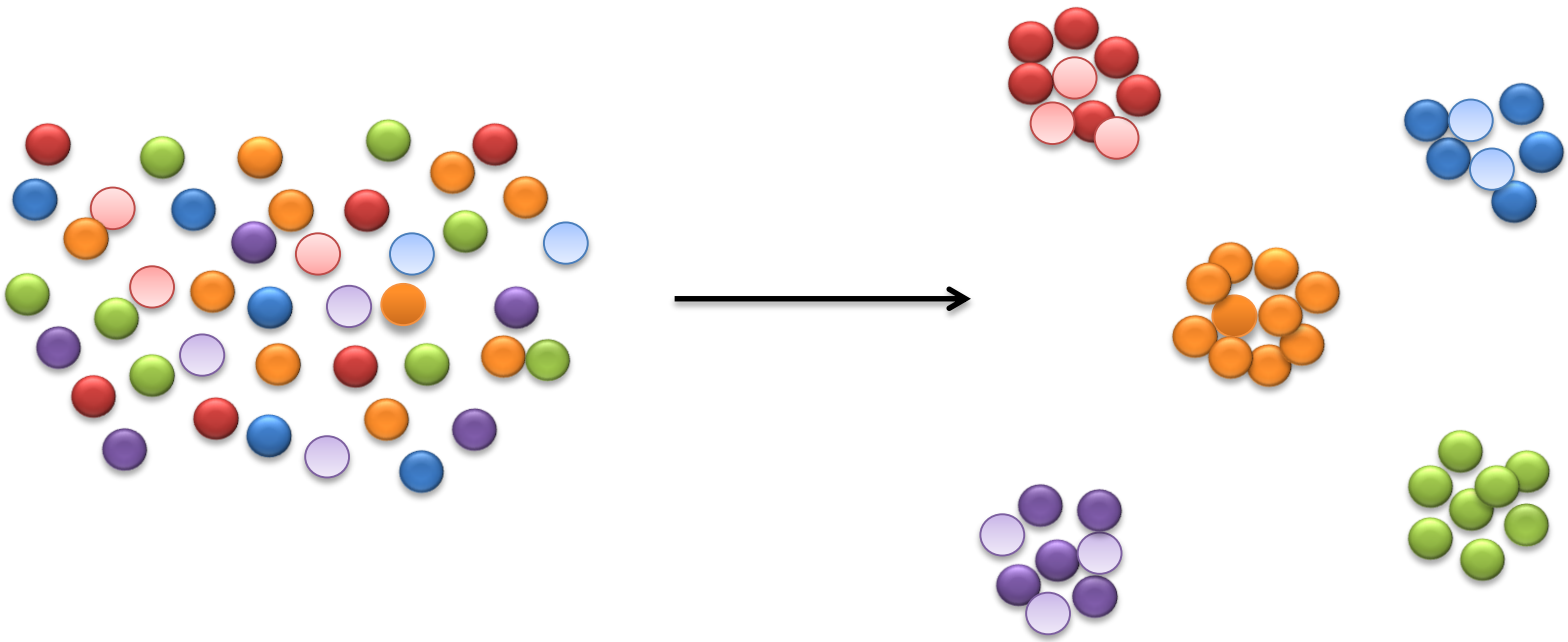
Plan

- Introduction
- Kmeans
- Métriques pour détecter le nombre de classes
- Classification basée sur la densité: DBSCAN
  - Principe de fonctionnement
  - Caractéristiques
- Classification par grille : SOM
  - Principe de fonctionnement
  - Caractéristiques
- Conclusion

Introduction

- **But** :Regrouper un ensemble de données en différents groupes homogènes.
- Les objets d’un même groupe: les plus similaires possibles
- Les objets de groupes différents: les plus dissimilaires possibles.

Classification Non Supervisée



1. **Les données du même groupe doivent être les plus semblables possible.**
2. **Les données de groupes différents doivent être les plus différentes possible.**
3. **Le nombre de groupes et leur signification ne sont pas connus à l'avance.**

Domaines d'application

- Marketing :

- Partitionner une clientèle en segments dotés chacun d'une offre et d'une communication spécifique.
- Répartir l'ensemble des magasins d'une enseigne selon le type de clientèle, rayon (selon type d'article) ou selon la taille du magasin.

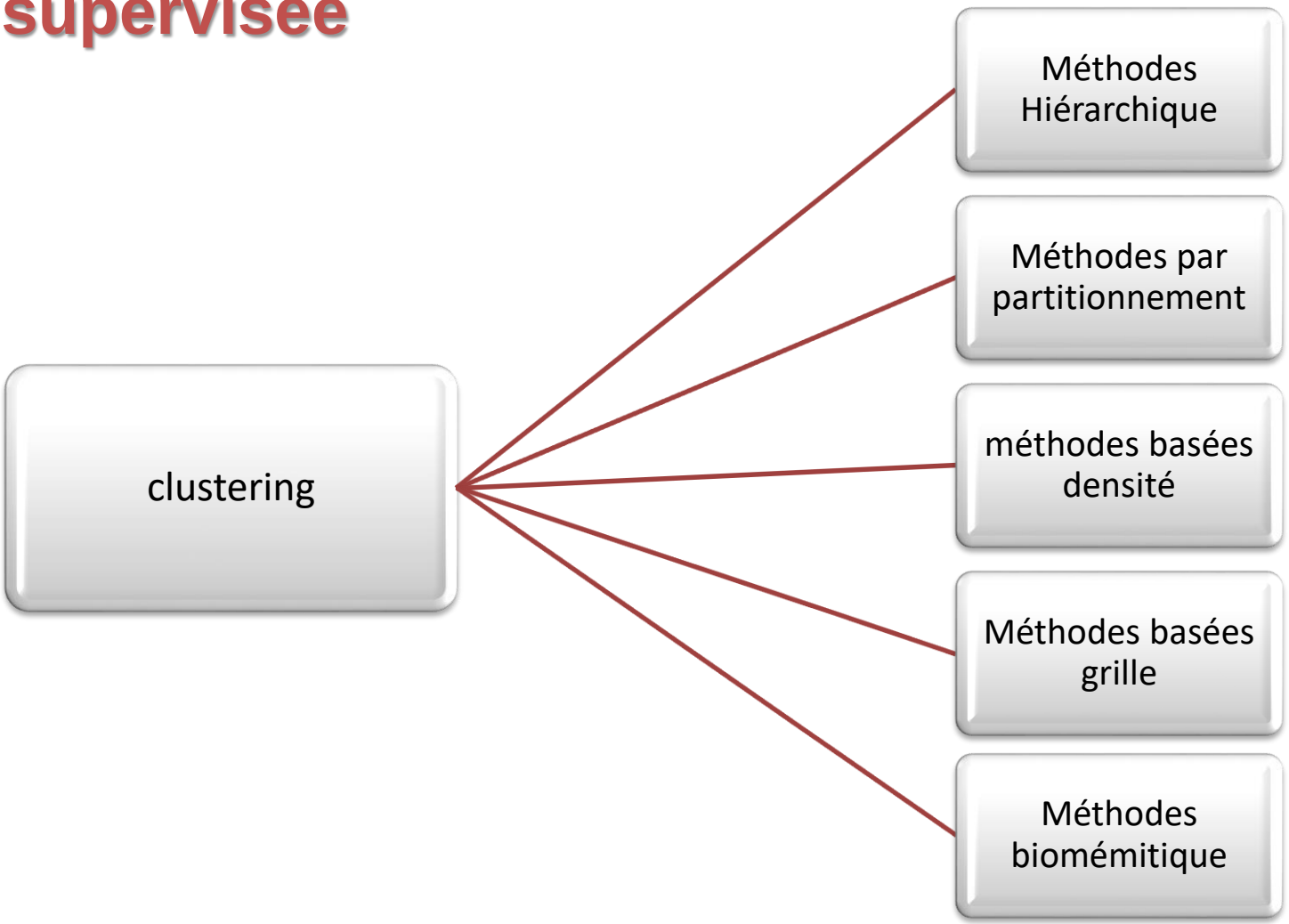
- Médical :

  Groupes de patients susceptibles d'être soumis à des protocoles thérapeutiques déterminés. Un groupe contient tous les patients réagissant identiquement.

- Sociologie : Découper la population en groupes homogènes du point de vue socio-démographique, style de vie, opinions ou attentes.

Classification Non Supervisée

- **Taxonomie de la classification non supervisée**



K-means

- Soit E l'ensemble à partitionner.
- Début, on choisit k points de E aléatoirement appelés centroides.

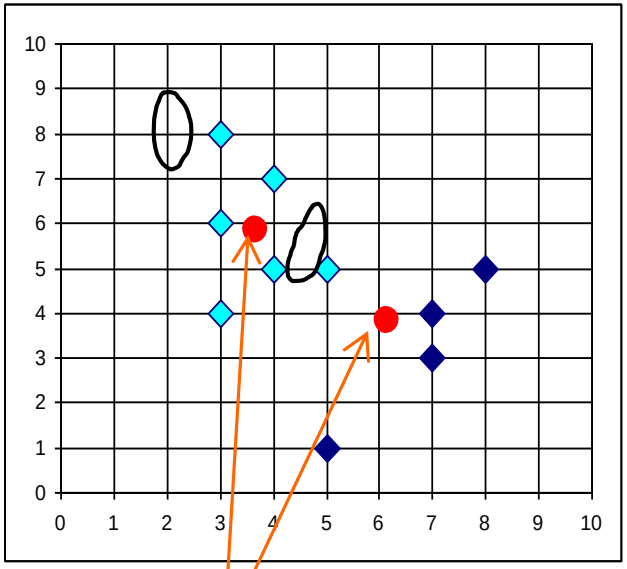
Entrée : un ensemble d'objets E, nombre de classes K.

- 1. Choisir K individus au hasard dans E(comme centres des classe initiaux).
- 2. Affecter chaque individu au centre le plus proche.
- 3. Recalculer le centre de chacune de ces classes.
- 4. Répéter l'étape (2) et (3) jusqu'à stabilité des centres.
- 5. Editer la partition obtenue.
- Sortie : Partition de E en K classes.

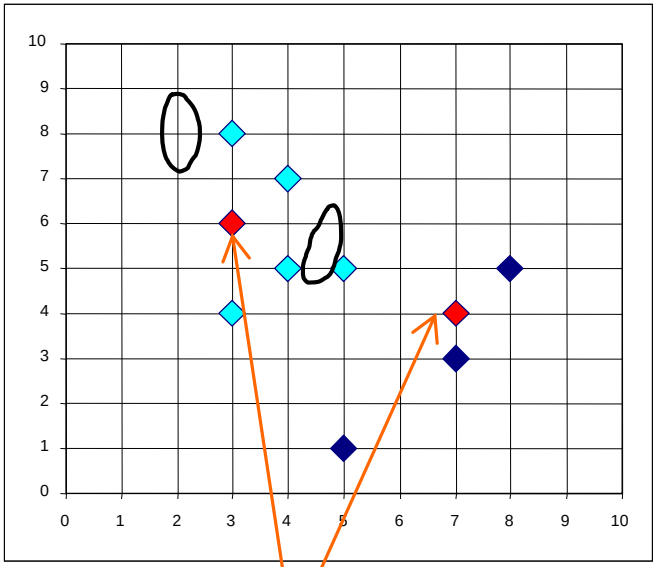
- Cette méthode est facile à mettre en œuvre.
- Elle converge très vite.
- Elle est sensible aux données aberrantes.
- Elle a tendance à donner des classes de formes sphériques.
- L'inconvénient majeur de la méthode est le choix aléatoire des noyaux à l'initialisation de l'algorithme. Car deux choix différents des représentants initiaux, mènent à des partitions différentes.

K-medoids

- Trouver k objets représentatives, appelés k-medoids
  - PAM (Partitioning Around Medoids)
  - CLARA (Clustering Large Applications)
  - CLARANS (Randomized sampling)



K-means



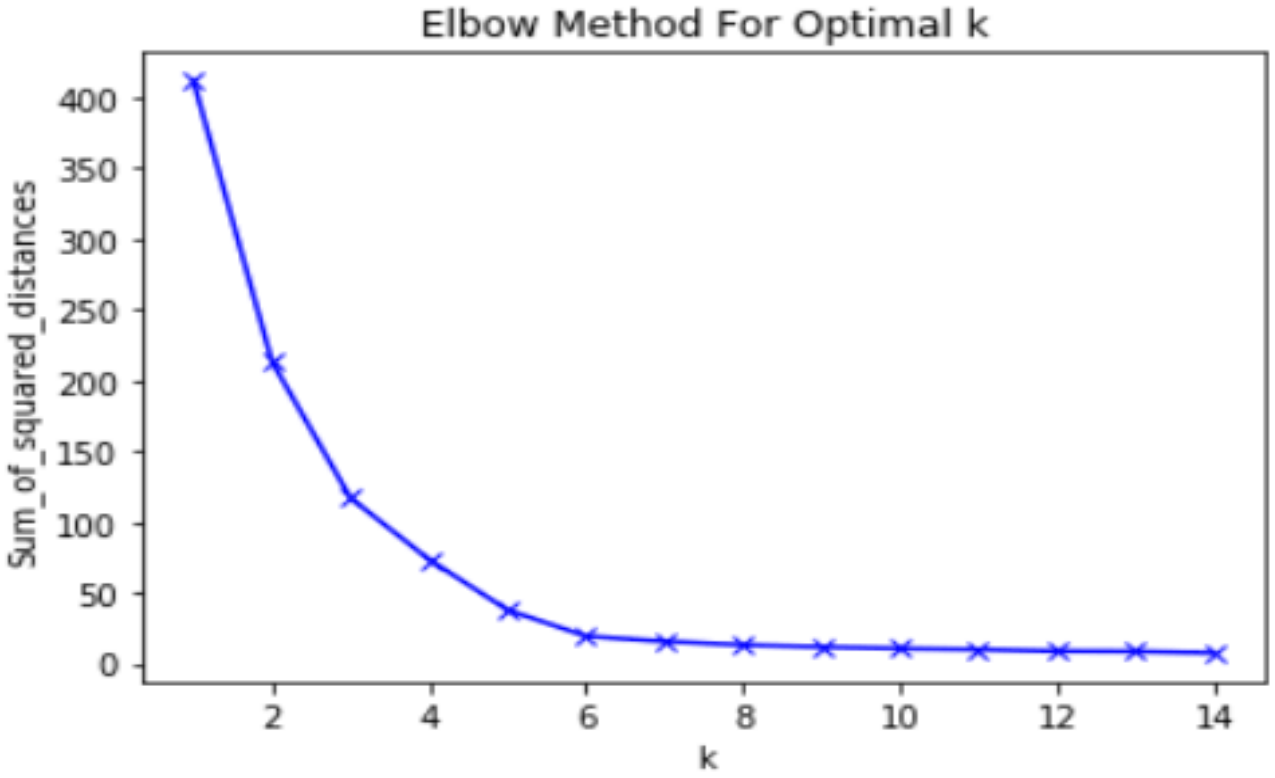
k-medoids

Métriques pour déterminer le nombre de classes

Méthode Elbow

Méthode du coude.

- Exécuter un clustering k-means sur l'ensemble de données pour des valeurs de k variant par exemple de 1 à 10
- Pour chaque valeur de k, calculer la somme des erreurs au carré (SSE) ou l’inertie intra classes
- Tracer la courbe des SSE en fonction de K
- Si le graphe ressemble à un bras, alors le «coude» est la meilleure valeur de k.


Elbow Method For Optimal k
Sum_of_squared_distances
400
350
300
250
200
150
100
50
0
2
4
6
8
10
12
14
k

Méthode Silhouette

- On dit que c'est la meilleure mesure pour décider du nombre de classes à définir à partir des données.
- Pour chaque objet i appartenant au cluster Ck, calculer a(i):

$$a(i) = \frac{1}{|C_k| - 1} \sum_{i \neq j} d(i,j)$$

- la distance moyenne entre i et tous les autres points de données du même cluster.

$$b(i) = \min_{l \neq k} \frac{1}{|C_l|} \sum_{j \in C_l} d(i,j)$$

- b(i) est la plus petite distance moyenne de i à tous les points de tout autre cluster auquel il n'appartient pas. Le cluster avec la plus petite dissimilarité moyenne est dit "cluster voisin" de i.

$$s(i) = \frac{a(i) - b(i)1}{\max(a(i), b(i))} \; si \; |C_k| > 1$$

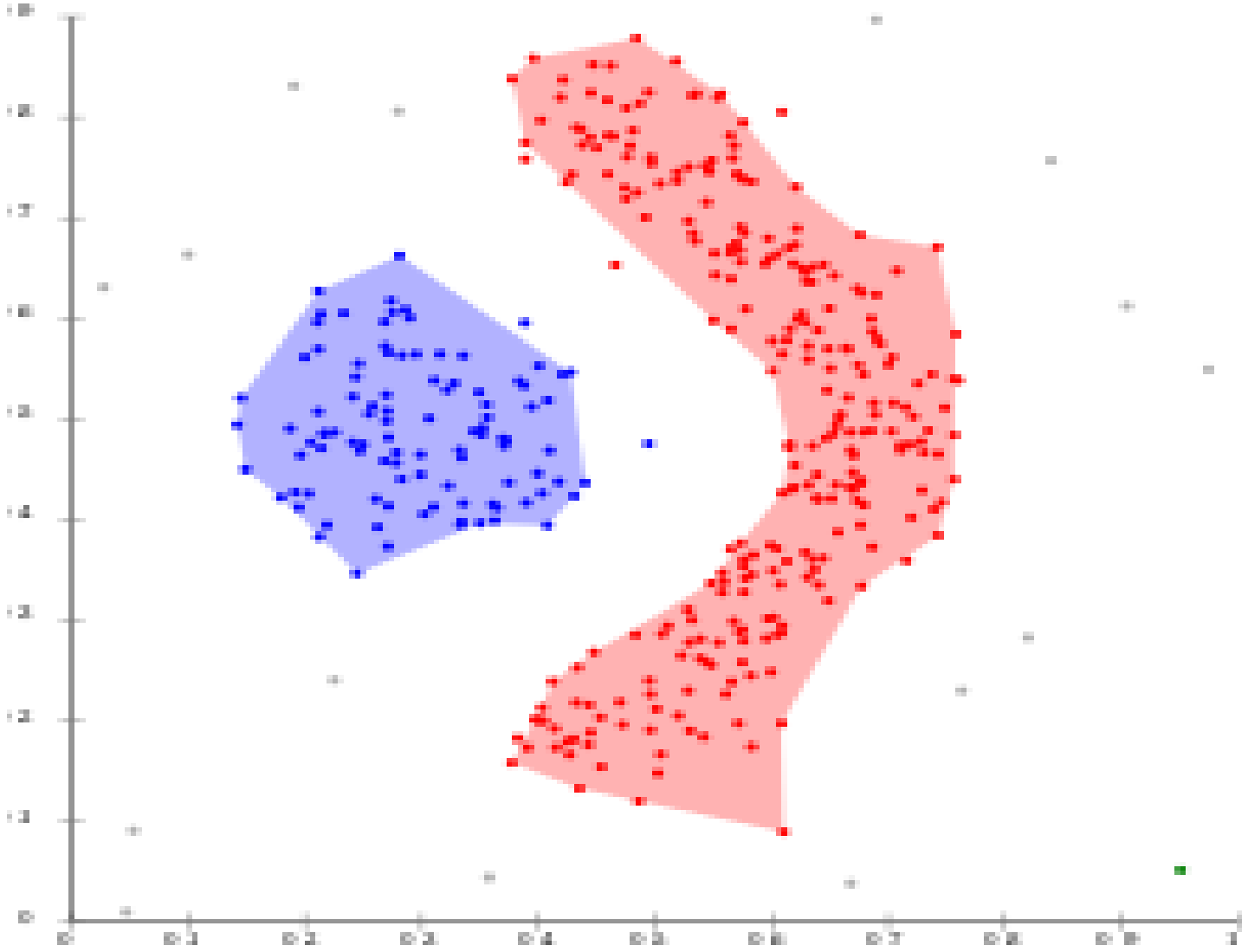
$$s(i) = 0 \; si \; |C_k| = 1$$

- Le coefficient varie entre -1 et 1.
- Une valeur proche de 1 implique que l'objet est bien classifié,
- Si s(i) est plutôt négative, ceci indique que l'objet n'est pas dans le bon cluster.
- Un s (i) proche de zéro signifie que la donnée est à la frontière de deux clusters naturels.

Avantages et inconvénients

- Cette méthode est meilleure car elle donne le nombre optimal de clusters.
- Mais cette métrique est coûteuse en calcul car le coefficient est calculé pour chaque instance.
- Le choix de la métrique optimale pour le nombre de clusters doit être pris en fonction des besoins.

DBSCAN – Density-Based Spatial Clustering of Applications with Noise



Composantes de DBSCAN

- Trois types de points:
  - Points denses
  - Points frontières
  - Points aberrants
- Deux paramètres principaux:
  - Epsilon: Rayon du voisinage
  - Minpoints: Nombre minimal de points dans ce voisinage.

Un point est dense s'il a un nombre de points supérieur à Minpoints dans son voisinage

Eléments de DBSCAN

DBSCAN: algorithme basé sur la densité

- Densité = Nombre de points à l'intérieur d'un rayon donné (Eps)
- Un point est un noyau s'il a plus qu'un nombre donné de points (MinPts) à l'intérieur d'un rayon donné (Eps)

Ces points sont à l'intérieur d'un même segment

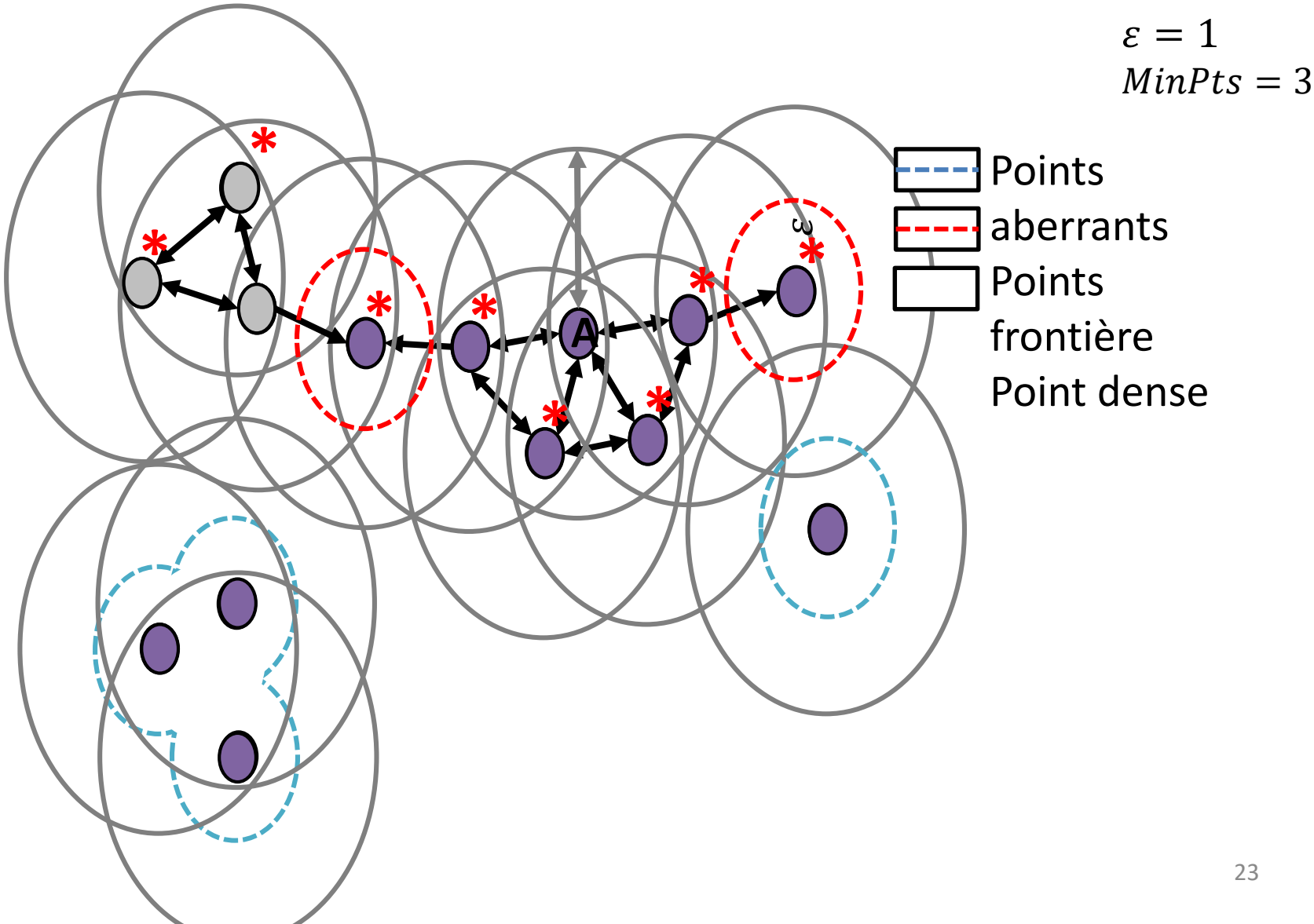
- Un point frontière est un point non noyau qui a un nombre de points inférieur à MinPts à l'intérieur de son Eps, mais qui se trouve à moins d'un Eps d'au moins un noyau
- Un point bruité est un point qui n'est ni un noyau, ni un point frontière

DBSCAN : Principe de fonctionnement

- Regroupe les objets dans le **même cluster**, s'ils sont **connectés** l'un à l'autre par une population de point dense.
- Si A est un **point dense** alors ses **voisins** sont dans le **même cluster**.
- **Idée de base:**
  - Démarrer d'un **point dense A**.
  - Tout les points joignable par A ou d'autres points dense connectés à A font partie du même cluster.
  - Tout les points non joignable par des points denses sont des bruits.
  - *On répète l'opération jusqu'à ce que tout les points soit explorés*

Source: [6]

DBSCAN : Principe de fonctionnement



DBSCAN : Caractéristiques

Avantages

- Détecte les points aberrants
- Le nombre de classes n'est pas spécifié
- Peut découvrir des cluster de forme arbitraire.
- Les deux paramètres peuvent être spécifiés par un expert du domaine.
- Peut trouver des clusters non linéairement séparables
- Est déterministe

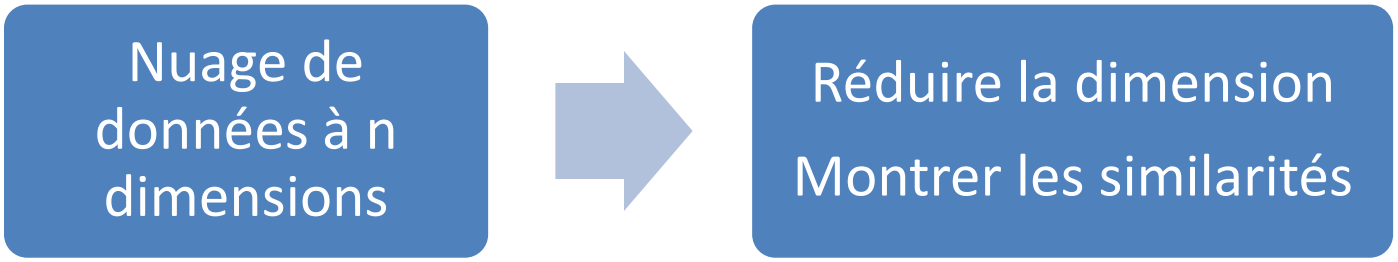
Inconvénients

- N'est pas adapté au datasets avec de hautes densités
- N'est pas adapté au datasets avec des points déconnectés
- Si le domaine n'est pas bien étudié, il est difficile de choisir une mesure de distance

Self Organizing Map (SOM)

SOM

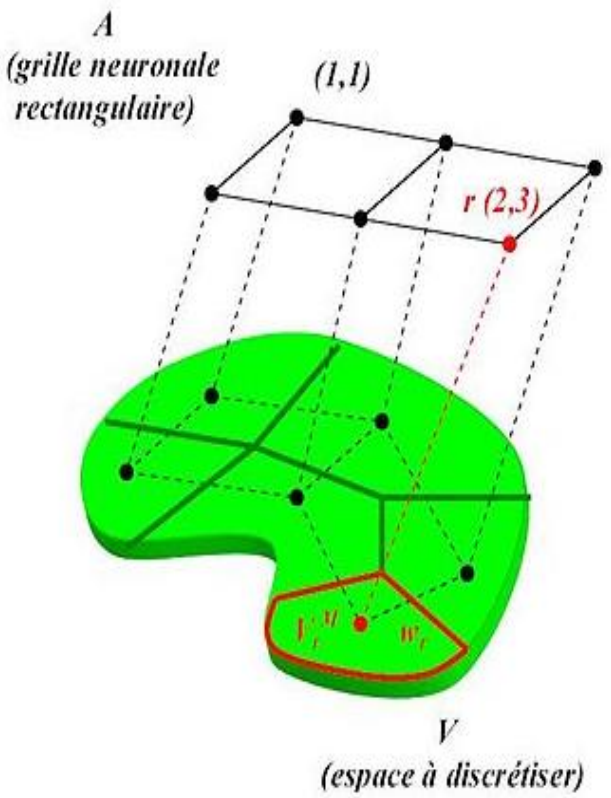
- SOM est une classe de réseaux de neurones pour l'apprentissage non supervisé.
- En pratique: Visualisation, discrétisation, clustering.
- Elle permet de représenter graphiquement des données de grandes dimension.
- Introduits pour la première fois par Teuvo Kohonen en 1982



Mapper les données

- Faire correspondre chaque donnée à une position dans l'espace 2D discret

- Les données les plus similaires sont mappés dans la même position



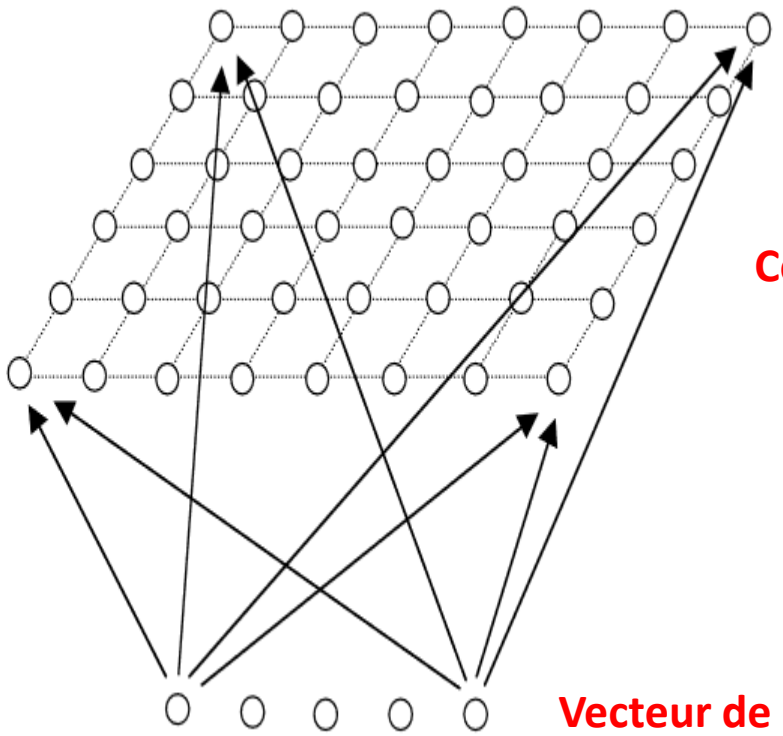
Source image: [3]

- L’objectif de SOM est de trouver un ensemble de nœuds (codebook en terminologie SOM) et affecter chaque donnée au nœud qui fournit sa meilleure approximation.
- Dans la terminologie des réseaux neuronaux, il existe un neurone associé à chaque noeud[2].
- les données similaires sont placées à proximité les uns des autres sur la grille SOM.

Principe de SOM

- Une grille SOM est composée de plusieurs nœuds de forme circulaire, rectangulaire ou plus généralement hexagonale,
- Chaque nœud possède :
  - Une position fixe sur la grille SOM.
  - Un vecteur de poids de la même dimension que l'espace d'entrée.
  - Chaque élément de donnée en entrée est "mappé" ou "lié" à un nœud sur la grille de la carte.
  - Un nœud peut représenter plusieurs échantillons d'entrées.

La grille SOM



Couche de calcul

Vecteur de donnée

Algorithme SOM

1: Initialiser les nœuds aléatoirement.

- 2: répéter

  - Sélectionner un élément

  - Déterminer le noeud le plus proche de l'objet

  - Mettre à jour ce noeud et les noeuds dans un voisinage spécifié.

$w_i(t+1) = w_i(t) + h(i,t)(x_i - w_i(t))$ si $i \in$ voisinage

$w_i(t+1) = w_i(t)$ si $i \notin$ voisinage

- avec $h(i,t) = \alpha(t).v(t)$

($\alpha(t)$: Le taux d'apprentissage. v(t) :la fonction de voisinage):

$$v(t)= \sigma_0 exp(-S^2/\sigma^2(t)$$

- *Tel que:* $S^2=d^2(i,i^*)$ *et* $\sigma^2(t)=\sigma_0 exp(t/t_{Max})$, $\alpha(t)=\alpha_0 exp(t/t_{Max})$. σ_0, t_{Max} *, et* α_0 *sont les paramètres du modèle.*

3- Faire décroître la taille de la zone de voisinage des nœuds gagnants (la zone qui contient les neurones subissant la transformation).

4- Faire décroître le coéfficient d'apprentissage, $\alpha(t)$,

5- Arrêt: Les nœuds ne changent pas beaucoup ou nombre d'itérations dépassé

6- Affecter chaque élément à son nœud le plus proche et retourner le résultat.

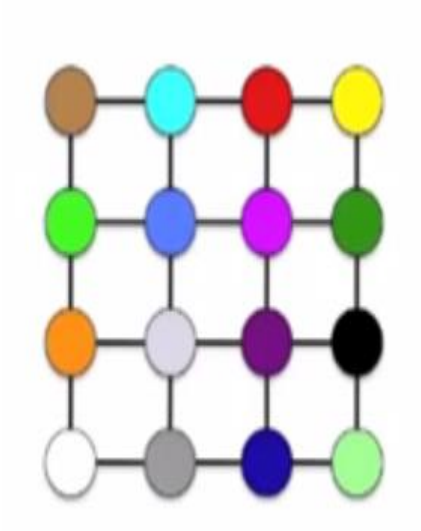
Algorithme détaillé

- Initialisation
- Choix du vecteur donnée
- Trouver le BMU
- Déterminer le voisinage
- Ajuster les poids
- Répéter jusqu'à convergence

Source image: [4]

Algorithme détaillé

- Initialisation
- Choix du vecteur donnée
- Trouver le BMU
- Déterminer le voisinage
- Ajuster les poids
- Répéter jusqu'à convergence



Source image: [4]

Algorithme détaillé

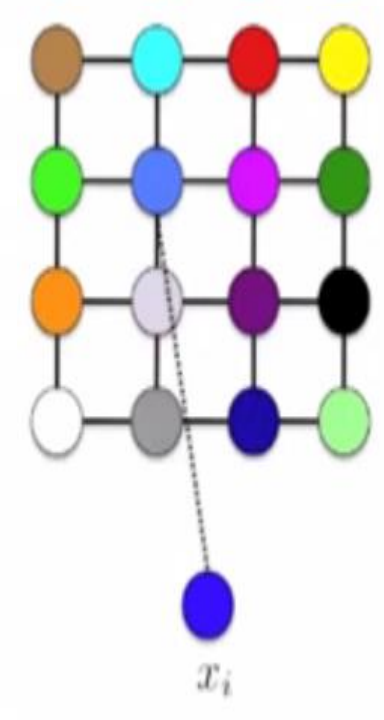
- Initialisation
- **Choix du vecteur donnée**
- Trouver le BMU
- Déterminer le voisinage
- Ajuster les poids
- Répéter jusqu'à convergence

- Aléatoirement
- Par balayage séquentiel de l'espace de données

Source image: [4]

Algorithme détaillé

- Initialisation
- Choix du vecteur donnée
- Trouver le BMU
- Déterminer le voisinage
- Ajuster les poids
- Répéter jusqu'à convergence

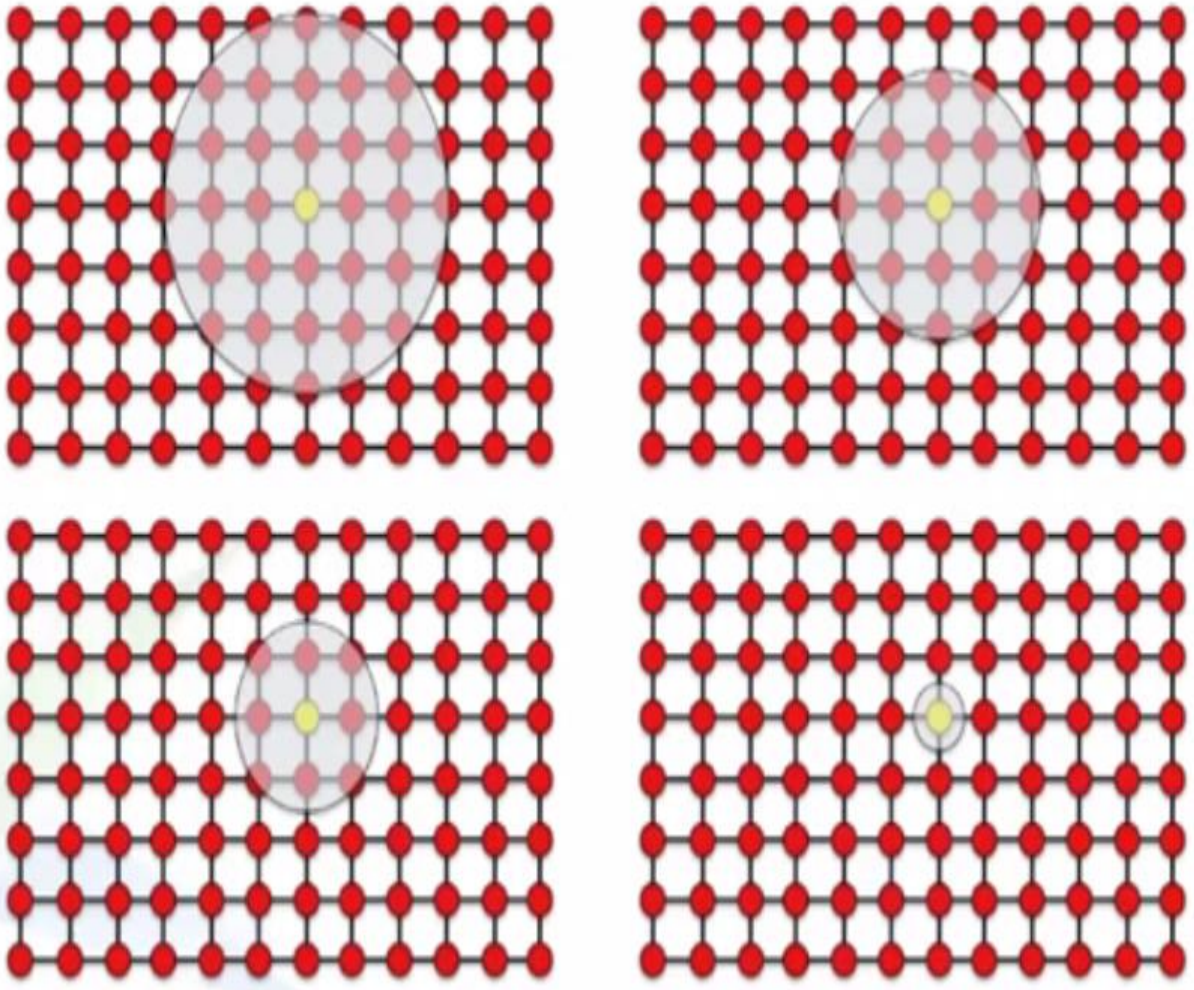


$$DistFromInput^2 = \sum_{i=0}^{i=n} (I_i - W_i)^2$$

Source image: [4]

Algorithme détaillé

- Initialisation
- Choix du vecteur donn
- Trouver le BMU
- Déterminer le voisina
- Ajuster les poids
- Répéter jusqu'à conve

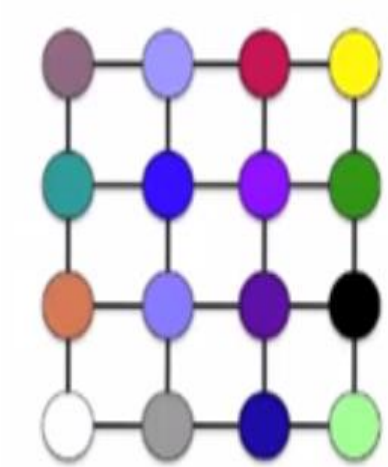


chaque iteration

Source image: [4]

Algorithme détaillé

- Initialisation
- Choix du vecteur donnée
- Trouver le BMU
- Déterminer le voisinage
- Ajuster les poids
- Répéter jusqu'à convergence



$$W(t+1)=W(t)+\Theta(t)L(t)(I(t)-W(t))$$

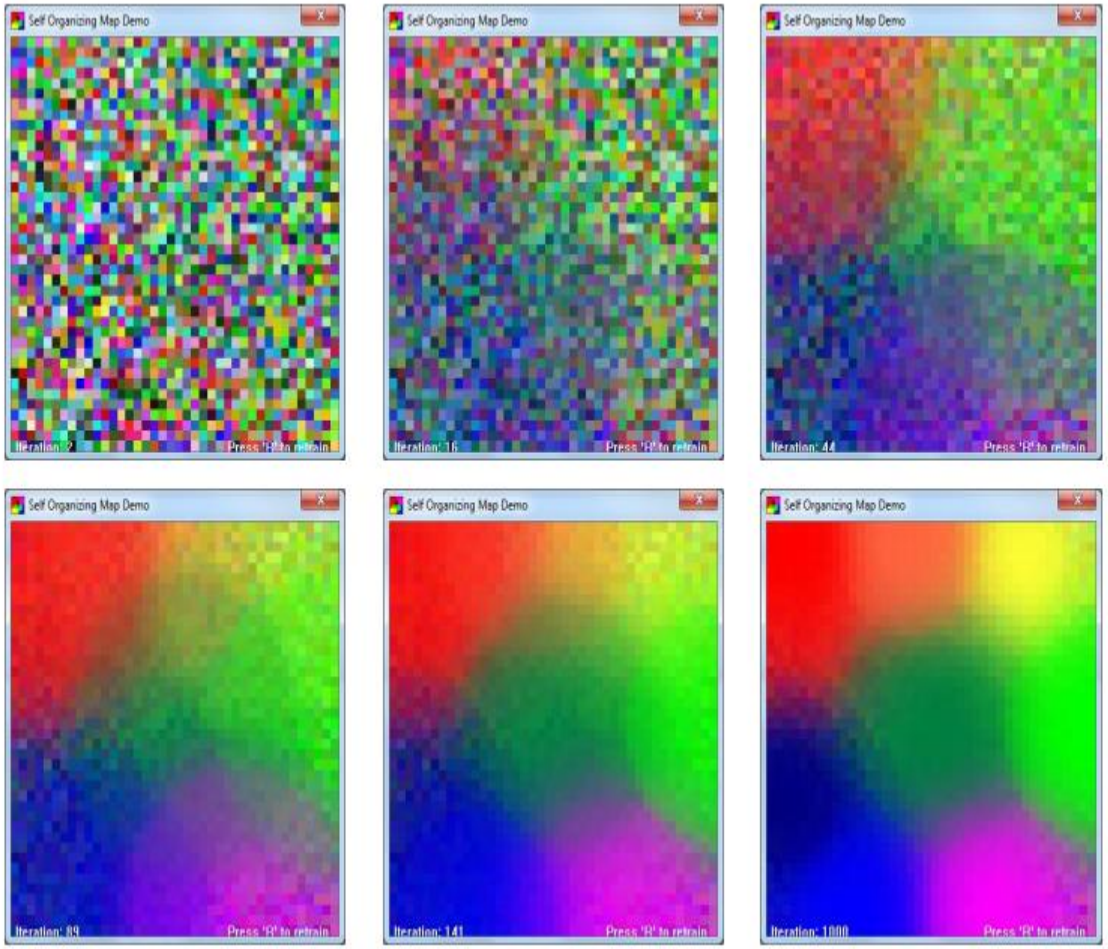
% d'apprentissage: décroit par itération

- Magnitude de l'ajustement proportionnelle à la distance

Source image: [4]

Algorithme détaillé

- Initialisation
- Choix du vecteur donné
- Trouver le BMU
- Déterminer le voisinage
- Ajuster les poids
- Répéter jusqu'à converg

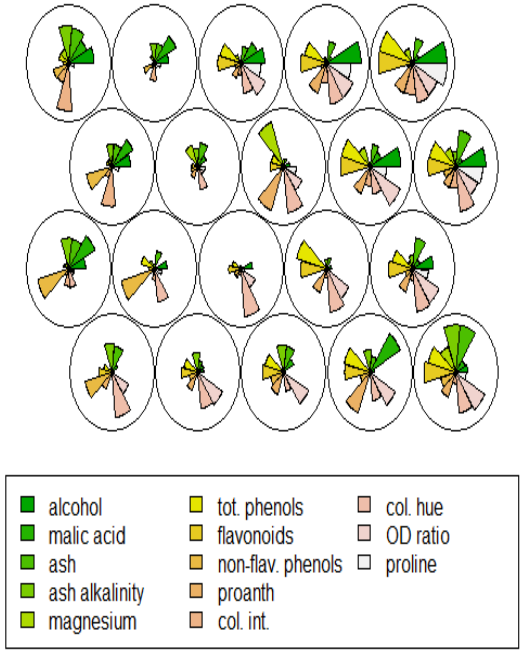


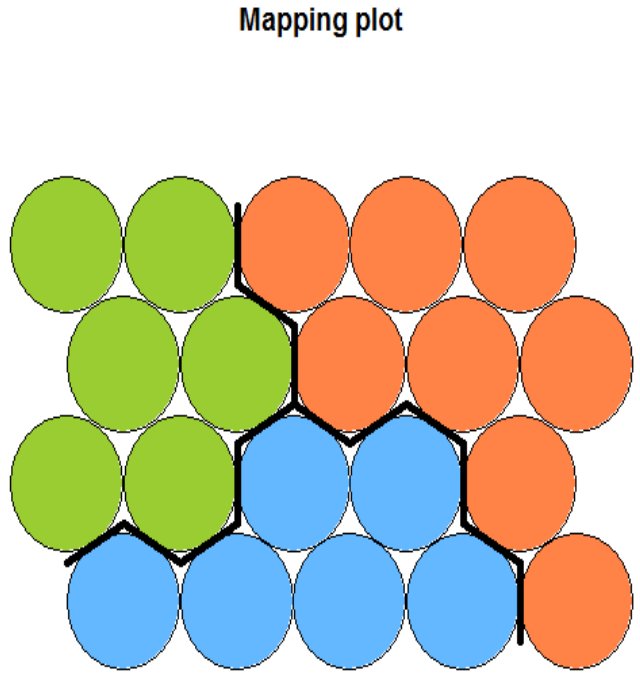
Source image: [4]

Clustering

- Appliquer le clustering hiérarchique sur la carte SOM:
- les nœuds adjacents de la carte topologique appartiendront à la même classe.
- Pouvoir traiter des très grandes bases, tout en bénéficiant des avantages de la CAH.

Exemple de clustering





SOM : Caractéristiques

Avantages

- Préserve la topologie dans les deux sens
- Processus d'apprentissage rapide

Inconvénients

- Voisinage fixe: on ne peut casser les liaisons entre neurones même pour mieux représenter les données discontinues.

% K -means

◎Choix du nombre de classes K **vs** nombre de neurones

◎Les deux sont moins efficaces que DBSCAN quand les données ne sont pas linéairement séparés

◎

Conclusion

Algorithm	No. of clusters found	Cluster quality (β)	Percent correct	CPU time (sec)
Art_data_1 artificial data set				
K-Means	2	1.7481265202402723	57.25	21
SOM	2	1.5714675087911916	63.66	50
DBSCAN	2	1.1745995850120072	99.99	168
Art_data_2 artificial data set				
K-Means	2	3.8910738021701077	99.9	4
SOM	2	3.8910738021701077	99.9	6
DBSCAN	2	3.8885998293751793	100.0	8
IRIS data set				
K-Means	3	8.624488765260416	99.9	4
SOM	3	8.624027263474284	88.67	3
DBSCAN	3	12.095993612888822	78.0	4

Source image: [8]