Apprentissage non supervisé
Partie 1

Analyse de données et Classification 2
ENSEEIHT - 3ème année Sciences du Numérique

Contact :
Sandrine.Mouysset@irit.fr
sandrine.mouysset@toulouse-inp.fr

Chaîne d'analyse des données

Classification non supervisée

vise à créer une partition (un ensemble de classes) d'un ensemble de données à partir des mesures de similarité entre ces données afin que des données appartenant à une même classe soit le plus semblable possible et des données appartenant à des classes soient le moins semblables possible.

On ne dispose pas de base d'apprentissage/connaissance *a priori*, la **classification non supervisée**

- définir des distances entre individus/variables,
- identifier des regroupements (agrégations/clusters).

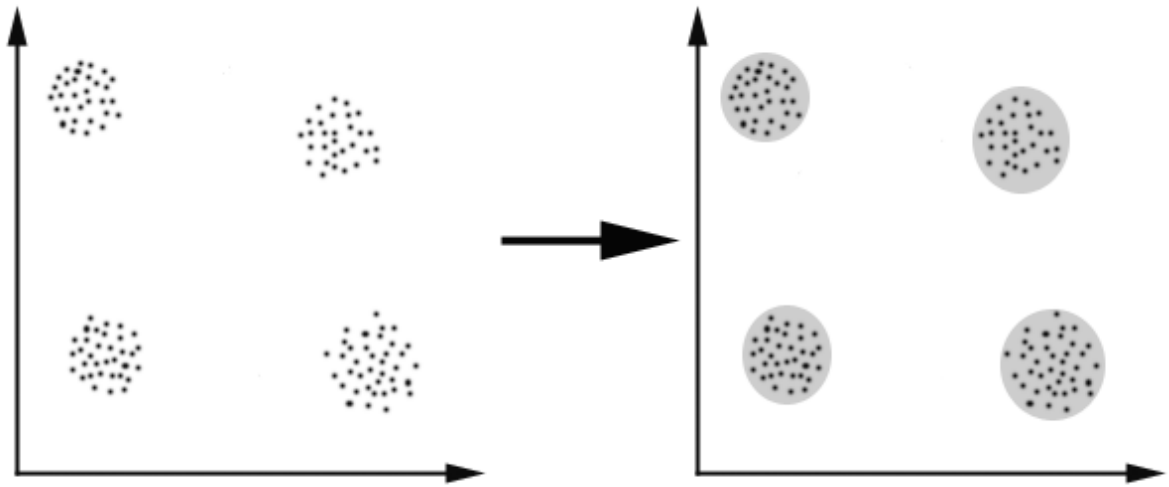


Figure: Exemple de classification non supervisée : diviser cet ensemble de points en 4 classes à partir de la distance entre les points

- Approches par partitionnement :
  - K-ppv
  - K-means
  - Classification hiérarchique
  - DBScan
  - Approche par noyau : Classification spectrale
  - Approche par graphe : Modularité

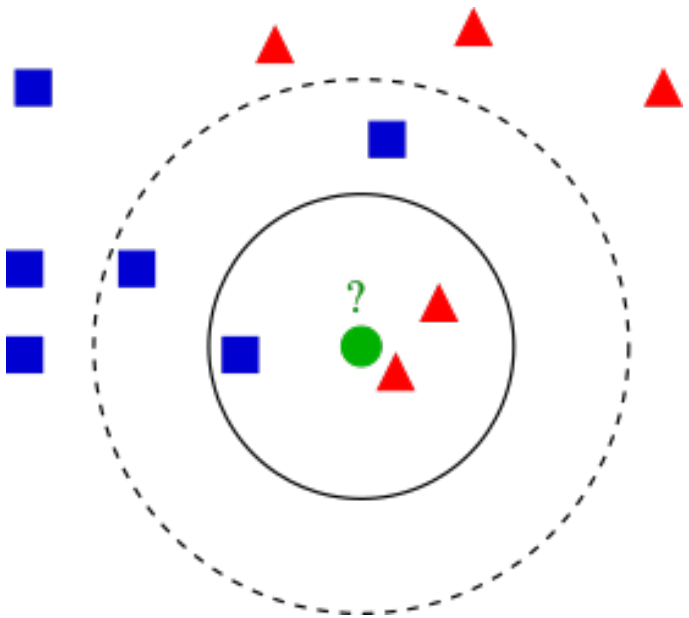
- Méthodes d'évaluation non supervisée de la partition :
  - Cohesion, Séparation
  - SSW, SSB et variantes

$\Rightarrow$ Toolbox *Scikit Learn* (Python)

Algorithm 1 Algorithme des k-ppv

Input : $Data_A$ et $Label_A$ ensemble de données et labels d'apprentissage, $Data_T$ ensemble de test.

1. Soit $x \in Data_T$ le point dont on cherche les k-ppv au sens d'une distance d.
   Calculer les distances $d(x, x_i)$, $\forall x_i \in Data_A$.
2. Trouver les k points $x_k \in Data_A$ plus proches voisins de x au sens de la distance d
3. Déterminer la classe C la plus représentée parmi les k plus proches voisins de x.
4. Assigner la classe C à la donnée x.



Algorithm 1 Algorithme des k-ppv

Input : $Data_A$ et $Label_A$ ensemble de données et labels d'apprentissage, $Data_T$ ensemble de test.

1. Soit $x \in Data_T$ le point dont on cherche les k-ppv au sens d'une distance d.
   Calculer les distances $d(x, x_i)$, $\forall x_i \in Data_A$.
2. Trouver les k points $x_k \in Data_A$ plus proches voisins de x au sens de la distance d
3. Déterminer la classe C la plus représentée parmi les k plus proches voisins de x.
4. Assigner la classe C à la donnée x.

Algorithm 2 Algorithme des 1-ppv (version non supervisée)

Input : S ensemble de données, distance seuil t.

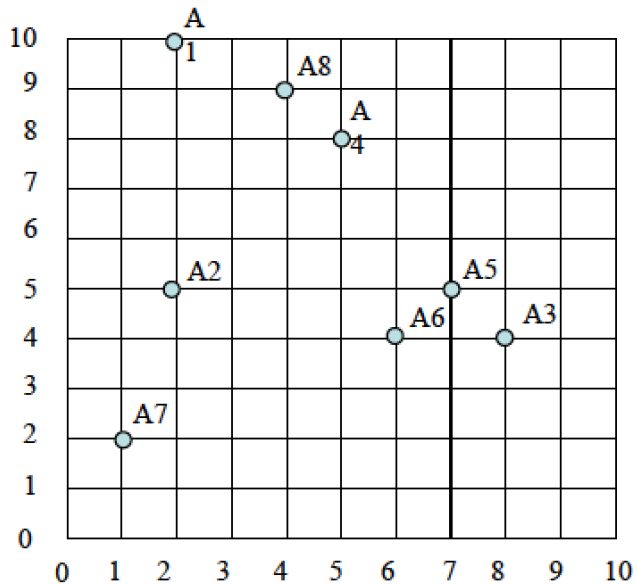
1. Soit $x \in S$ le point dont on cherche les 1-ppv au sens d'une distance d.
   Calculer les distances $d(x, x_i)$, $\forall x_i \in S$.
2. Trouver le point $x_j \in S$ plus proche voisin de x au sens de la distance d.
3. Si $d(x, x_j) < t$, alors définir une nouvelle classe C regroupant les 2 points les plus proches.
4. Assigner la classe C à la donnée x.

Exercice

On considère un jeu de points dans $\mathbb{R}^2$ suivant que l'on veut partitionner avec une méthode de classification non supervisée:

$$A1 = (2, 10),\ A2 = (2, 5),\ A3 = (8, 4),\ A4 = (5, 8)$$
$$A5 = (7, 5),\ A6 = (6, 4),\ A7 = (1, 2),\ A8 = (4, 9)$$

1. Calculer la matrice des distances euclidiennes au carré entre les points $(A_i)_{i=1..8}$.
2. Réaliser la méthode des K-ppv avec $k = 1$ et un seuil de 16.



Partitionner un ensemble de données en k classes représentées par les k centres, notés $\mathcal{C} = \{\mathcal{C}^1 \dots \mathcal{C}^k\}$.
On associe alors à chaque point/donnée le centre le plus proche au sens d'une certaine distance.

Dans le cadre non supervisé on cherche généralement à partitionner l'ensemble de données de manière à minimiser la variance intra-classe, qui se traduit par l'**énergie, entropie (ou variance intra-classe)** :

$$E = \frac{1}{n} \sum_{i=1}^{k} \sum_{x \in \mathcal{C}^i} \|x - m_i\|^2$$

où $\begin{cases} S = \{x_1 \dots x_n\} \in \mathbb{R}^p \text{ ensemble des données} \\ \#S = n \\ \mathcal{C}^i = \text{classe i } \forall i \in 1 \dots k \\ m_i = \text{centre de la classe i , } \forall i \in \{1 \dots k\} \end{cases}$

Soit $S = \{x_1 \dots x_n\}, x_i \in \mathbb{R}^p$. On veut partitionner S en k classes $\mathcal{C}^i$ afin de minimiser les distances intra-classes (ou l'entropie).

Algorithm 2 Algorithme kmeans

1. Initialisation des centres $m_i^{(0)}, i \in 1 \dots k$
2. Répéter jusqu'à convergence :
   - assigner à chaque donnée la classe la plus proche :

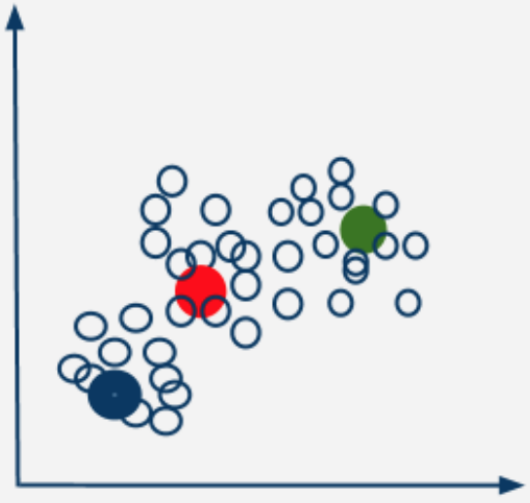
$$C_i^{(t)} = \{x_j : \|x_j - m_i\| \leq \|x_j - m_d\| \forall d \in 1 \dots k\}$$

   - mettre à jour :

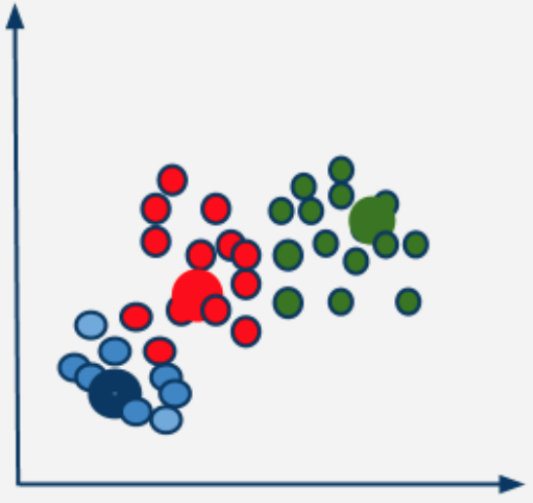
$$m_i^{(t+1)} = \frac{1}{\#\mathcal{C}} \sum_{x_j \in C_i^{(t)}} x_j$$

3. Fin

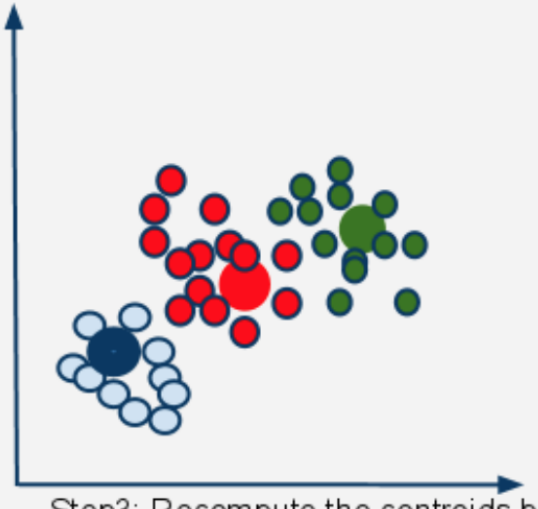
Algorithme K-means



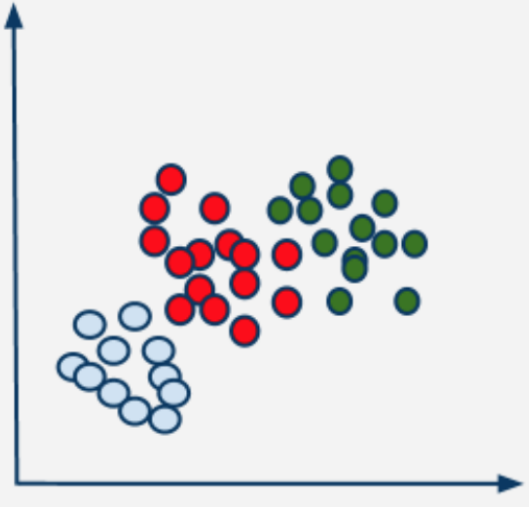
Step1: Lets assume K=3. Choose the cluster centroids randomly



Step2: Compute the distance from the centroids and assign elements to the cluster of the nearest centroid



Step3: Recompute the centroids based on the assignments from the previous step... Repeat step2 and step 3 until clusters converge



Final set of converged clusters

K-means : limites

Quelques points cruciaux !

- Bien choisir les centres à l'initialisation pour éviter les convergences locales;
- Nombre de classes à déterminer;
- Méthode de séparation linéaire.

K-means : limites

Quelques points cruciaux !

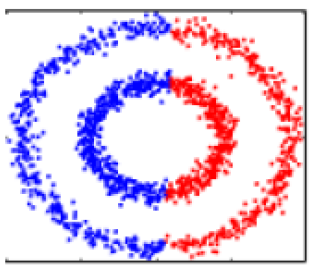
- Bien choisir les centres à l'initialisation pour éviter les convergences locales;
  $\rightarrow$ heuristiques sur la distribution des points;
- Nombre de classes à déterminer;
- Méthode de séparation linéaire.

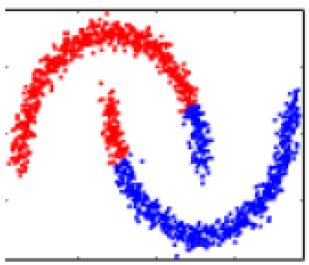
Quelques points cruciaux !

- Bien choisir les centres à l'initialisation pour éviter les convergences locales;
  $\rightarrow$ heuristiques sur la distribution des points;
- Nombre de classes à déterminer;
  $\rightarrow$ varier le nombre de classes et étudier l'énergie;
- Méthode de séparation linéaire.

Quelques points cruciaux !

- Bien choisir les centres à l'initialisation pour éviter les convergences locales;
  $\rightarrow$ heuristiques sur la distribution des points;
- Nombre de classes à déterminer;
  $\rightarrow$ varier le nombre de classes et étudier l'énergie;
- Méthode de séparation linéaire.



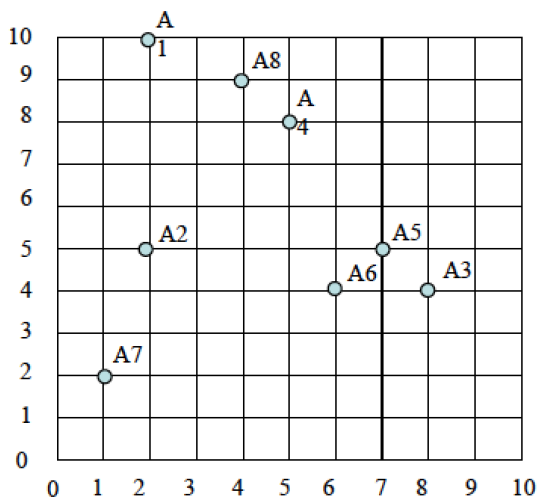


Exercice

On considère un jeu de points dans $\mathbb{R}^2$ suivant que l'on veut partitionner avec une classification hiérarchique :

$$A1 = (2, 10),\ A2 = (2, 5),\ A3 = (8, 4),\ A4 = (5, 8)$$
$$A5 = (7, 5),\ A6 = (6, 4),\ A7 = (1, 2),\ A8 = (4, 9)$$

1. Réaliser une itération de la classification par K-means en prenant des centres initiaux les points A_1, A_4 et A_7.
2. Obtient-on la même partition que pour les Kppv ?



	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
A_1	0	25	72	13	50	52	65	5
A_2		0	37	18	25	17	10	20
A_3			0	25	2	4	53	41
A_4				0	13	17	52	2
A_5					0	2	45	25
A_6						0	29	29
A_7							0	58
A_8								0

Contrairement à la méthode des K-moyennes, les méthodes de clustering hiérarchique ne dépendent :

- ni d'un nombre K de clusters prédéfini;
- ni d'une configuration initiale (choisie souvent arbitrairement).

La seule chose à fournir est une **mesure de dissimilarité** entre groupes (disjoints) d'observations. Naturellement, une telle mesure est bâtie sur les dissimilarités prises 2 à 2 au sein de chaque groupe.

Soient G et H deux groupes d'observations/de données.

Dissimilarité moyenne/Group Average :

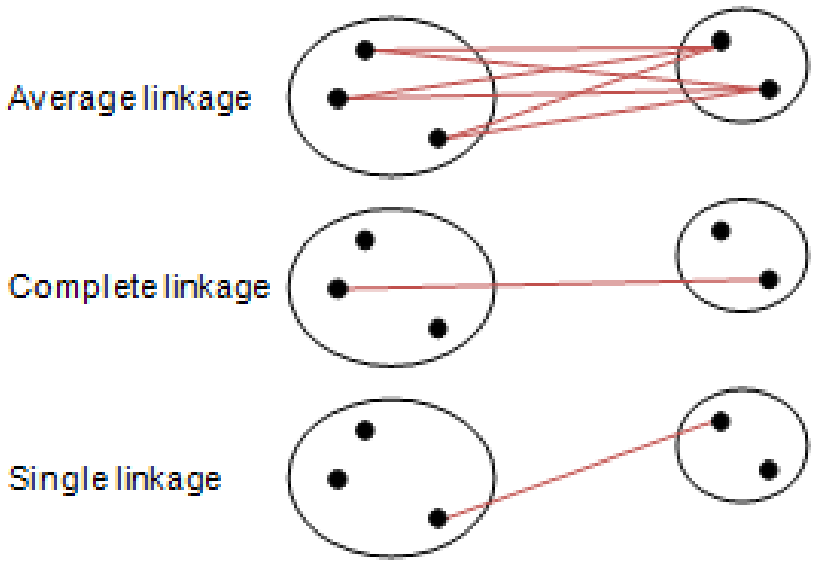
$$d_{GA}(G,H) = \frac{1}{|G||H|} \sum_{i \in G, i' \in H} d_{ii'}$$

Dissimilarité du lien complet/complete linkage

$$d_{CL}(G,H) = \max_{i \in G,\ i' \in H} d_{ii'}$$

Dissimilarité du lien simple/single linkage :

$$d_{SL}(G,H) = \min_{i \in G,\ i' \in H} d_{ii'}$$



Les méthodes de classification hiérarchique (CH) fournissent des représentations pour lesquelles chaque cluster au sein de la hiérarchie et à un certain niveau est créé par fusion de clusters à un niveau plus fin.
Au niveau le plus fin, chaque cluster est un singleton; au niveau le plus haut, un seul cluster regroupe toutes les données disponibles.

Définition : Une partition Π_i est plus fine que partition Π_j ssi tout bloc de Π_j est un bloc de Π_i ou est l'union de plusieurs blocs de Π_i.

Les stratégies pour la classification hiérarchique sont :

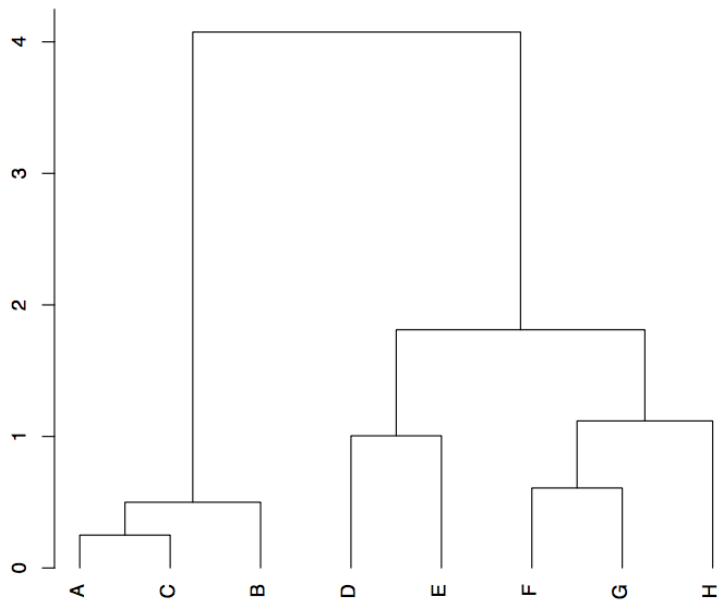
- **ascendantes/agglomératives/bottom-up :** démarrer du bas niveau (une donnée=une classe) et opèrer des fusions récursives de clusters en plus gros clusters;
- **descendantes/séparatrices/top-down :** commencer par séparer la classe regroupant toutes les données en 2 clusters de dissimilarités maximales.

Etant donnée une représentation hiérarchique issue d'un CH, c'est à **l'utilisateur** de décider quel est le niveau naturel de partitionnement/clustering qui correspond à la structure (inconnue) des données.

On peut afficher l'arbre binaire de sorte que la hauteur de chaque noeud soit équivalente à la valeur de la dissimilarité intercluster entre ces deux enfants. Ce type d'affichage est appelé **dendrogramme**.

Exemple de hiérarchie sur un ensemble initial de données :
$S = \{a, b, c, d, e, f, g, h\}$.

On parle aussi de chaîne de partitions
$\Pi_8 : (a, b, c, d, e, f)$
$\Pi_7 : (a, b, c)\ (d, e, f, g, h)$
$\Pi_6 : (a, b, c)\ (d, e)(f, g, h)$
$\Pi_5 : (a, b, c)\ (d, e)(f, g)\ (h)$
$\Pi_4 : (a, b, c)\ (d)\ (e)\ (f, g)\ (h)$
$\Pi_3 : (a, b, c)\ (d)\ (e)\ (f)\ (g)\ (h)$
$\Pi_2 : (a, c)\ (b)\ (d)\ (e)\ (f)\ (g)\ (h)$
$\Pi_1 : (a)\ (b)\ (c)\ (d)\ (e)\ (f)\ (g)\ (h)$



Exercice :

A partir de la matrice des distances entre les points suivante :

	A	B	C	D
A	0	1	4	5
B		0	2	6
C			0	3
D				0

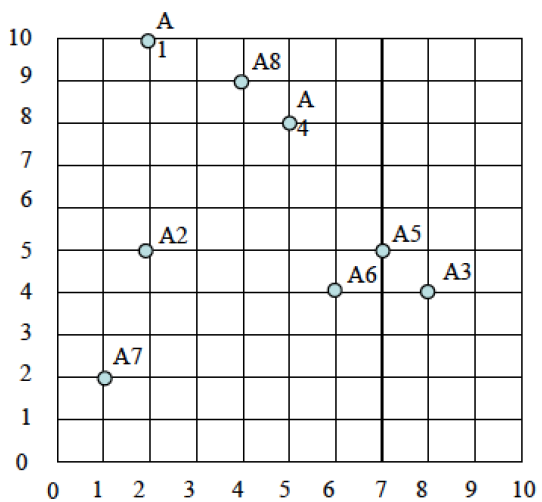
1. Réaliser la classification hiérarchique des données par **lien simple**
2. Réaliser la classification hiérarchique des données par **Lien complet**
3. Représenter ces partitions par dendrogrammes.

Exercice

On considère un jeu de points dans $\mathbb{R}^2$ suivant que l'on veut partitionner avec une classification hiérarchique :

$$A1 = (2, 10),\ A2 = (2, 5),\ A3 = (8, 4),\ A4 = (5, 8)$$
$$A5 = (7, 5),\ A6 = (6, 4),\ A7 = (1, 2),\ A8 = (4, 9)$$

1. Réaliser une classification hiérarchique par lien simple et lien complet.
2. Représenter les dendrogrammes respectifs.
3. Pour quelle(s) mesure(s) de dissimilarité obtient-on la même partition que pour les Kppv et Kmeans?



	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
A_1	0	25	72	13	50	52	65	5
A_2		0	37	18	25	17	10	20
A_3			0	25	2	4	53	41
A_4				0	13	17	52	2
A_5					0	2	45	25
A_6						0	29	29
A_7							0	58
A_8								0

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) est un algorithme de partitionnement de données proposé en 1996 par M. Ester *et al* basé sur la densité estimée des clusters.

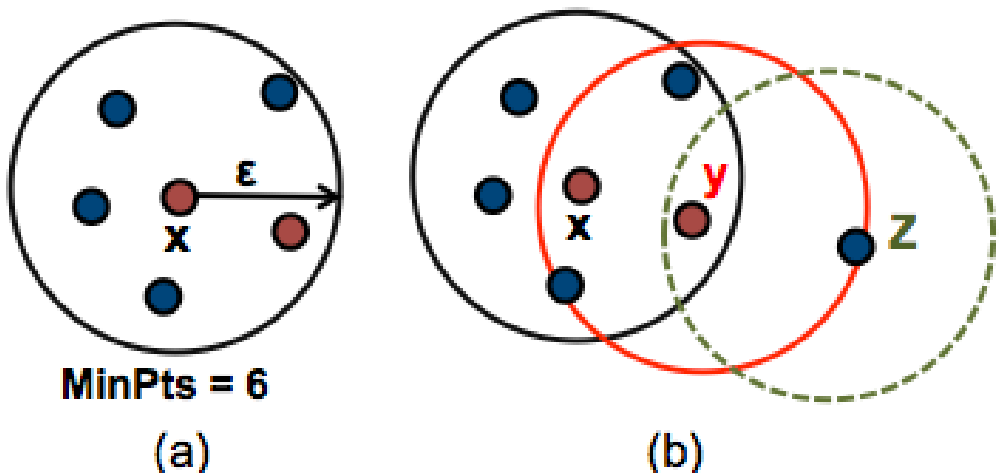
L'algorithme DBSCAN utilise en entrée 2 paramètres :

- la distance ϵ
- le nombre minimum de points *MinPts* devant se trouver dans un rayon pour que ces points soient considérés comme un cluster.

Les paramètres d'entrée donnent donc une estimation de la densité de points des clusters.

L'idée de base de l'algorithme est ensuite, pour un point donné :

- de récupérer son ϵ-voisinage
- de vérifier qu'il contient bien *MinPts* points ou plus
- Si oui, ce point est alors considéré comme faisant partie d'un cluster, sinon il reste isolé.
- On parcourt ensuite l'ϵ-voisinage de proche en proche afin de trouver l'ensemble des points du cluster.



Soit $S = \{x_1 \dots x_n\}, x_i \in \mathbb{R}^p$. DBSCAN requiert 2 paramètres : ϵ et le nombre minimum de points requis pour constituer un cluster *MinPts*.

Algorithm 4 Algorithme DBSCAN

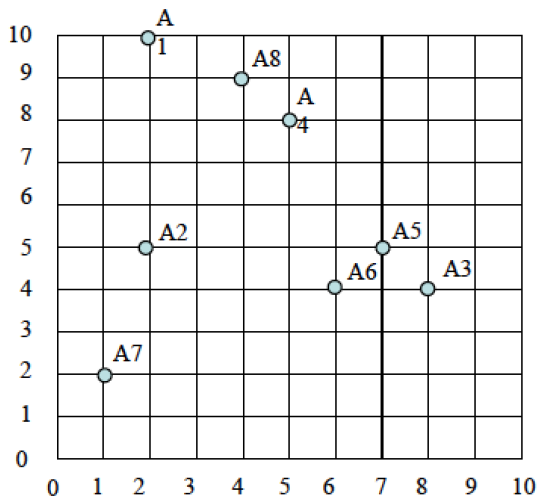
1. Commencer par un points arbitraire non visité.
2. Extraire le voisinage de ce point en utilisant ϵ
   - S'il y a suffisament de points (au moins $MinPts$) autour de ce point arbitraire alors le processus de clustering commence et le point est considéré comme "visité" sinon ce point est labelisé comme "bruit"
   - Si un point est trouvé pour faire partie du cluster alors son ϵ voisinage est aussi une part du cluster et la procédure ci dessus est répétée pour tous les ϵ voisinage des points. Cette étape est répétée jusqu'à ce que tous les points du cluster soient déterminés.
3. Un point non visité est ensuite recherché et le processus est appliqué sur lui aboutissant à un cluster ou du bruit.
4. Le processus continue jusqu'à ce que tous les points soient visités.

Exercice

On considère un jeu de points dans $\mathbb{R}^2$ suivant que l'on veut partitionner avec une classification hiérarchique :

$$A1 = (2, 10),\ A2 = (2, 5),\ A3 = (8, 4),\ A4 = (5, 8)$$
$$A5 = (7, 5),\ A6 = (6, 4),\ A7 = (1, 2),\ A8 = (4, 9)$$

1. En fixant $\epsilon = 4$ et $MinPts = 2$, quelle partition obtient-on par DBScan ?
2. Obtient-on le même résultat avec $\epsilon = 10$?



	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
A_1	0	25	72	13	50	52	65	5
A_2		0	37	18	25	17	10	20
A_3			0	25	2	4	53	41
A_4				0	13	17	52	2
A_5					0	2	45	25
A_6						0	29	29
A_7							0	58
A_8								0