Introduction à l'Apprentissage Automatique

M1 Informatique

Thomas Pellegrini, équipe SAMoVA, IRIT,
thomas.pellegrini@irit.fr

IRIT - UPS

Présentation de l'UE

Répartition des heures

- 4 cours de 2h
- 5 TD de 2h, 2 groupes
- 6 TP de 2h, 3 groupes

Modalités de Contrôle des Connaissances

- CT : 70%
- CC : 30%, note : un rendu par TP

Syllabus

Objectifs

- Acquérir des bases solides, théoriques et pratiques, en AA
- Devenir autonomes face à un problème qui fait appel à des techniques d'AA

Bibliographie

- Cornuéjols & Miclet, Apprentissage artificiel, concepts et algorithmes, Eyrolles
- Alpaydin, Introduction to Machine Learning, The MIT Press
- Certains slides directement issus du cours de Hugo Larochelle :

  `http://www.dmi.usherb.ca/~larocheh/cours/ift603\_H2015/contenu.html`
- Langage Python pour les TPs, recommandé : lire le tutoriel Python :

  `http://www.dmi.usherb.ca/~larocheh/cours/tutoriel_python.html`

Qu'est-ce que l'apprentissage automatique (*Machine Learning*) ?

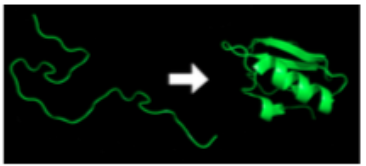
Une définition

Création et analyse de méthodes et modèles qui tirent des informations des données et servent à faire des prédictions

Domaine vaste qui implique des outils et des idées de plusieurs champs

- Algorithmique et Informatique
- Probabilités et Statistiques
- Optimisation
- Algèbre

Quelques exemples



Terminologie

Observations. Exemples ou données utilisés pour l'apprentissage ou pour l'évaluation, ex. des emails

Paramètres (Features) : x. Attributs (numériques) représentant une observation, ex. taille, date, présence de mots clé

Étiquettes / cibles (Labels) : t. Valeurs / catégories associées aux observations, ex. spam / non-spam

Corpus d'apprentissage / de test (Train / Test Data). Observations utilisées pour entraîner et évaluer une méthode d'apprentissage, ex. une collection d'emails avec leurs étiquettes

Remarque : corpus de train et de test **disjoints**

Terminologie

Sujets: données d'entraînement vs. généralisation

- Les algorithmes d'apprentissage procèdent comme suit :
  - on fournit à l'algorithme des **données d'entraînement** ...

'9' '6' '6' '5' '4' '0'

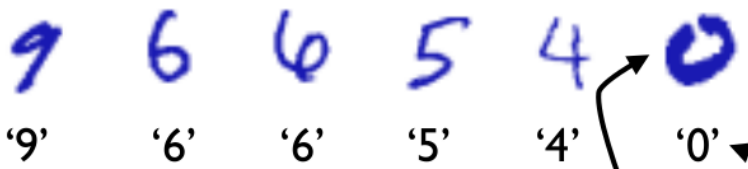
  - ... et l'algorithme retourne un «programme» capable de **généraliser** à de nouvelles données

? ? ? ? ? ?

Terminologie

Sujets: ensemble d'entraînement, entrée, cible

- Les algorithmes d'apprentissage procèdent comme suit :
  - on fournit à l'algorithme des **données d'entraînement** ...



  - on note l'**ensemble d'entraînement**

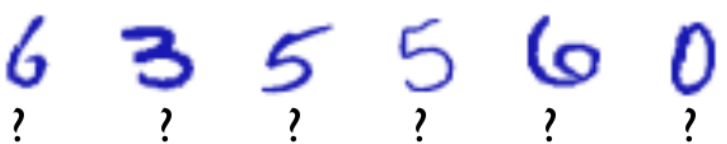
$$\mathcal{D} = \{(\mathbf{x}_1, t_1), \ \ldots \ , \ (\mathbf{x}_N, t_N)\}$$

  - on appelle $\mathbf{x}_n$ une **entrée** et t_n la **cible**

Terminologie

Sujets: modèle

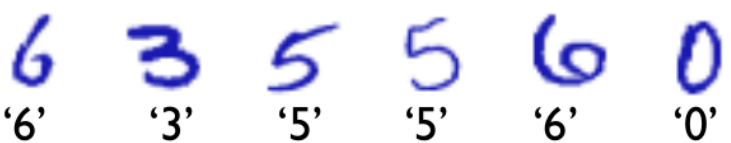
- Les algorithmes d'apprentissage procèdent comme suit :
  - on note le «programme» généré par l'algorithme d'apprentissage $\mathbf{y}(\mathbf{x})$ - on va aussi appeler $\mathbf{y}(\mathbf{x})$ un **modèle**
  - ... et l'algorithme retourne un «programme» capable de **généraliser** à de nouvelles données



Terminologie

Sujets: ensemble de test

- Les algorithmes d'apprentissage procèdent comme suit :
  - on utilise un **ensemble de test** $\mathcal{D}_{\text{test}}$ pour mesurer la performance de généralisation de notre modèle $\mathbf{y}(\mathbf{x})$
  - ... et l'algorithme retourne un «programme» capable de **généraliser** à de nouvelles données



Différents grands paradigmes d'apprentissage

Apprentissage supervisé. Apprentissage à partir d'observations étiquetées (des cibles ou *labels*) : on a la "vérité terrain" (*Groundtruth*)

- Les labels "enseignent" à l'algorithme à établir des correspondances entre les observations et les labels

$$\mathcal{D} = \{(\mathbf{x}_1, t_1), \ \ldots \ , (\mathbf{x}_N, t_N)\}$$

Différents grands paradigmes d'apprentissage (suite)

Apprentissage non-supervisé. Apprentissage à partir d'observations non-étiquetées (pas de *labels*) : on n'a pas de "vérité terrain"

- L'algorithme doit trouver la structure "latente" à partir des features seules
- Peut être l'objectif en soi : trouver des patterns cachés, faire de l'exploration de données
- Peut être un moyen : une étape de pré-traitement des données à une tâche supervisée

$$\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$$

Exemples d'apprentissage supervisé

Classification. Assigner une catégorie à chaque observation

- Les catégories sont discrètes
- La cible est un indice de classe : $t \in \{0, \ldots, K-1\}$
- Exemple : reconnaissance de chiffres manuscrits
  - $\mathbf{x}$: vecteur ou matrice des intensités des pixels de l'image
  - t : identité du chiffre

Régression. Prédire une valeur réelle à chaque observation

- Les catégories sont continues
- La cible est un nombre réel $t \in \mathcal{R}$
- Exemple : prédire le cours d'une action
  - $\mathbf{x}$: vecteur contenant l'information sur l'activité économique récente
  - t : valeur de l'action le lendemain

Exemples d'apprentissage non-supervisé

Partitionnement de données / **Clustering**. Partition d'un ensemble d'observations en régions homogènes, ex. identifier des "communautés" dans les réseaux sociaux

Réduction de dimension. Transformation d'une représentation initiale de features en une représentation plus concise, ex. représenter des images digitales

Pipeline typique d'apprentissage supervisé

collecter données brutes

collecter données brutes

observations initiales de format quelconque

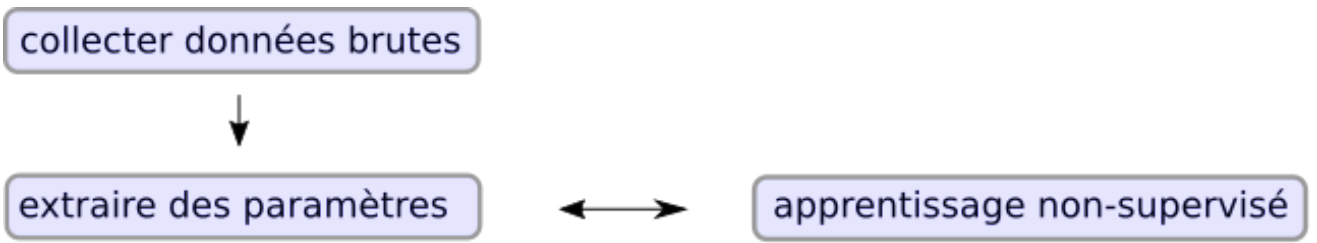
extraire des paramètres

- les features représentent les observations

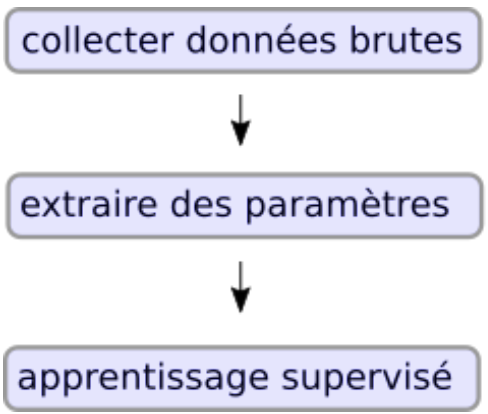
- elles sont des valeurs numériques

- on peut intégrer des connaissances

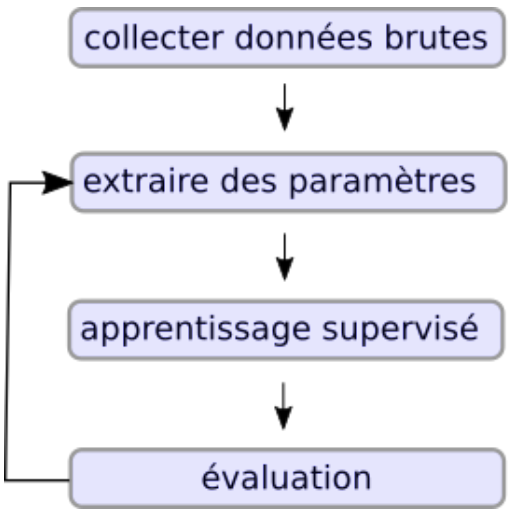
- le succès de la méthode dépend grandement dans le choix de bonnes représentations !



étape de pré-traitement pour du supervisé



entraîner un modèle supervisé avec des donnés étiquetées, ex. classification ou régression



Q : comment déterminer la qualité d'un modèle ?

R : sur un corpus de test avec labels, pas utilisé lors de l'apprentissage

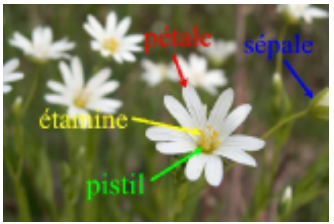
Si pas satisfait des résultats, itérer...

chap 1 : classification **non-supervisée**

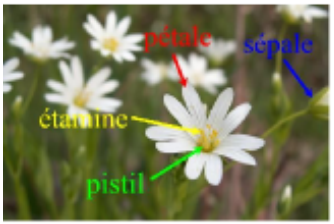
- classification non-hiérarchique : algorithme des k-moyennes, variante à noyaux
- classification hiérarchique : arbre de classification

L'apprentissage dit non-supervisé est un sujet de recherches actuelles en AA... Ici on ne voit qu'une brève introduction qui concerne le partitionnement de données...

Clustering : principe



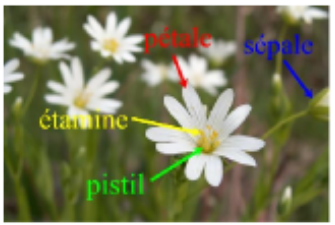
Clustering : principe



	longueur sépale	largeur sépale	longueur pétale	largeur pétale
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2
⋮				
147	6.5	3.0	5.2	2.0
148	6.2	3.4	5.4	2.3
149	5.9	3.0	5.1	1.8

$$X \in \mathbb{R}^4$$

Clustering : principe

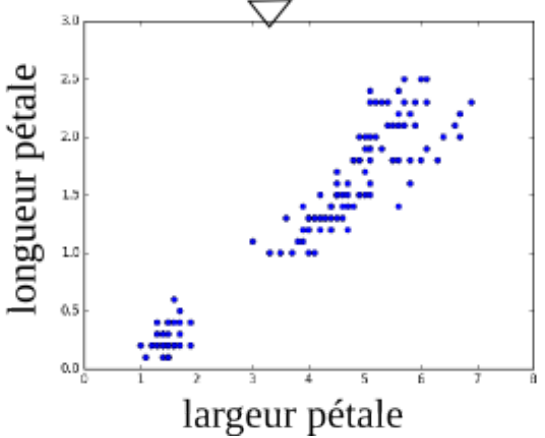


	longueur sépale	largeur sépale	longueur pétale	largeur pétale
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2
⋮				
147	6.5	3.0	5.2	2.0
148	6.2	3.4	5.4	2.3
149	5.9	3.0	5.1	1.8

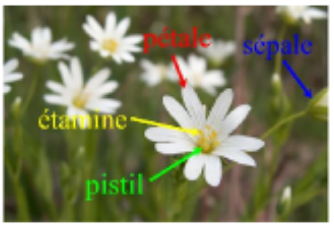
$x \in \mathbb{R}^4$

pour visualiser

$x \in \mathbb{R}^2$



Clustering : principe

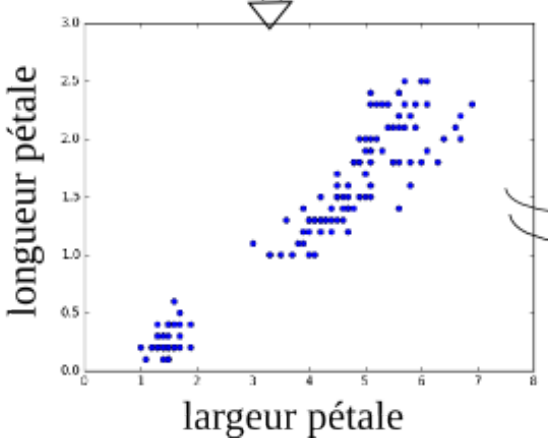


	longueur sépale	largeur sépale	longueur pétale	largeur pétale
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2
⋮				
147	6.5	3.0	5.2	2.0
148	6.2	3.4	5.4	2.3
149	5.9	3.0	5.1	1.8

$x \in \mathbb{R}^4$

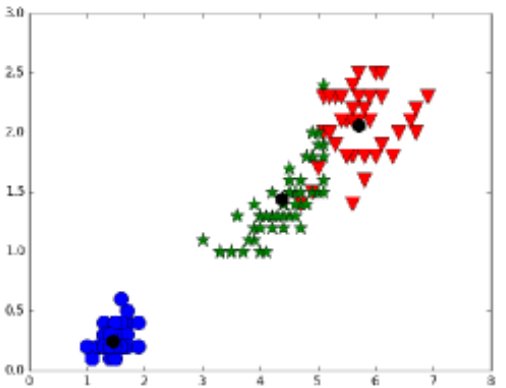
pour visualiser

$x \in \mathbb{R}^2$



k-moyennes

$k = 3$



Algorithme k moyennes

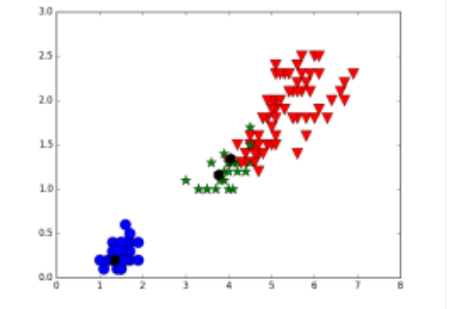
1: Choisir k points qui représentent la position moyenne des partitions $m_1^1, \ldots, m_k^1$ initiales (au hasard par exemple)
2: Répéter jusqu'à convergence :
3: a. assigner chaque observation à la partition la plus proche (i.e. effectuer une partition de Voronoï selon les moyennes) :

$$S_i^{(t)} = \left\{ \mathbf{x}_j : ||\mathbf{x}_j - \mathbf{m}_i^{(t)}|| \leq ||\mathbf{x}_j - \mathbf{m}_{i^*}^{(t)}||, \forall i^* \in [1, k] \right\}$$

4: b. mettre à jour la moyenne de chaque cluster (nuage) :

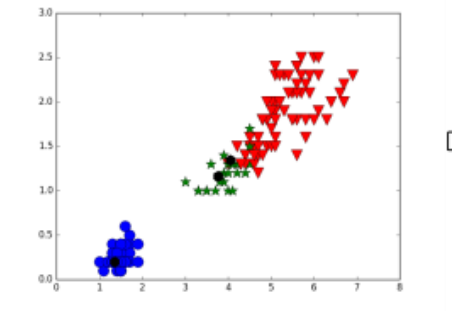
$$\mathbf{m}_i^{(t+1)} = \frac{1}{|S_i^{(t)}|} \sum_{\mathbf{x}_j \in S_i^{(t)}} \mathbf{x}_j$$

1. choix aléatoire de 3 centres
2. attributions des points aux 3 clusters

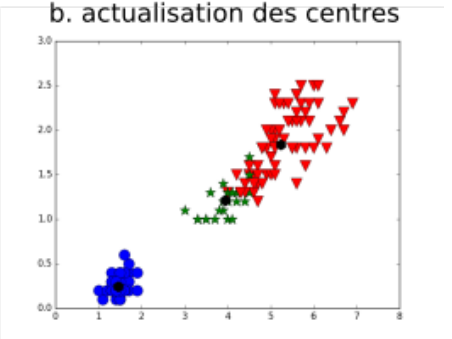


init : choix aléatoire de 3 centres

a. attribution des points aux 3 clusters

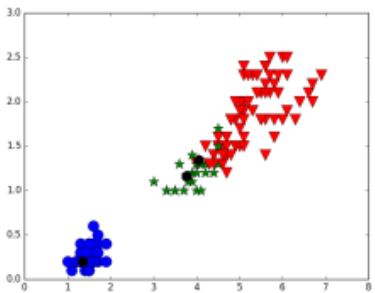


b. actualisation des centres

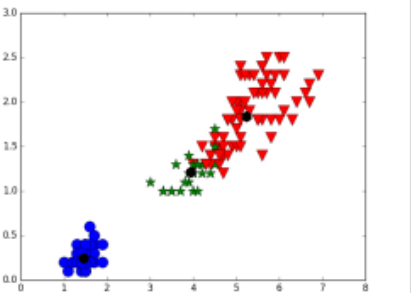


init : choix aléatoire de 3 centres

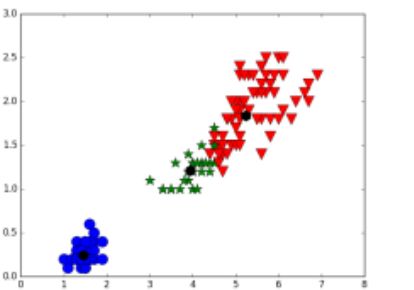
a. attribution des points aux 3 clusters



b. actualisation des centres

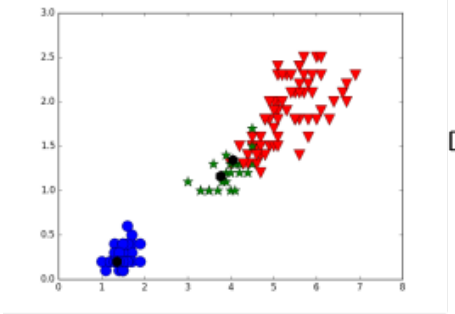


a. attribution des points aux 3 clusters

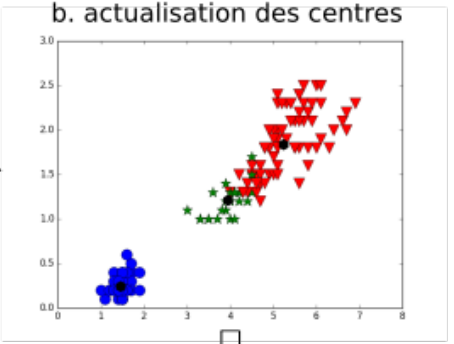


init : choix aléatoire de 3 centres

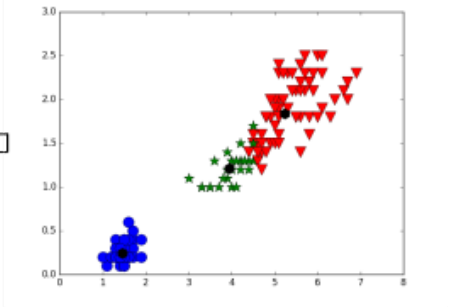
a. attribution des points aux 3 clusters



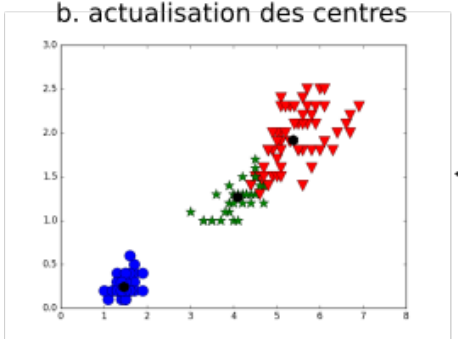
b. actualisation des centres



a. attribution des points aux 3 clusters

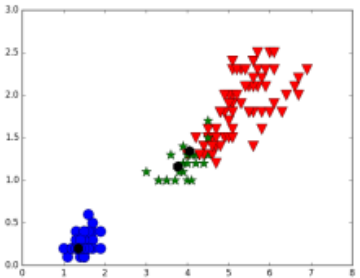


b. actualisation des centres

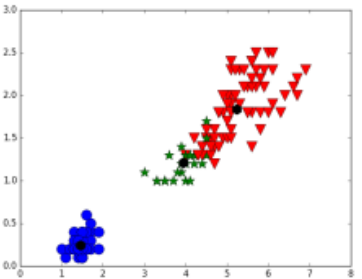


init : choix aléatoire de 3 centres

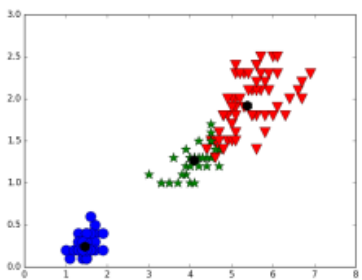
a. attribution des points aux 3 clusters



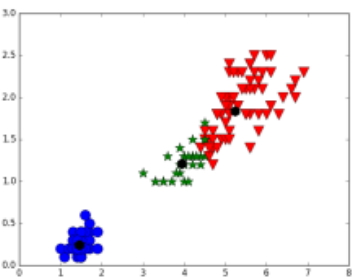
b. actualisation des centres

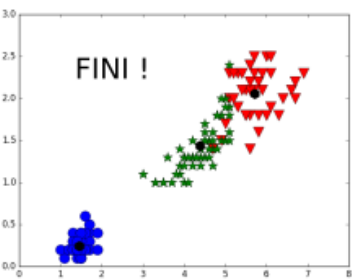


b. actualisation des centres



a. attribution des points aux 3 clusters





Évaluation d'un partitionnement

- Deux métriques : **Sum of Squared Errors (SSE) ou inertie (I)** et pureté

Évaluation d'un partitionnement

- Deux métriques : **Sum of Squared Errors (SSE) ou inertie (I)** et pureté
- $\mathrm{SSE}(\omega) = I(\omega) = \sum_{k=1}^{K} \sum_{x \in \omega_k} ||\mathbf{x} - \mathbf{m}_k^{(t)}||^2$: pas besoin d'une vérité terrain pour la calculer

Évaluation d'un partitionnement

- Deux métriques : **Sum of Squared Errors (SSE) ou inertie (I)** et pureté
- $\mathrm{SSE}(\omega) = I(\omega) = \sum_{k=1}^{K} \sum_{x \in \omega_k} ||\mathbf{x} - \mathbf{m}_k^{(t)}||^2$: pas besoin d'une vérité terrain pour la calculer
- En Python :

```
def computeSSE(data, centers, clusterID):
    sse = 0
    nData = len(data)
    for i in range(nData):
        c = clusterID[i]
        sse += distance(data[i], centers[c])**2
    return sse
```

Évaluation d'un partitionnement

Théorème de Huyghens

Soient $(A_k)_{k=1..K}$ un ensemble de K nuages de points (clusters), de cardinal N_k points chacun, et de centroïde $\mathbf{m}_k$ chacun, les clusters étant disjoints deux à deux. Soit $\mathbf{m}$ le centroïde global de l'union des clusters. Alors :

$$I(\bigcup_k A_k) = \sum_k I(A_k) + \sum_k N_k ||\mathbf{m}_k - \mathbf{m}||^2$$

Interprétation :

```
Inertie = Inerties intra-classes + Inerties
                    inter-classes
```

Évaluation d'un partitionnement

Cas Particulier : 2 clusters A_1 et A_2

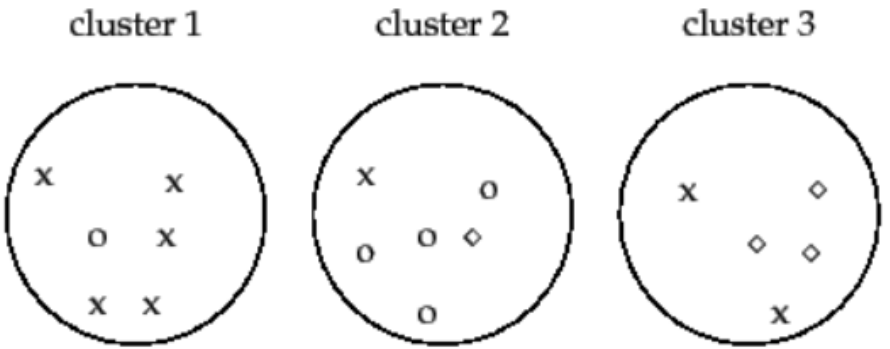
$$I(A_1 \cup A_2) = I(A_1) + I(A_2) + N_1||\mathbf{m}_1 - \mathbf{m}||^2 + N_2||\mathbf{m}_2 - \mathbf{m}||^2$$

$$I(A_1 \cup A_2) = I(A_1) + I(A_2) + \frac{N_1 N_2}{N_1 + N_2}||\mathbf{m}_1 - \mathbf{m}_2||^2$$

Le terme le plus à droite peut être utilisé comme une pseudo-distance qui quantifie l'inertie inter-classe

Évaluation d'un partitionnement

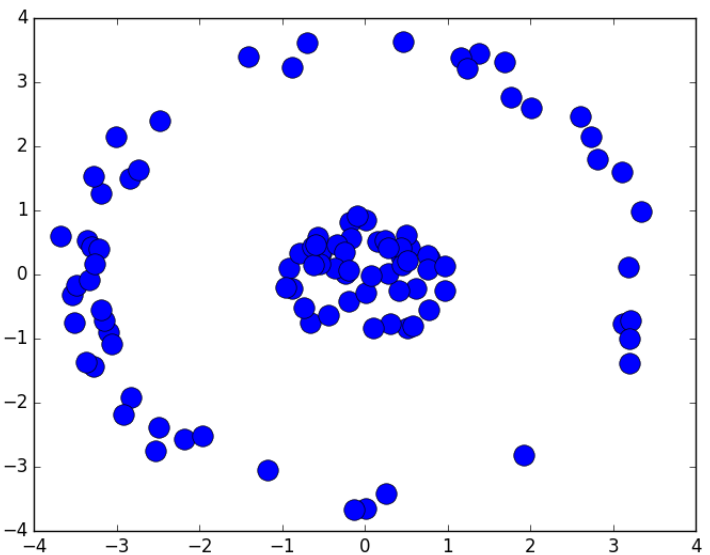
- $\text{Purete}(\omega, c) = \frac{1}{N} \sum_{k=1}^{K} \max_j |\omega_k \cap c_j|$
- Exemple :



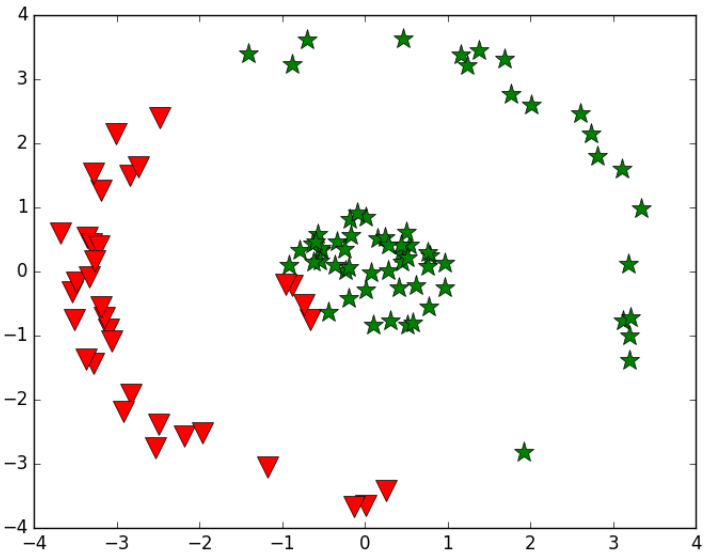
- Pureté = (5+4+3)/17 = 0,706

Limites des k moyennes

- Les k moyennes ne peuvent détecter que des clusters linéairement séparables
- Que donne l'algorithme sur les données suivantes ?



Limites des k moyennes



- Pureté = 0.75
- SSE = 359.80
- Satisfaisant ?

k-moyennes à noyaux / *Kernel k-means*

- 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

k-moyennes à noyaux / *Kernel k-means*

- 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 $N \times N$

k-moyennes à noyaux / *Kernel k-means*

- 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 :

k-moyennes à noyaux / *Kernel k-means*

- 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 : $K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i \mathbf{x}_j + 1)^h$

k-moyennes à noyaux / *Kernel k-means*

- 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 : $K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i \mathbf{x}_j + 1)^h$
  - Noyau gaussien radial (RBF) : $K(\mathbf{x}_i, \mathbf{x}_j) = e^{-||\mathbf{x}_i - \mathbf{x}_j||^2 / 2\sigma^2}$

Exemple : noyau gaussien radial

données originales :
100 points en 2 dim

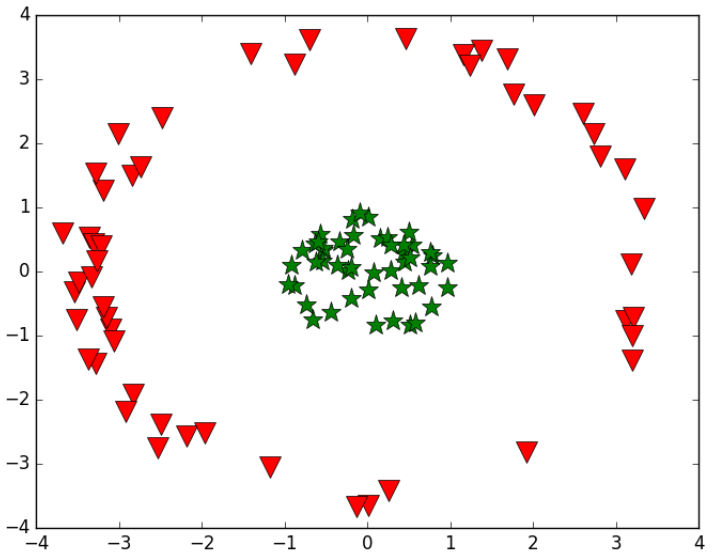
X_1	X_2
-0.524	0.339
0.152	0.515
...	
2.078	2.865
3.149	-1.811

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 ?

k-moyennes : hyperparamètres

Outre les données, ce qui est fourni / choisi en entrée à l'algorithme est :

- Le nombre de clusters recherché k,
- Le type de distance utilisée pour mesurer l'éloignement des points entre eux,
- Le déplacement minimum des centroïdes attendu en-dessous duquel l'algorithme est arrêté (critère d'arrêt),
- Le nombre maximal éventuel d'itérations de l'algorithme au-delà duquel on arrête,
- Autres ?

Ces informations, choisies *a piori* par le concepteur / utilisateur de l'algorithme, sont appelées des **"hyper-paramètres"**.

Distances et mesures de similarité

- Distance euclidienne (norme L_2) :
  $d_2(M_i, M_j) = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}$

Distances et mesures de similarité

- Distance euclidienne (norme L_2) :
  $d_2(M_i, M_j) = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}$
- Distance de Manhattan ou taxi-distance (norme L_1) :
  $d_1(M_i, M_j) = |x_j - x_i| + |y_j - y_i|$

Distances et mesures de similarité

- Distance euclidienne (norme L_2) :
  $d_2(M_i, M_j) = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}$
- Distance de Manhattan ou taxi-distance (norme L_1) :
  $d_1(M_i, M_j) = |x_j - x_i| + |y_j - y_i|$
- Distance uniforme (norme L_∞) :
  $d_\infty(M_i, M_j) = \max(|x_j - x_i|, |y_j - y_i|)$

Distances et mesures de similarité

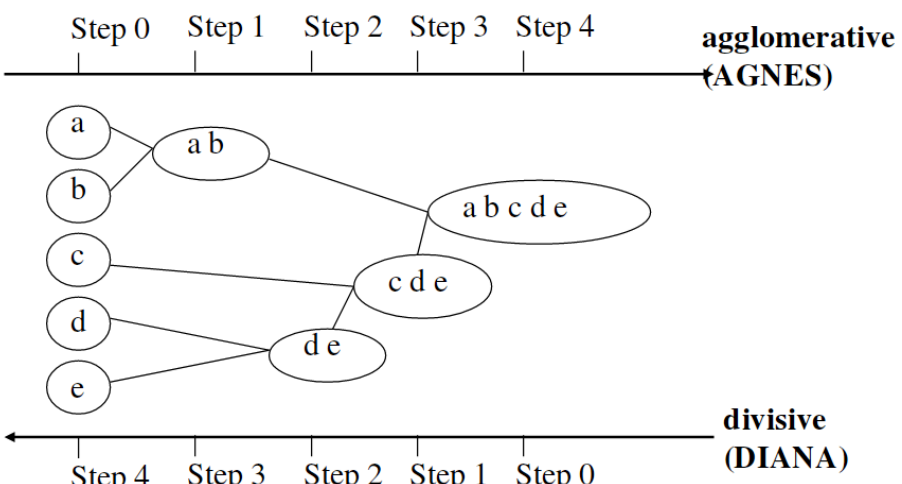
- Distance euclidienne (norme L_2) :
  $d_2(M_i, M_j) = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}$
- Distance de Manhattan ou taxi-distance (norme L_1) :
  $d_1(M_i, M_j) = |x_j - x_i| + |y_j - y_i|$
- Distance uniforme (norme L_∞) :
  $d_\infty(M_i, M_j) = \max(|x_j - x_i|, |y_j - y_i|)$
- Distance de Mahalanobis :
  $d_m(M_i, M_j) = \sqrt{(\mathbf{M_j} - \mathbf{M_i})^t \Sigma^{-1} (\mathbf{M_j} - \mathbf{M_i})}$, où Σ est la matrice de covariance du nuage de points

Distances et mesures de similarité

- Distance euclidienne (norme L_2) :
  $d_2(M_i, M_j) = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}$
- Distance de Manhattan ou taxi-distance (norme L_1) :
  $d_1(M_i, M_j) = |x_j - x_i| + |y_j - y_i|$
- Distance uniforme (norme L_∞) :
  $d_\infty(M_i, M_j) = \max(|x_j - x_i|, |y_j - y_i|)$
- Distance de Mahalanobis :
  $d_m(M_i, M_j) = \sqrt{(\mathbf{M_j} - \mathbf{M_i})^t \Sigma^{-1} (\mathbf{M_j} - \mathbf{M_i})}$, où Σ est la matrice de covariance du nuage de points
- et bien d'autres...

Classification hiérarchique / Hierarchical Clustering

- Une alternative au k-moyennes lorsqu'on a aucune idée sur le nombre de clusters *a priori*
- On construit un arbre de classification appelé dendrogramme ou dendogramme = une hiérarchie de clusters
- De manière ascendante ou descendante
- Les clusters sont obtenus en "coupant" l'arbre au niveau désiré



Exemple : classification d'évènements sonores

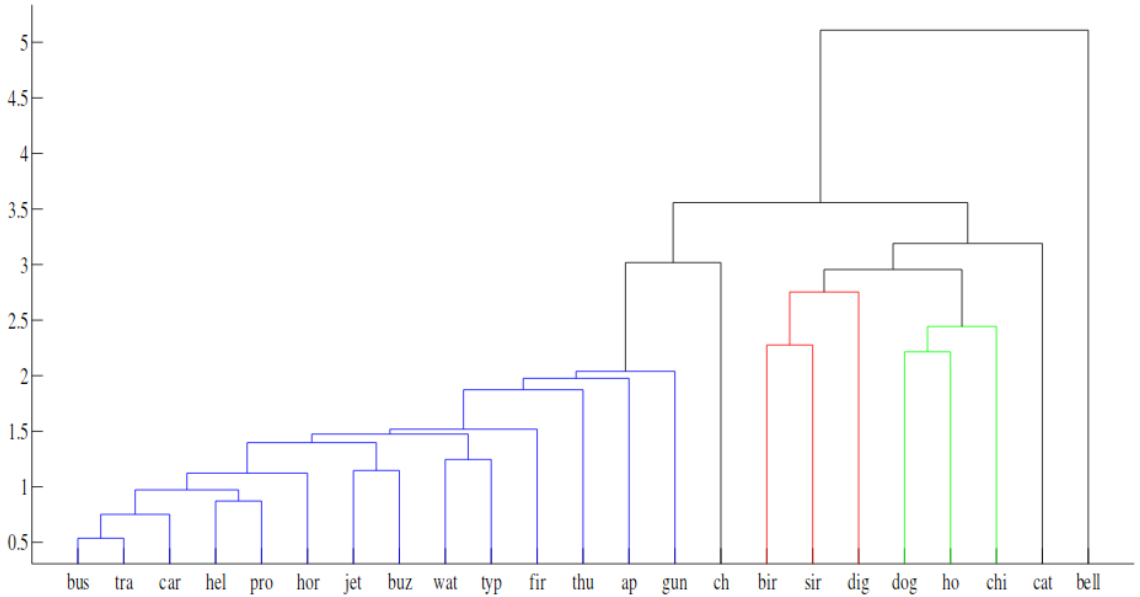
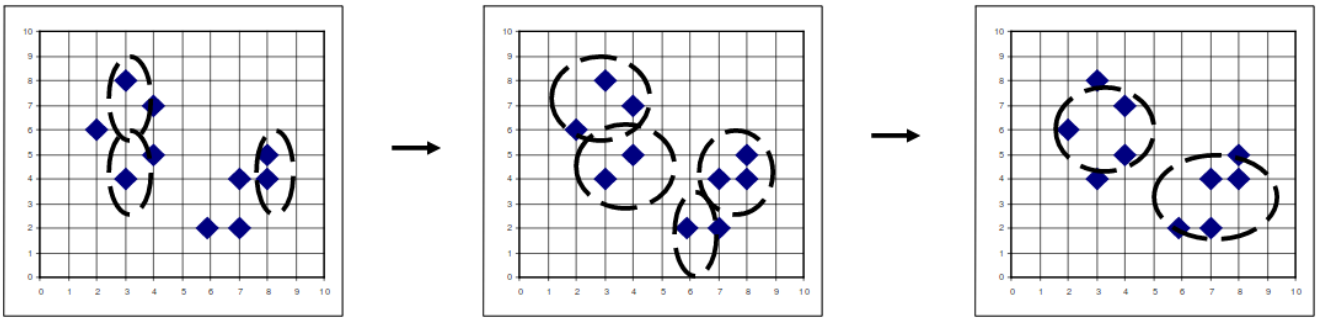


Figure 6: Dendrogram for the 23 concepts, achieved with the modified Euclidean distance. Y-axis represents cluster distances. Al
viations: tra *traffic*, pro *propeller airplane*, hel *helicopter*, hor *horse*, fir *fire*, thu *thunder*, ap *applause*, wat *water*, typ *typing*, bir
chi *chicken*, ho *horn*, ch *insect chirp*, dig *digital telephone*, sir *siren*.

Pellegrini, Thomas, et al. "Hierarchical clustering experiments for application to audio event detection." Proceedings of the 13th International Conference on Speech and Computer. 2009.

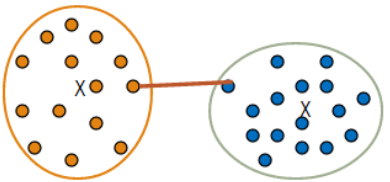
Variante agglomérative

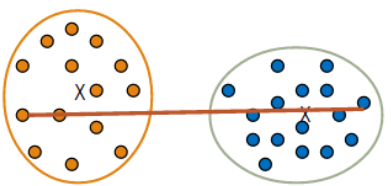


- Initialisation : pré-calcul de la matrice contenant les distances de tous les points entre eux
- On itère :
  - Sélection des deux points ou clusters les plus ”proches”
  - Actualisation de la matrice de distances

Variante agglomérative

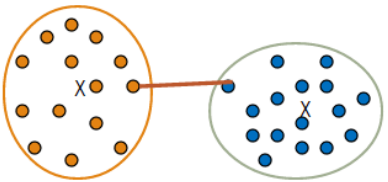
- *Single link* : distance **minimale** entre les points des deux clusters à regrouper

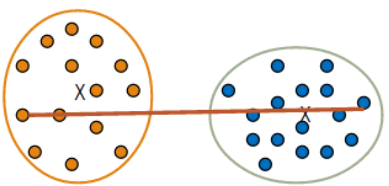




Variante agglomérative

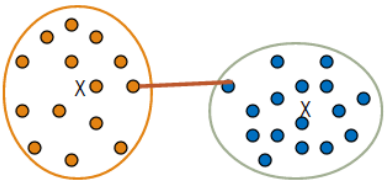
- *Single link* : distance **minimale** entre les points des deux clusters à regrouper
  - ”Plus proches voisins”

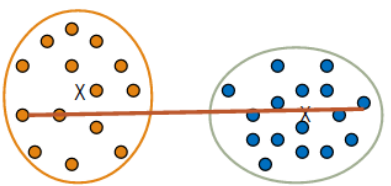




Variante agglomérative

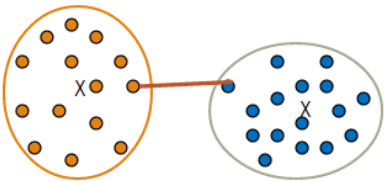
- *Single link* : distance **minimale** entre les points des deux clusters à regrouper
  - ”Plus proches voisins”
  - Similarités locales privilégiées : structure globale des clusters ignorée

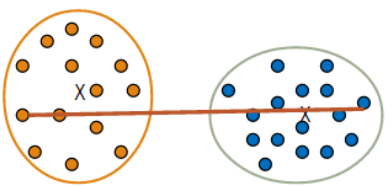




Variante agglomérative

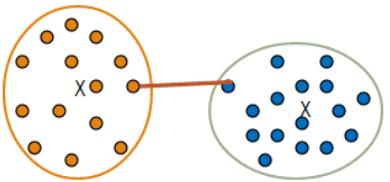
- *Single link* : distance **minimale** entre les points des deux clusters à regrouper
  - ”Plus proches voisins”
  - Similarités locales privilégiées : structure globale des clusters ignorée
  - Sensible au bruit et aux outliers



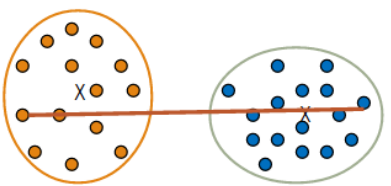


Variante agglomérative

- *Single link* : distance **minimale** entre les points des deux clusters à regrouper
  - ”Plus proches voisins”
  - Similarités locales privilégiées : structure globale des clusters ignorée
  - Sensible au bruit et aux outliers

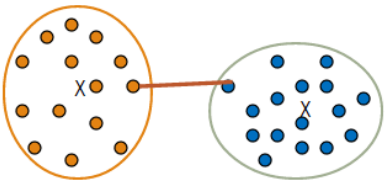


- *Complete link* : distance **maximale** entre les points des deux clusters à regrouper

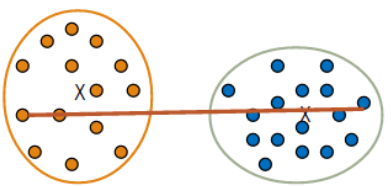


Variante agglomérative

- *Single link* : distance **minimale** entre les points des deux clusters à regrouper
  - ”Plus proches voisins”
  - Similarités locales privilégiées : structure globale des clusters ignorée
  - Sensible au bruit et aux outliers

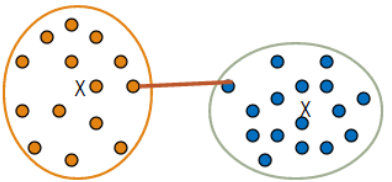


- *Complete link* : distance **maximale** entre les points des deux clusters à regrouper
  - Similarité entre les points les plus dissimilaires de deux clusters

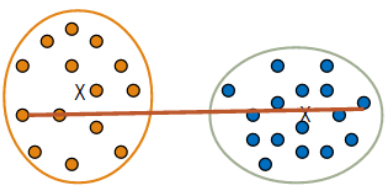


Variante agglomérative

- *Single link* : distance **minimale** entre les points des deux clusters à regrouper
  - ”Plus proches voisins”
  - Similarités locales privilégiées : structure globale des clusters ignorée
  - Sensible au bruit et aux outliers

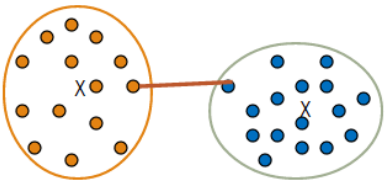


- *Complete link* : distance **maximale** entre les points des deux clusters à regrouper
  - Similarité entre les points les plus dissimilaires de deux clusters
  - Forme le cluster avec le plus petit ”diamètre” à chaque itération, clusters assez compacts

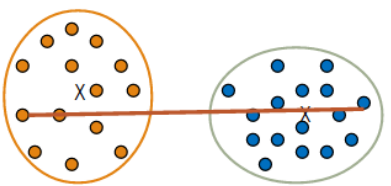


Variante agglomérative

- *Single link* : distance **minimale** entre les points des deux clusters à regrouper
  - ”Plus proches voisins”
  - Similarités locales privilégiées : structure globale des clusters ignorée
  - Sensible au bruit et aux outliers

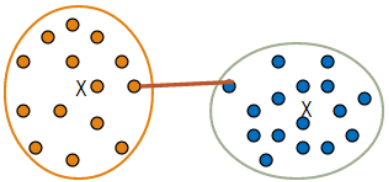


- *Complete link* : distance **maximale** entre les points des deux clusters à regrouper
  - Similarité entre les points les plus dissimilaires de deux clusters
  - Forme le cluster avec le plus petit ”diamètre” à chaque itération, clusters assez compacts
  - Comportement ”non-local”

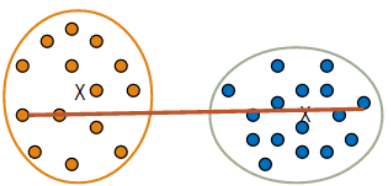


Variante agglomérative

- *Single link* : distance **minimale** entre les points des deux clusters à regrouper
  - ”Plus proches voisins”
  - Similarités locales privilégiées : structure globale des clusters ignorée
  - Sensible au bruit et aux outliers

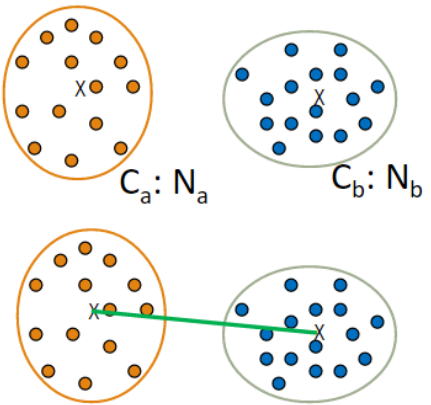


- *Complete link* : distance **maximale** entre les points des deux clusters à regrouper
  - Similarité entre les points les plus dissimilaires de deux clusters
  - Forme le cluster avec le plus petit ”diamètre” à chaque itération, clusters assez compacts
  - Comportement ”non-local”
  - Plus robuste que single link *a priori* mais sensible aux outliers



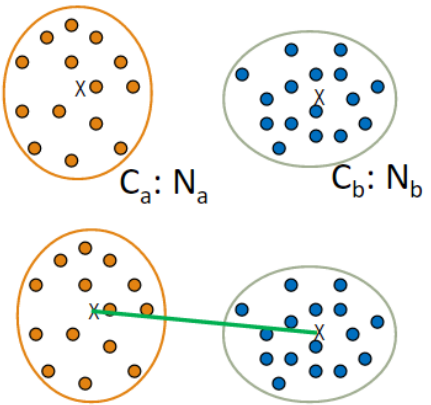
Variante agglomérative

- *Average link* : distance **moyenne** entre toutes les paires de points des deux clusters à regrouper



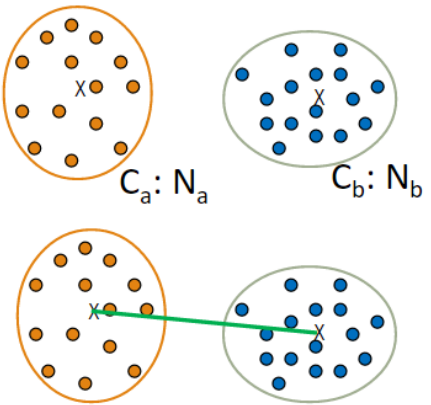
Variante agglomérative

- *Average link* : distance **moyenne** entre toutes les paires de points des deux clusters à regrouper
  - Cher en temps de calcul



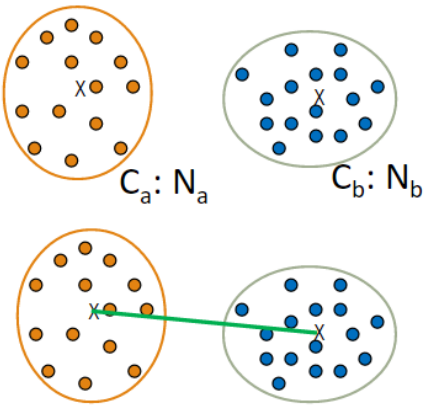
Variante agglomérative

- *Average link* : distance **moyenne** entre toutes les paires de points des deux clusters à regrouper
  - Cher en temps de calcul
  - Moins sensible au bruit et aux outliers



Variante agglomérative

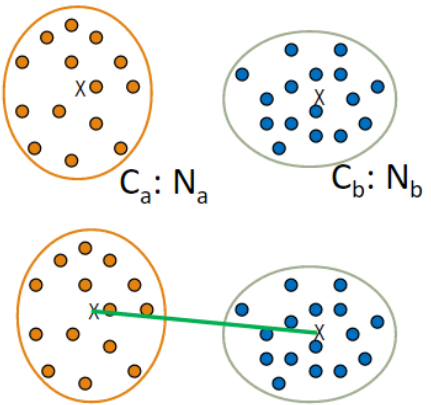
- *Average link* : distance **moyenne** entre toutes les paires de points des deux clusters à regrouper
  - Cher en temps de calcul
  - Moins sensible au bruit et aux outliers
- *Centroid link* : distance **maximale** entre les centroïdes des deux clusters à regrouper



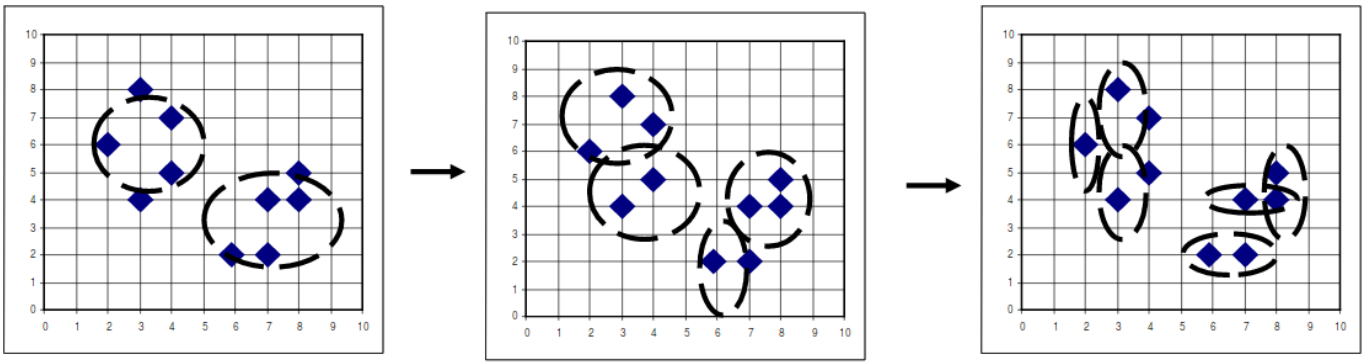
Variante agglomérative

- *Average link* : distance **moyenne** entre toutes les paires de points des deux clusters à regrouper
  - Cher en temps de calcul
  - Moins sensible au bruit et aux outliers
- *Centroid link* : distance **maximale** entre les centroïdes des deux clusters à regrouper
- Critère de Ward : choix des deux clusters dont le regroupement minimise l'augmentation de l'erreur quadratique :

$$\Delta \mathrm{SSE} = \frac{N_a N_b}{N_a + N_b} d(c_a, c_b)^2$$



Variante descendante ou divisive ou top-down

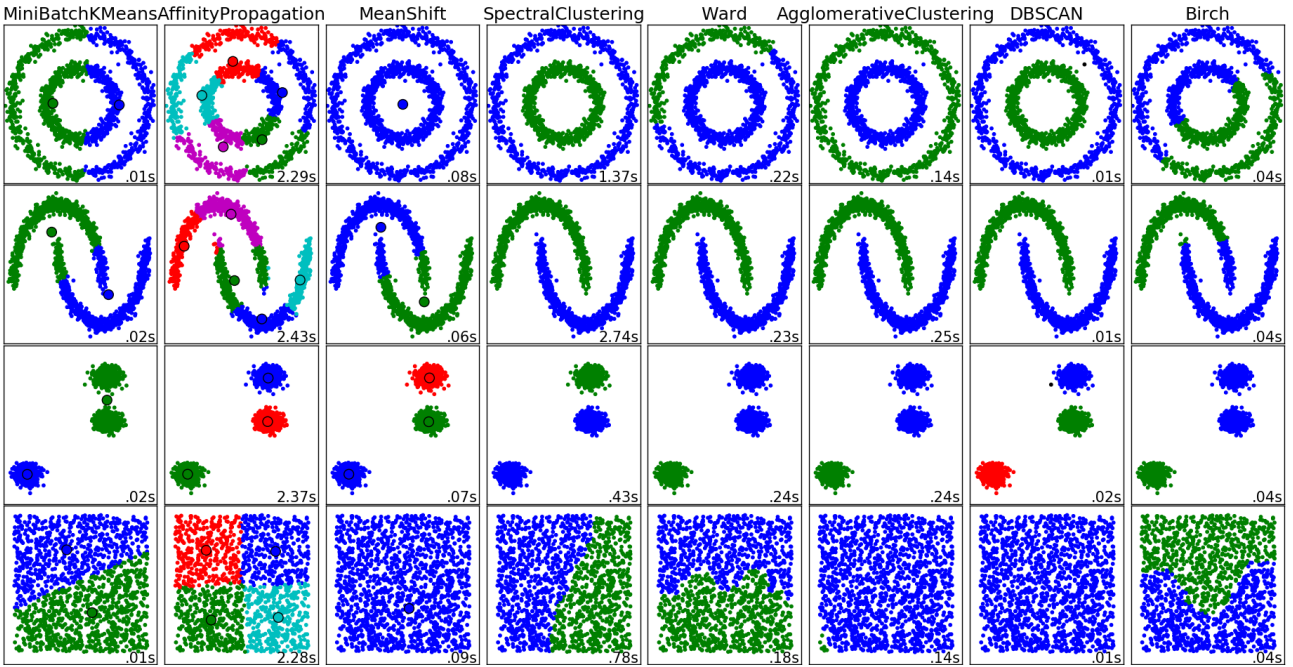


- Initialisation : un seul cluster contenant tous les points
- On itère :
  - Sélection des deux points ou clusters dont la division maximise la différence de SSE (critère de Ward)
  - Actualisation de la valeur de la SSE

Limites de la classification hiérarchique

- 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.

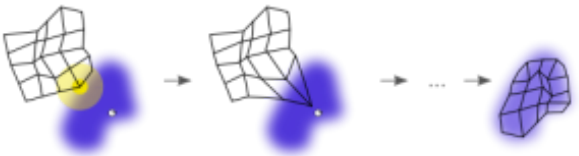
Beaucoup d'autres techniques de clustering



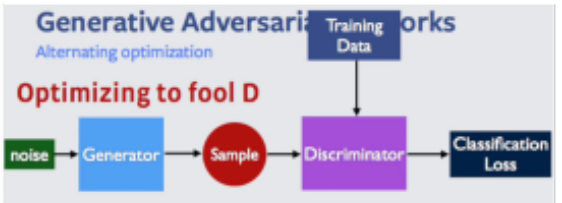
http://scikit-learn.org/stable/modules/clustering.html

Le non-supervisé : un monde...

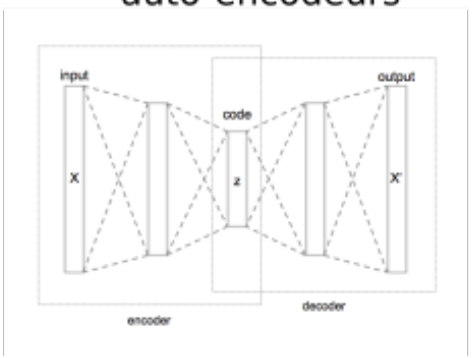
cartes auto-organisatrices



réseaux adversaires générateurs



auto-encodeurs



machines de Boltzmann