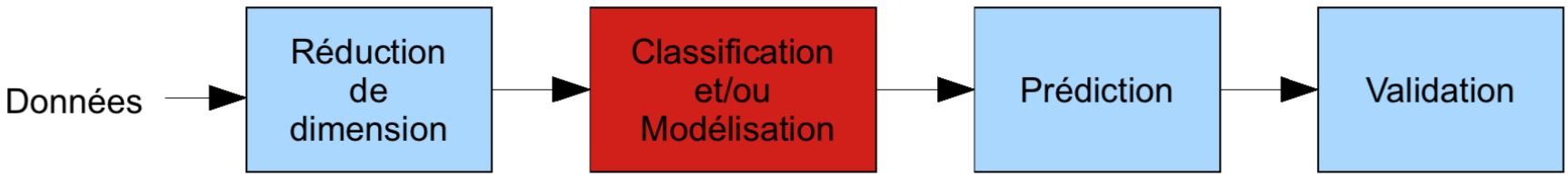
Apprentissage Automatique

M1 IAFA-SECIL
Université Paul Sabatier

Contacts :
Sandrine.Mouysset@irit.fr
thomas.pellegrini@irit.fr

Apprentissage non supervisé



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** sert à

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

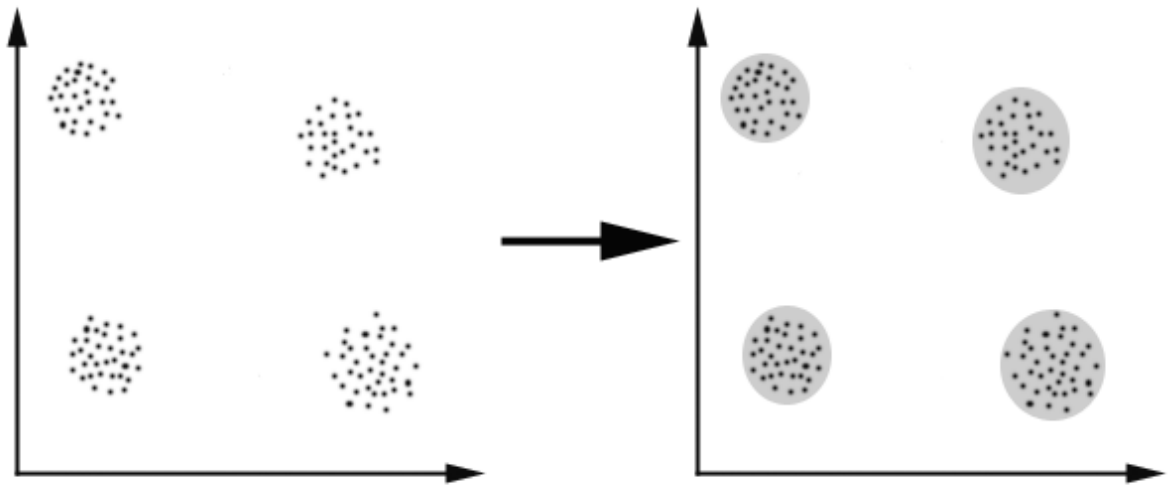


Figure: Exemple de classification non supervisée : diviser cet ensemble de points en 4 classes à partir de la distance entre les points

- Approches par partitionnement :
  - K-ppv
  - K-means
  - Classification hiérarchique

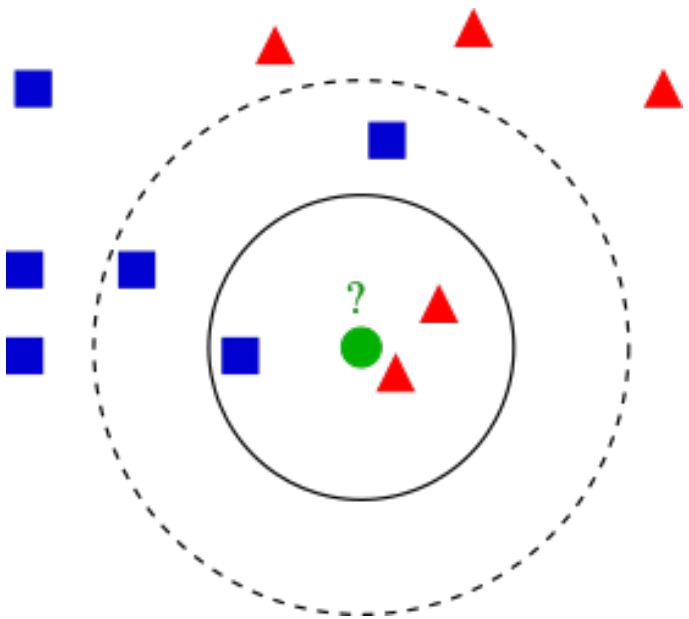
- Méthodes d'évaluation non supervisée de la partition :
  - Cohesion, Séparation
  - SSW, SSB et variantes

$\Rightarrow$ 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.

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 :

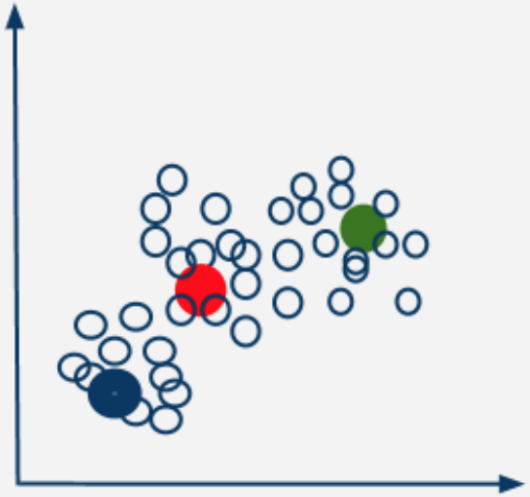
$$\mathcal{C}_i^{(t)} = \{x_j : \|x_j - m_i\| \leq \|x_j - m_d\| \forall d \in 1 \dots k\}$$

   - mettre à jour :

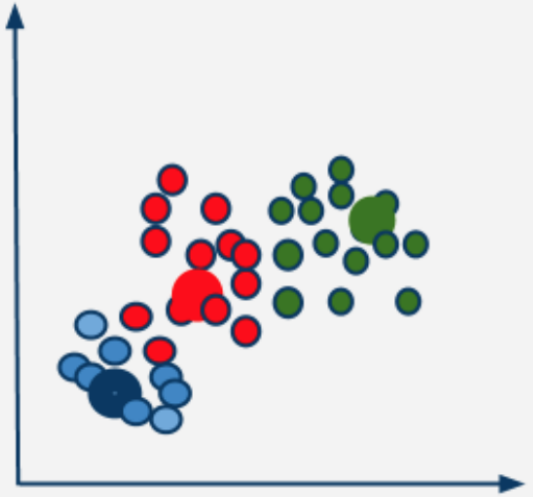
$$m_i^{(t+1)} = \frac{1}{\#\mathcal{C}} \sum_{x_j \in \mathcal{C}_i^{(t)}} x_j$$

3. Fin

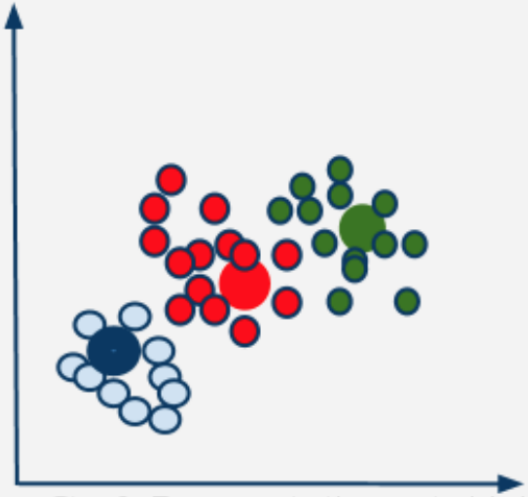
Algorithme K-means



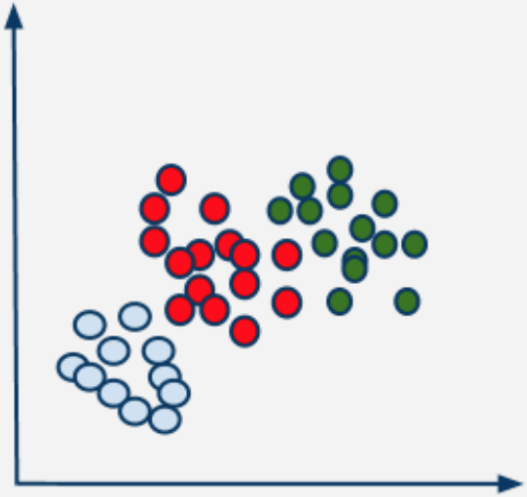
Step1: Lets assume K=3. Choose the cluster centroids randomly



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

Quelques points cruciaux !

- Bien choisir les centres à l'initialisation pour éviter les convergences locales;
- 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;

- 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 !

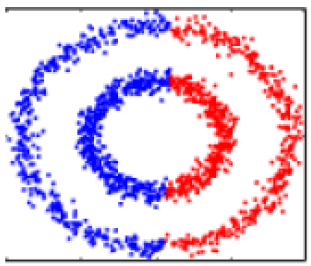
- Bien choisir les centres à l'initialisation pour éviter les convergences locales;

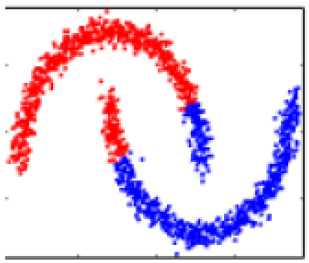
  $\rightarrow$ heuristiques sur la distribution des points;

- Nombre de classes à déterminer;

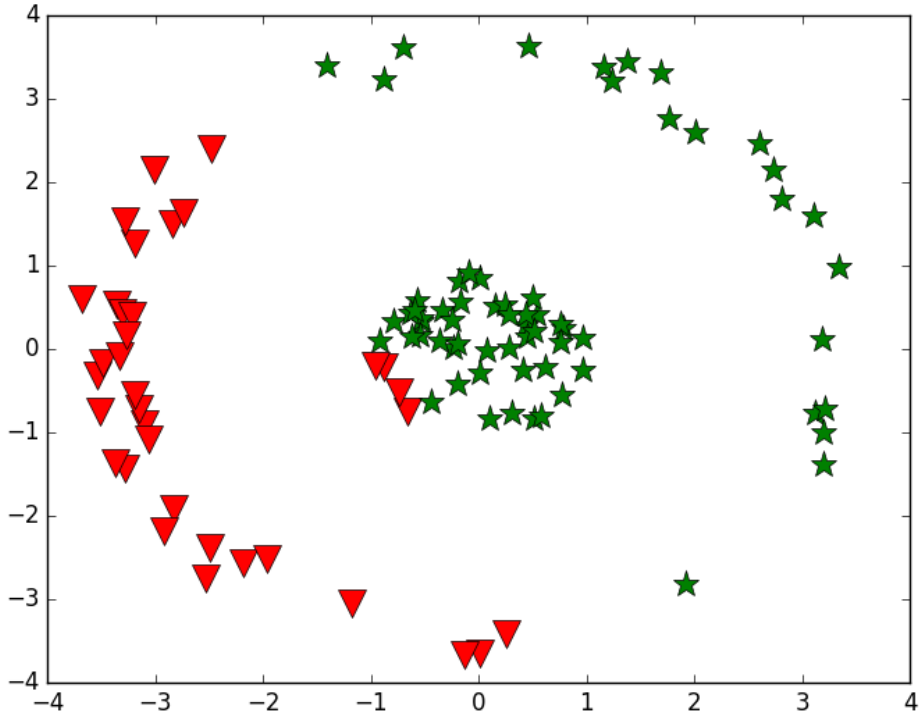
  $\rightarrow$ varier le nombre de classes et étudier l'énergie;

- Méthode de séparation linéaire.





K-means : limites



- Pureté = 0.75
- SSE = 359.80
- Satisfaisant ?

- **Idée :** projeter les données sur un espace à plus grande dimension grâce à une fonction "noyau" et appliquer ensuite l'algorithme k-moyennes dans cet espace

- **Attention :** cela implique de calculer et avoir en mémoire une matrice de taille $N \times N$

- **Exemples** de fonctions à noyaux :

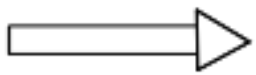
  - Noyau polynomial de degré h : $K(x_i, x_j) = (x_i x_j + 1)^h$
  - Noyau gaussien radial (RBF) : $K(x_i, x_j) = e^{-||x_i - x_j||^2 / 2\sigma^2}$

Exemple : noyau gaussien radial

données originales :
100 points en 2 dim

X_1	X_2
-0.524	0.339
0.152	0.515
...	
2.078	2.865
3.149	-1.811

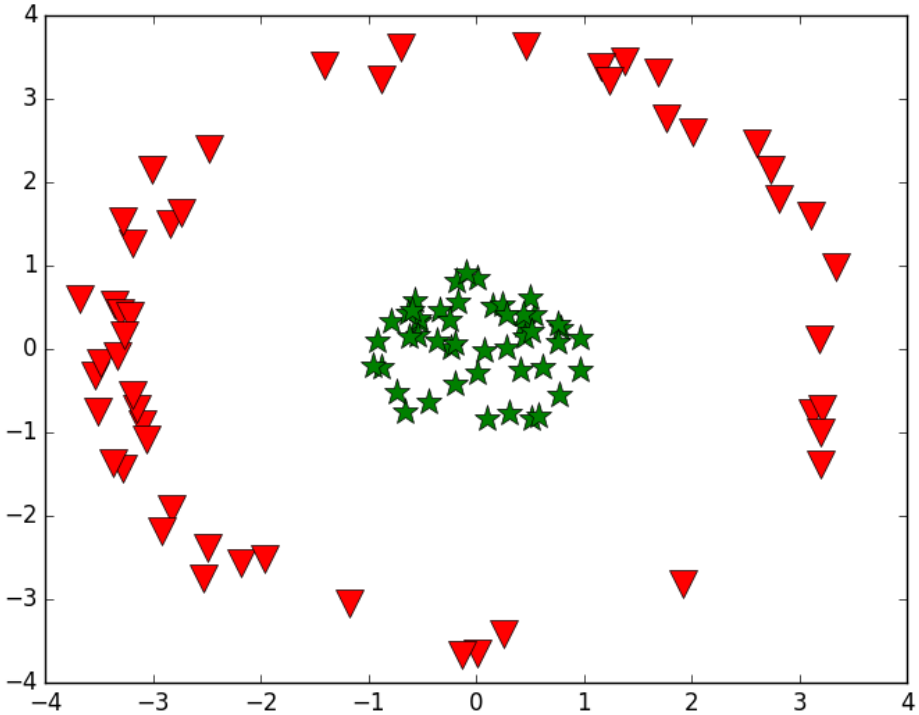
noyau RBF



Matrice 100x100

$$\begin{bmatrix} 0.000 & 0.784 & 0.294 & \dots & 0.004 \\ 0.784 & 0.000 & 0.377 & \dots & 0.001 \\ \dots & & & & \\ & \dots & & & \\ & & \dots & & \\ 0.004 & \dots & & & 0.000 \end{bmatrix}$$

Exemple : noyau gaussien radial



- Pureté = 1.0
- SSE = 309.81
- Satisfaisant ?

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 :

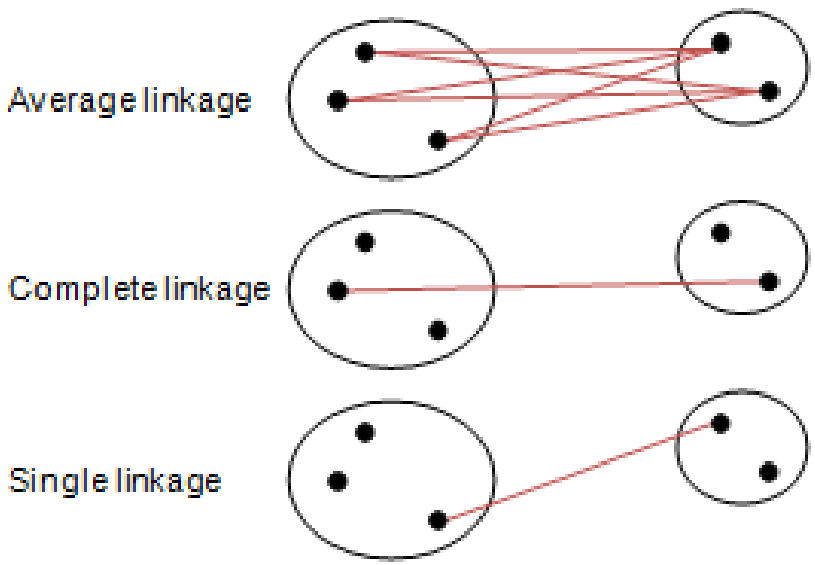
$$d_{GA}(G,H) = \frac{1}{|G||H|} \sum_{i \in G, i' \in H} d_{ii'}$$

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 :

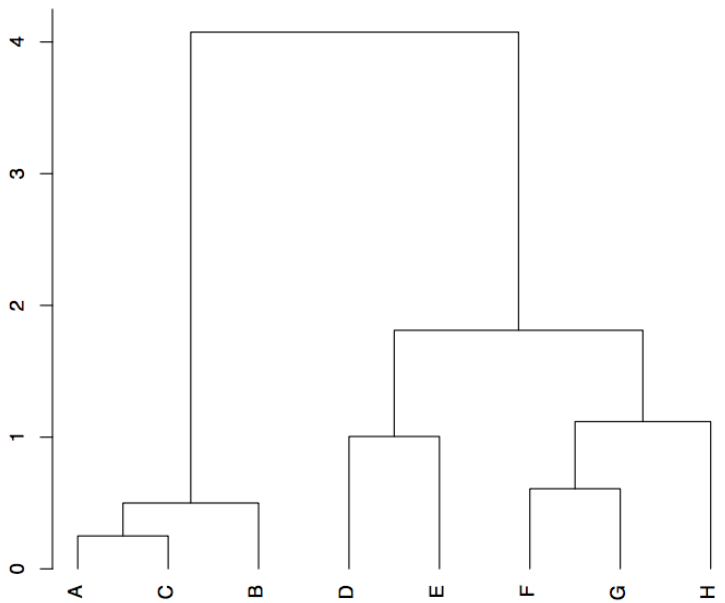
- **ascendantes/agglomératives/bottom-up :** démarrer du bas niveau (une donnée=une classe) et opèrer des fusions récursives de clusters en plus gros clusters;
- **descendantes/séparatrices/top-down :** commencer par séparer la classe regroupant toutes les données en 2 clusters de dissimilarités maximales.

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 :

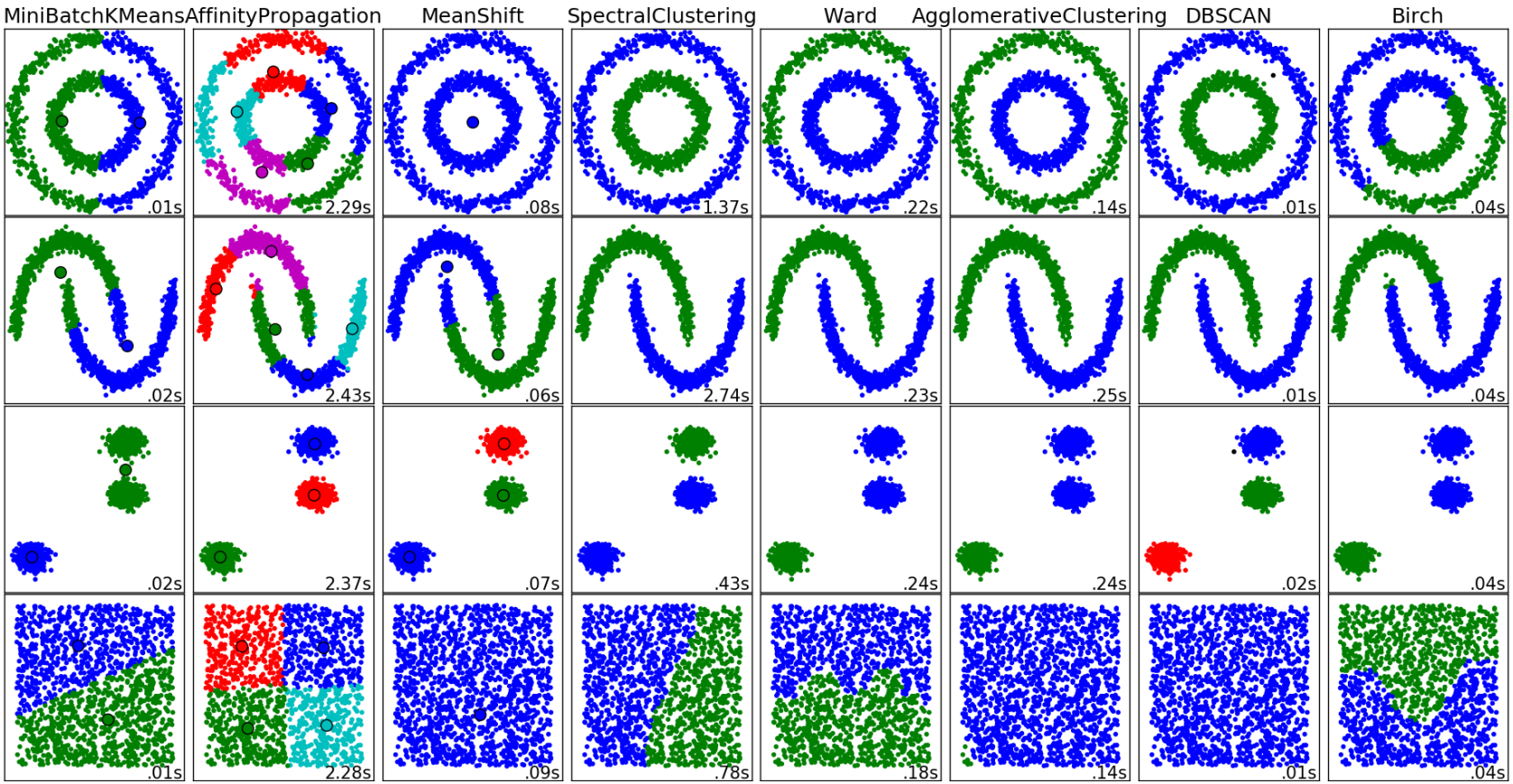
A partir de la matrice des distances entre les points suivante :

	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.

- Une fois qu'un regroupement ou une division est réalisé, pas de possibilité de revenir en arrière
- **Complexité en** $O(n^2)$: problème de passage à l'échelle
- Des **variantes** plus compliquées existent, comme par exemple BIRCH : Balanced Iterative Reducing and Clustering using Hierarchies, (Zhang, et al., 1996).
- **Principe de BIRCH :** clustering multi-niveaux qui construit un arbre de features. Les feuilles de l'arbre sont l'objet d'un clustering, par exemple un k-moyennes.

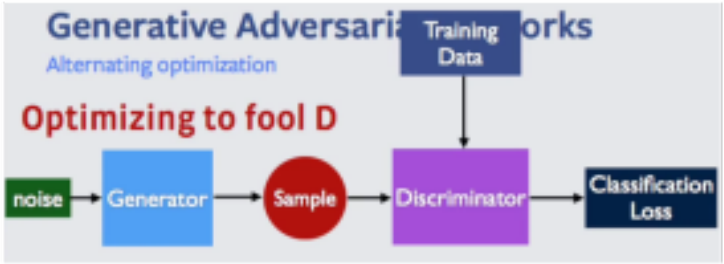
Beaucoup d'autres techniques de clustering



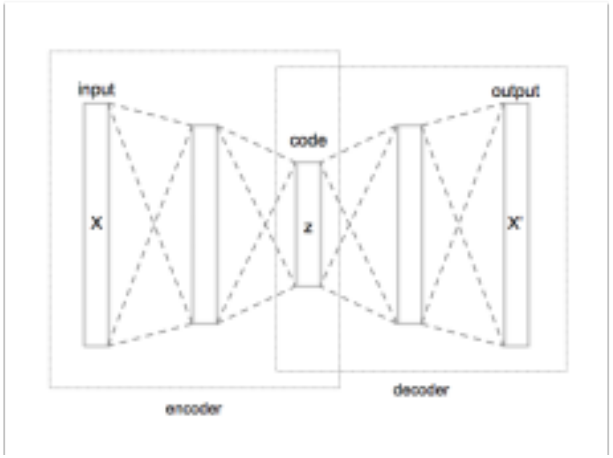
http://scikit-learn.org/stable/modules/clustering.html

cartes auto-organisatrices

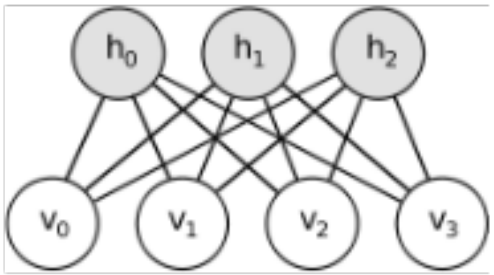
réseaux adversaires générateurs



auto-encodeurs



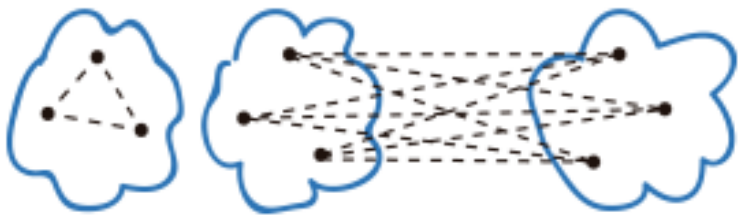
machines de Boltzmann



Comment évaluer les méthodes non supervisées ?

Cohésion et Séparation

- la cohésion d'une classe mesure comment les objets d'un cluster sont étroitement liés;
- la séparation d'une classe mesure comment une classe est distincte ou bien séparée de l'autre.



Pour une classe C_i,

$$Cohesion(C_i) = \sum_{x \in C_i, y \in C_i} prox(x, y)$$

$$Separation(C_i, C_j) = \sum_{x \in C_i, y \in C_j} prox(x, y), \ \ i \neq j$$

où *prox* est une fonction de proximité (similarité, dissimilarité, distances...)

- Sum of Squared Error (SSE) ou Sum of Squared Errors Within Cluster (SSW) correspond à la mesure de cohésion où *prox* est la distance euclidienne (à minimiser)

$$SSE(C_i) = \sum_{x \in C_i} d(c_i, x)^2 = \frac{1}{2|C_i|} \sum_{x \in C_i} \sum_{y \in C_i} d(x, y)^2$$

où x est un élement de C_i, c_i est le centre de C_i et $|C_i|$ le cardinal de la classe C_i.

- Between group Sum of Squares (BSS) basé sur la séparation (à maximiser)

$$BSS = \sum_{i=1}^{K} |C_i| d(c_i, c)^2$$

où c est le centroid de l'ensemble de données.

- Coefficient Calinski-Harabasz (CH) est un ratio (à maximiser) entre la variance inter-classe et la variance intra-classe :

$$CH = \frac{\frac{BSS}{K-1}}{\frac{SSE}{K}}$$

- Indice de Hartigan basé sur le logarithme du ratio BSS/SSE :

$$H = \log\left(\frac{BSS}{SSE}\right)$$

- Indice de Dunn est le ratio de la plus petite distance entre les données de différents groupes et de la plus grande distance entre les groupes.
  Pour une classe C_i,

$$D_i = \frac{\min_{1<j<K, i\neq j} \delta(C_i, C_j)}{\max_{1<l<K} \Delta(C_l)}$$

avec $\Delta(C_i) = \max_{x,y\in C_i} d(x,y)$ et $\delta(C_i, C_j) = \min_{x\in C_i, y\in C_j} d(x,y)$.

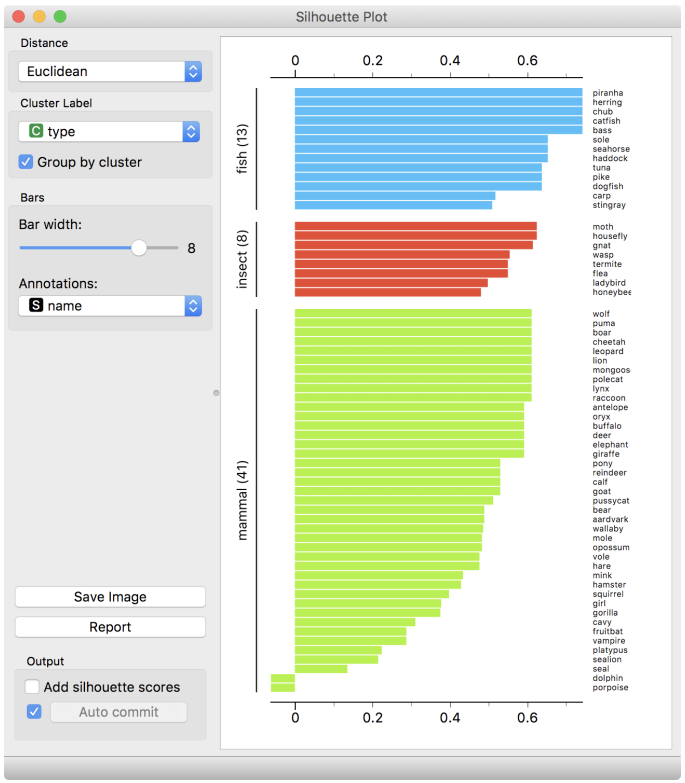
- Silhouette est le coefficient le plus connu combinant les mesures de cohésion et de séparation.

  Le **calcul du coefficient de silhouette** à un point donné comprend les trois étapes suivantes, pour chaque donnée $x \in C_i$,

  1. calcul de la distance moyenne entre x et tous les autres points au sein de la même classe :
  $$a(x) = \frac{1}{|C_i| - 1} \sum_{y \in C_i, x \neq y} d(x, y)$$
  2. calcul de la dissimilarité moyenne minimale entre le point x et une autre classe C_k $(i \neq k)$:
  $$b(x) = \min_{k \neq i} \frac{1}{|C_k|} \sum_{y \in C_k} d(x, y)$$
  3. calcul du coefficient silhouette $s(x) \in [-1, 1]$:
  $$s(x) = \begin{cases} \dfrac{b(x) - a(x)}{\max(a(x), b(x)} & \text{si } |C_i| > 1, \\ 0 \text{ si } |C_i| = 1. \end{cases}$$

Interprétation : le coefficient de silhouette permet de qualifier simplement le partitionnement:

- Des valeurs positives indiquent une séparation élevée entre les clusters.
- Les valeurs négatives indiquent que les classes sont mélangées entre elles (c'est-à-dire qu'il y a chevauchement de classes).
- Lorsque le coefficient de silhouette est nul, c'est une indication que les données sont uniformément réparties dans l'espace euclidien.



Pour évaluer la pertinence de la structure issue du CH, on calculer la distance cophénétique entre deux objets. C'est la hauteur du dendrogramme où les deux branches qui comprennent les deux objets fusionnent en une seule branche.

On peut ainsi calculer le **coefficient de correlation cophénétique** qui mesure la fidélité avec laquelle un dendrogramme préserve les distances par paires entre les points de données originaux non modélisés.

$$c = \frac{\sum_{i<j}(d(x_i,x_j) - \bar{d})(T(x_i,x_j) - \bar{T})}{\sqrt{\sum_{i<j}(d(x_i,x_j) - \bar{d})^2 \sum_{i<j}(T(x_i,x_j) - \bar{T})^2}} \in [-1,1]$$

avec

- $d(x_i,x_j)$ distace entre les données x_i et x_j
- $T(i,j)$ la distance cophénétique entre les données x_i et x_j
- $\bar{d}$ (resp. $\bar{T}$) est la moyenne des distances (resp. distances cophénétiques).

$\Rightarrow$ Très utilisé en biostatistique pour évaluer des modèles de séquences d'ADN.