# Ingénierie de la fouille et de la visualisation de données massives (RCP216)

## Classification automatique

Michel Crucianu
(prenom.nom@cnam.fr)
http://cedric.cnam.fr/vertigo/Cours/RCP216/

Département Informatique
Conservatoire National des Arts & Métiers, Paris, France

22 septembre 2025

# Plan du cours

# Objectifs et utilisations de la classification automatique

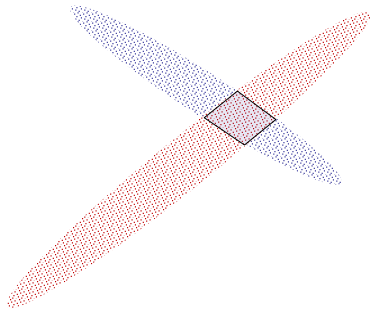
(*cluster analysis, clustering*)

- Objectif général : répartir un ensemble donné de $N$ observations en groupes (catégories, classes, taxons, clusters) de façon à regrouper les observations similaires et à séparer les observations dissimilaires
  - Partitionnement des données, ou
  - Hiérarchie de groupes ($\rightarrow$ plusieurs partitionnements disponibles)
- Conditions :
  - Aucune information n'est disponible concernant l'appartenance de certaines données à certaines « classes »
  - Le nombre de groupes recherchés peut être connu *a priori* ou non
- Utilisations :
  - Mettre en évidence une structure (simple) dans un ensemble de données
  - Résumer un grand ensemble de données par les représentants des groupes

# Typologie des méthodes de classification automatique

Choix méthode $\Leftarrow$ connaissance des données et de la nature des groupes recherchés

- Nature des données : numériques, catégorielles, mixtes
- Représentation des données :
  - Représentation vectorielle $\rightarrow$ définir centres de gravité, densités, intervalles, différentes distances $\Rightarrow$ complexité en général $O(N)$
  - Simple : seules sont disponibles les distances entre observations $\Rightarrow$ complexité $\geq O(N^2)$
- Groupes mutuellement exclusifs ou non ?
  - A quel groupe appartiennent les données entourées ?



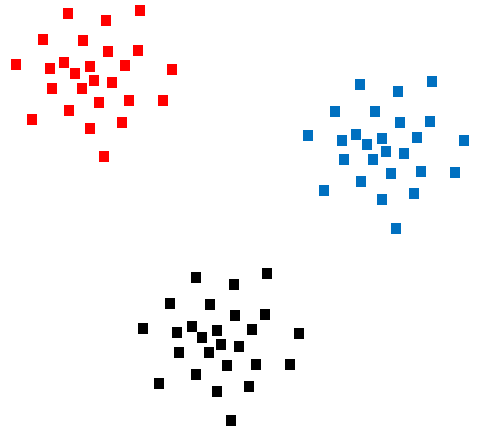
- Nature des groupes :
  - Nets : une observation appartient ou n'appartient pas à un groupe
  - Flous : une observation peut appartenir à différents degrés à plusieurs groupes $\Rightarrow$ convergence souvent plus robuste de l'algorithme de classification
    - Groupes flous $\rightarrow$ nets : chaque observation affectée au groupe auquel elle appartient le plus

# Typologie des méthodes de classification automatique (2)

- Critère de regroupement (définition des groupes) : en général n'est pas explicite !

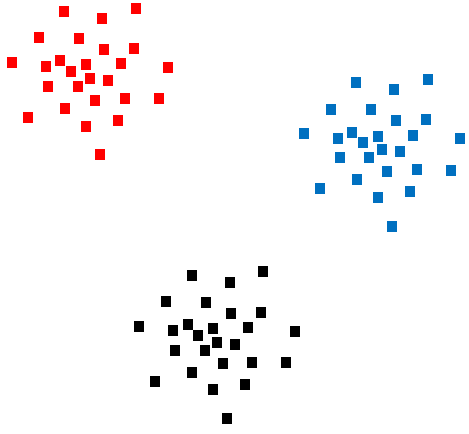
# Typologie des méthodes de classification automatique (2)

- Critère de regroupement (définition des groupes) : en général n'est pas explicite !
  - Ensembles compacts éloignés entre eux :

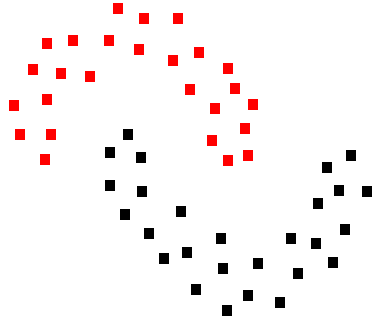


# Typologie des méthodes de classification automatique (2)

- Critère de regroupement (définition des groupes) : en général n'est pas explicite !
  - Ensembles compacts éloignés entre eux :

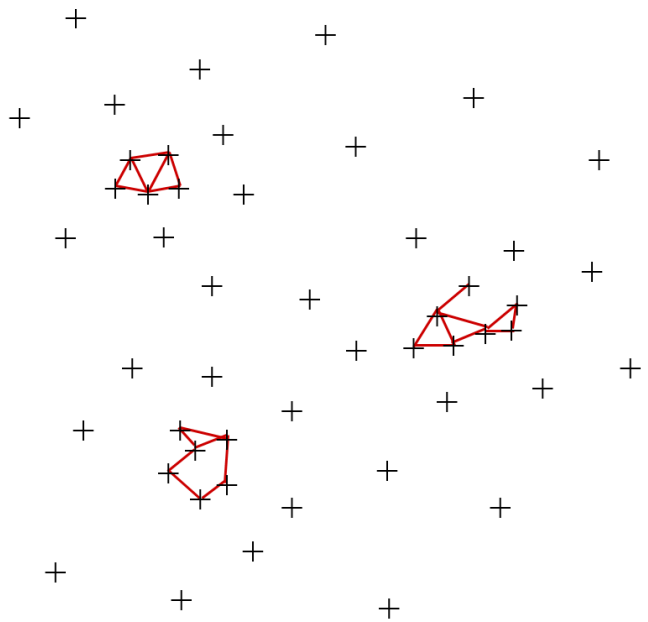


  - Ensembles denses séparés par des régions moins denses :

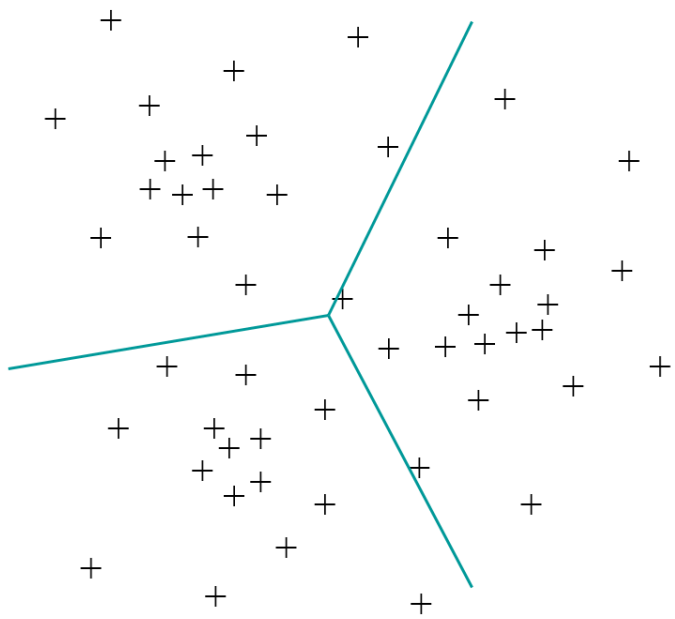


# Classification automatique *vs* auto-jointure par similarité

- Auto-jointure par similarité
  - Données à distance $< \theta$
  - Extraction ultérieure de cliques, graphes connexes
  - Autres données : ignorées



- Classification automatique
  - Regroupement des données par similarité
  - Chaque donnée appartient à une partition



# Centres mobiles : la méthode

- Ensemble $\mathcal{E}$ de $N$ données décrites par $p$ variables à valeurs dans $\mathbb{R}$
- Objectif : répartir les $N$ données en $k$ groupes disjoints $\mathcal{E}_1, \ldots, \mathcal{E}_k$ (inconnus *a priori*) en optimisant la somme des inerties intra-classe

$$\phi_{\mathcal{E}}(\mathcal{C}) = \sum_{j=1}^{k} \sum_{\mathbf{x}_i \in \mathcal{E}_j} d^2(\mathbf{x}_i, \mathbf{m}_j) \tag{1}$$

avec $\mathcal{C} = \{\mathbf{m}_j, 1 \leq j \leq k\}$ l'ensemble des centres des $k$ groupes, $d$ la distance dans $\mathbb{R}^p$ qui définit la nature des dissimilarités

# Centres mobiles : l'algorithme

**Data :** Ensemble $\mathcal{E}$ de $N$ données de $\mathbb{R}^p$

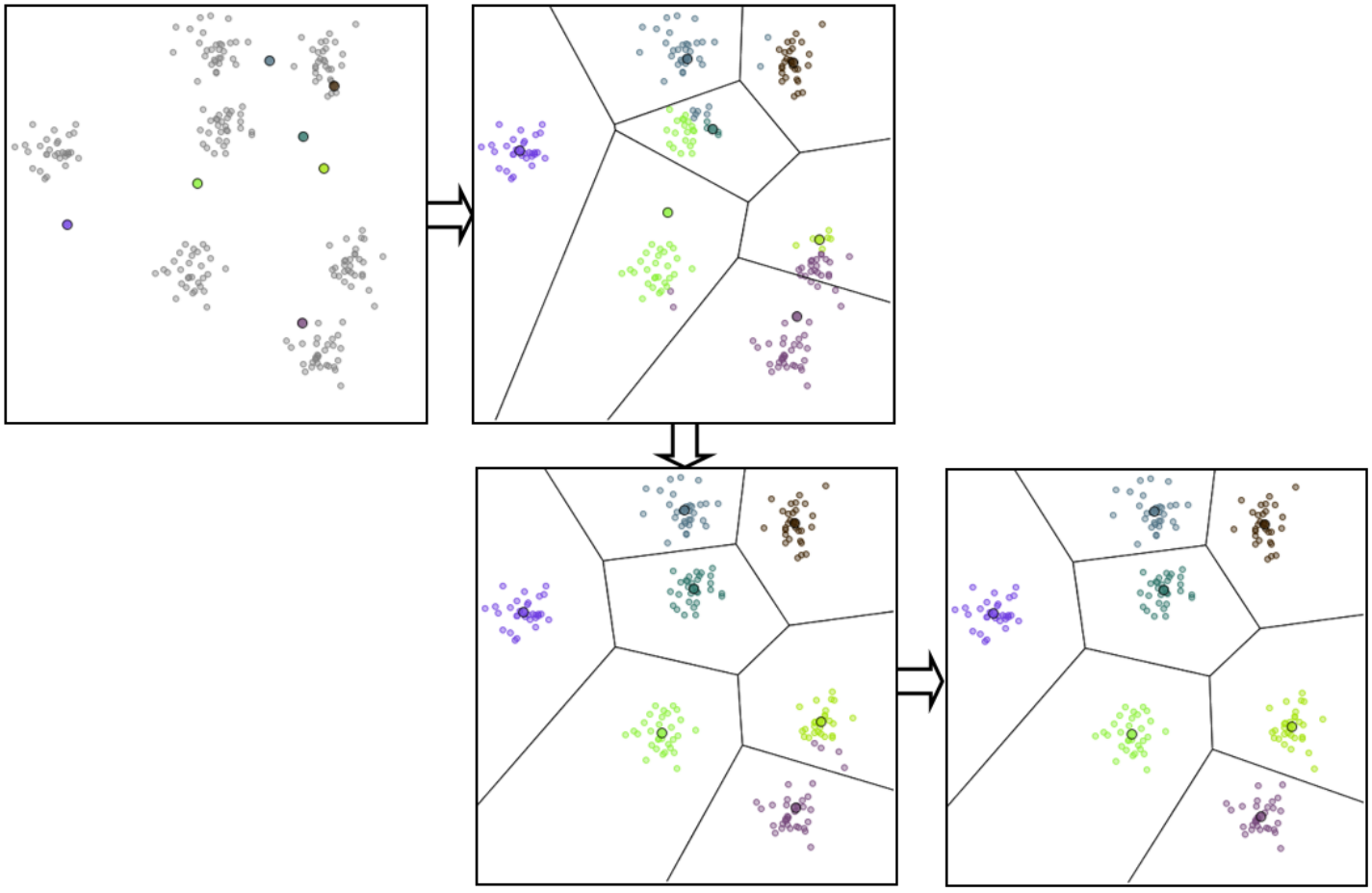
**Result :** $k$ groupes (*clusters*) disjoints $\mathcal{E}_1, \ldots, \mathcal{E}_k$ et ensemble $\mathcal{C}$ de leurs centres

1 Initialization aléatoire des centres $\mathbf{m}_j$, $1 \leq j \leq k$ ;
2 **while** *centres non stabilisés* **do**
3 | Affectation de chaque donnée au groupe du centre le plus proche ;
4 | Remplacement des anciens centres par les centres de gravité des groupes ;
5 **end**

- $\phi_{\mathcal{E}}(\mathcal{C})$ diminue lors de chacune des deux étapes du processus itératif ; comme $\phi_{\mathcal{E}}(\mathcal{C}) \geq 0$, le processus itératif doit converger
- mais la solution obtenue sera un minimum local, dépendant de l'initialisation, souvent beaucoup moins bon que le minimum global

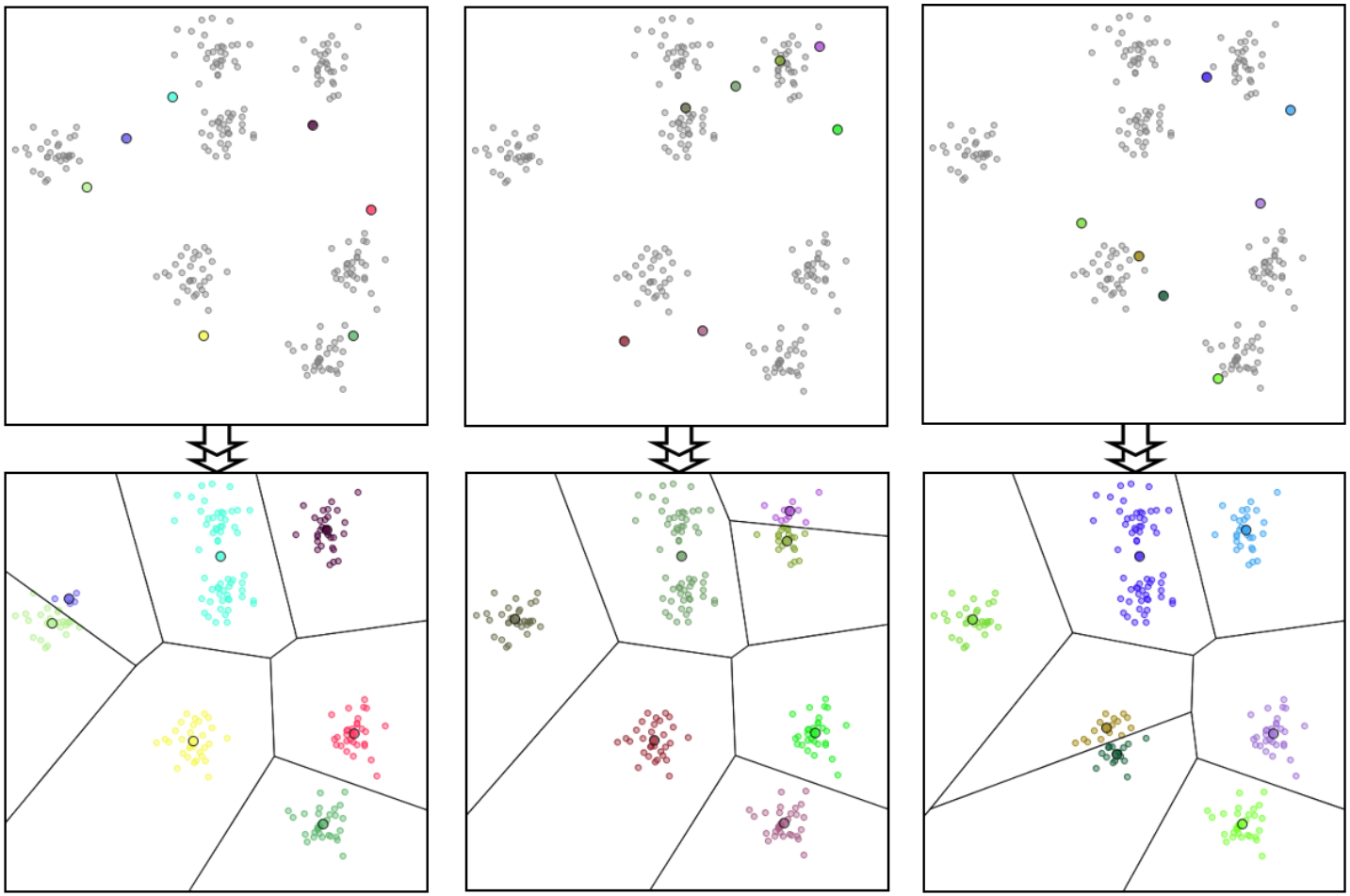
# Centres mobiles : illustration

(données issues de 7 lois normales bidimensionnelles, classification avec 7 centres)



# Centres mobiles : illustration (2)

(résultats avec 3 initialisations différentes)



→ Faire tourner l'algorithme plusieurs fois, à partir d'initialisations aléatoires différentes, ne garantit pas d'arriver à une bonne solution !

# Centres mobiles : convergence

$\phi_{\mathcal{E}}(\mathcal{C})$ diminue de façon monotone non stricte à chaque étape de chaque itération :

1. Affectation de chaque donnée au groupe du centre le plus proche : $\mathbf{x}_i$ passe du groupe de centre $\mathbf{m}_p$ au groupe de centre $\mathbf{m}_q$ si $d^2(\mathbf{x}_i, \mathbf{m}_p) > d^2(\mathbf{x}_i, \mathbf{m}_q)$, donc

$$d^2(\mathbf{x}_i, \mathbf{m}_p) + \sum_{l \neq i} d^2(\mathbf{x}_l, \mathbf{m}_{C(l)}) > d^2(\mathbf{x}_i, \mathbf{m}_q) + \sum_{l \neq i} d^2(\mathbf{x}_l, \mathbf{m}_{C(l)})$$

2. Remplacement des anciens centres par les centres de gravité des groupes : si $\widetilde{\mathbf{m}}_j$ est l'ancien centre du groupe $j$ et $\mathbf{m}_j$ le nouveau, alors

$$\begin{aligned}
\sum_{\mathbf{x}_i \in \mathcal{E}_j} d^2(\mathbf{x}_i, \widetilde{\mathbf{m}}_j) &= \sum_{\mathbf{x}_i \in \mathcal{E}_j} \left\| \mathbf{x}_i - \mathbf{m}_j + \mathbf{m}_j - \widetilde{\mathbf{m}}_j \right\|^2 \\
&= \sum_{\mathbf{x}_i \in \mathcal{E}_j} \left\| \mathbf{x}_i - \mathbf{m}_j \right\|^2 + \sum_{\mathbf{x}_i \in \mathcal{E}_j} \left\| \mathbf{m}_j - \widetilde{\mathbf{m}}_j \right\|^2 + 2 \left( \mathbf{m}_j - \widetilde{\mathbf{m}}_j \right)^T \underbrace{\sum_{\mathbf{x}_i \in \mathcal{E}_j} (\mathbf{x}_i - \mathbf{m}_j)}_{=0} \\
&\geq \sum_{\mathbf{x}_i \in \mathcal{E}_j} \left\| \mathbf{x}_i - \mathbf{m}_j \right\|^2 \left( = \sum_{\mathbf{x}_i \in \mathcal{E}_j} d^2(\mathbf{x}_i, \mathbf{m}_j) \right)
\end{aligned}$$

# *K-means* : l'algorithme *online* de [4]

- *K-means* de [4] est une variante *online* (non *batch*) de la méthode des centres mobiles ; souvent, *K-means* est utilisé comme synonyme des centres mobiles

  **Data :** Ensemble $\mathcal{E}$ de $N$ données de $\mathbb{R}^p$

  **Result :** $k$ groupes (*clusters*) disjoints $\mathcal{E}_1, \ldots, \mathcal{E}_k$ et ensemble $\mathcal{C}$ de leurs centres

1 Initialisation aléatoire des centres $\mathbf{m}_j$, $1 \leq j \leq k$ ;
2 Chaque groupe est initialisé avec son centre comme seul membre du groupe ;
3 **while** *centres non stabilisés* **do**
4 | Choix aléatoire d'une des données ;
5 | Affectation de la donnée au groupe du centre le plus proche ;
6 | Recalcul des centres pour le groupe que la donnée vient de rejoindre et celui qu'elle vient de quitter ;
7 **end**

- Recalcul du centre $j$ rejoint par la donnée $i$ : $\mathbf{m}_j = \frac{1}{n_j}(\widetilde{n}_j \widetilde{\mathbf{m}}_j + \mathbf{x}_i)$, avec $n_j = \widetilde{n}_j + 1$
- Recalcul du centre $l$ quitté par la donnée $i$ : $\mathbf{m}_l = \frac{1}{n_l}(\widetilde{n}_l \widetilde{\mathbf{m}}_l - \mathbf{x}_i)$, avec $n_l = \widetilde{n}_l - 1$
- Entre *batch* et *online* : à chaque itération, échantillon de $b$ données $\rightarrow$ *mini-batch*

# *K-means* : une implémentation simple MapReduce

**Data :** Ensemble $\mathcal{E}$ de $N$ données de $\mathbb{R}^p$

**Result :** $k$ groupes (*clusters*) disjoints $\mathcal{E}_1, \ldots, \mathcal{E}_k$ et ensemble $\mathcal{C}$ de leurs centres

L'ensemble $\mathcal{E}$ de $N$ données est découpé en fragments distribués aux nœuds de calcul ; un fragment doit tenir dans la mémoire d'un nœud ;

Un nœud initialise les $k$ centres ;

**while** *centres non stabilisés* **do**

  Transmettre l'ensemble $\mathcal{C}$ des centres à tous les nœuds de calcul ;

  Chaque tâche *Map(t)*, pour chaque élément $\mathbf{x}_i$ de son fragment $t$, trouve le centre le plus proche $j$ ; ensuite, pour chaque centre $j$ ainsi trouvé, génère $(j, (n_{jt}, \tilde{\mathbf{m}}_{jt} = \sum_t \mathbf{x}_i))$, où $n_{jt}$ est le nombre de données du fragment $t$ qui ont comme centre le plus proche le centre $j$ et la somme est faite sur les $\mathbf{x}_i$ du fragment $t$ plus proches du centre $j$ ;

  Les paires $(j, (n_{jt}, \tilde{\mathbf{m}}_{jt}))$ sont groupées par $j$ pour les tâches *Reduce* ;

  Chaque tâche *Reduce* reçoit toutes les paires correspondant à une valeur de $j$, calcule $\mathbf{m}_j = \frac{\sum_t \tilde{\mathbf{m}}_{jt}}{\sum_t n_{jt}}$ et stocke le $\mathbf{m}_j$ résultant ;

**end**

# Initialisation *K-means* : *K-means++*

- Une bonne initialisation de l'algorithme *K-means*
  - permet d'obtenir une solution de meilleure qualité et
  - une convergence plus rapide (avec moins d'itérations) vers cette solution
- Parmi les nombreux algorithmes d'initialisation nous considérerons *K-means++* [1]
- Idée : choisir les centres successivement, suivant une loi non uniforme qui privilégie les candidats éloignés des centres déjà sélectionnés

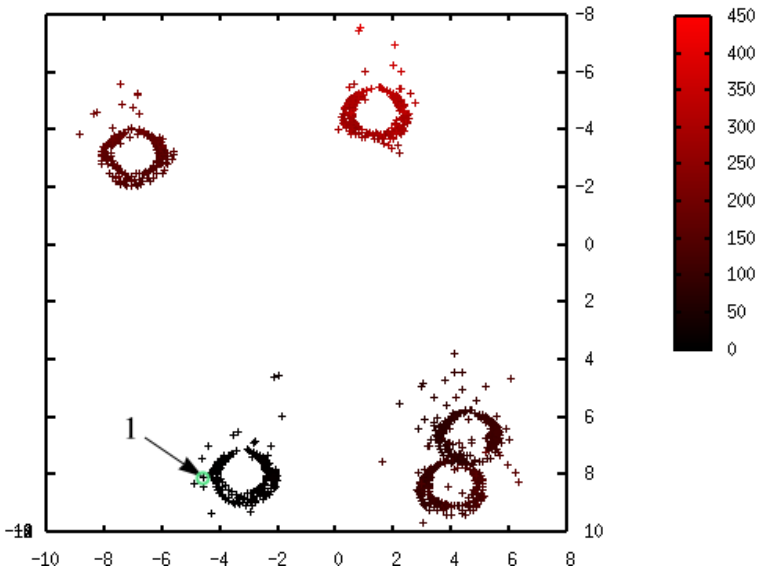
**Data :** Ensemble $\mathcal{E}$ de $N$ données de $\mathbb{R}^p$ ; nombre souhaité de centres $k$
**Result :** $\mathcal{C} = \{\mathbf{c}_j, 1 \leq j \leq k\}$

$\mathcal{C} \leftarrow$ un $\mathbf{x}$ de $\mathcal{E}$ choisi au hasard ;
**while** $\|\mathcal{C}\| \leq k$ **do**
  Sélectionner $\mathbf{x} \in \mathcal{E}$ avec la probabilité $\frac{d^2(\mathbf{x},\mathcal{C})}{\phi_{\mathcal{E}}(\mathcal{C})}$ ;
  $\mathcal{C} \leftarrow \mathcal{C} \cup \{\mathbf{x}\}$ ;
**end**

- Notations : $d^2(\mathbf{x},\mathcal{C}) = \min_{j=1,\ldots,t} d^2(\mathbf{x},\mathbf{c}_j)$, $\phi_{\mathcal{E}}(\mathcal{C}) = \sum_{\mathbf{x}\in\mathcal{E}} d^2(\mathbf{x},\mathcal{C})$
- Problème : *K-means++* n'est pas directement parallélisable

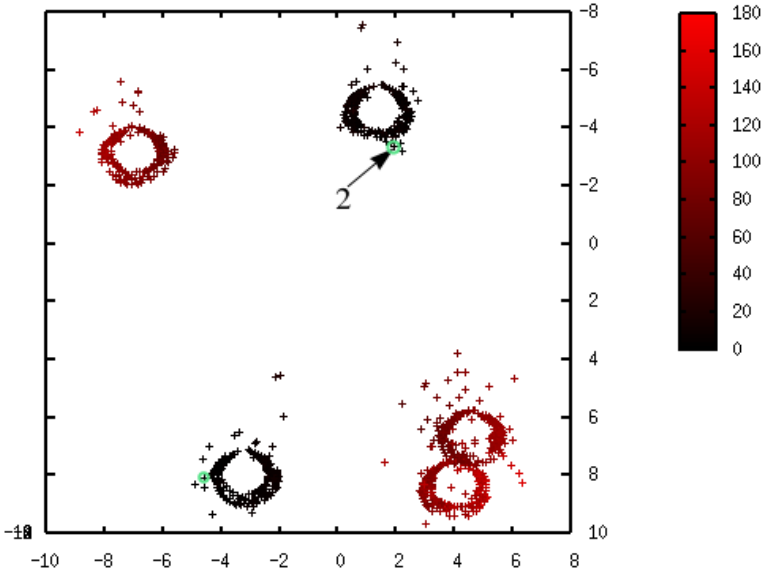
# *K-means++* : évolution des probabilités

(probabilité de sélection proportionnelle à $d^2(\mathbf{x}, \mathcal{C})$, représentée par la couleur rouge)




Après la sélection d'un point

$$\mathcal{C} = \left\{ \begin{pmatrix} -4,6 \\ 8,0 \end{pmatrix} \right\}$$

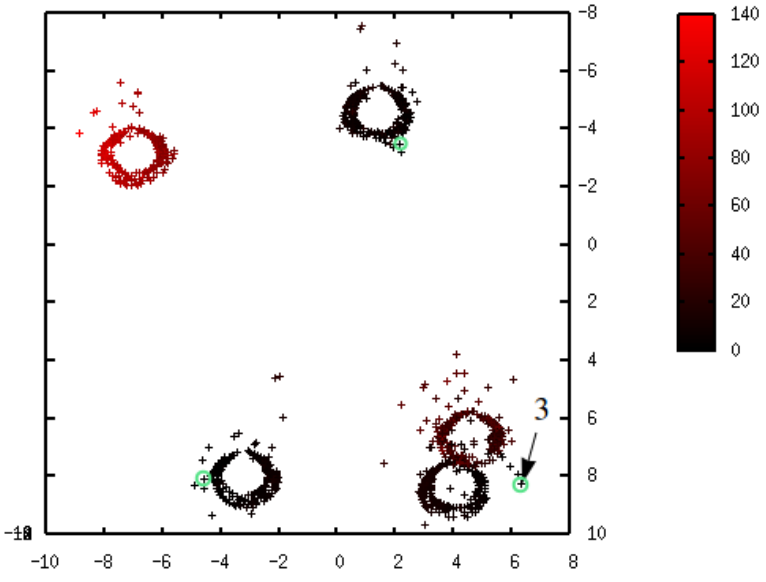



Après la sélection de 2 points

$$\mathcal{C} = \left\{ \begin{pmatrix} -4,6 & 2,15 \\ 8,0 & -3,45 \end{pmatrix} \right\}$$

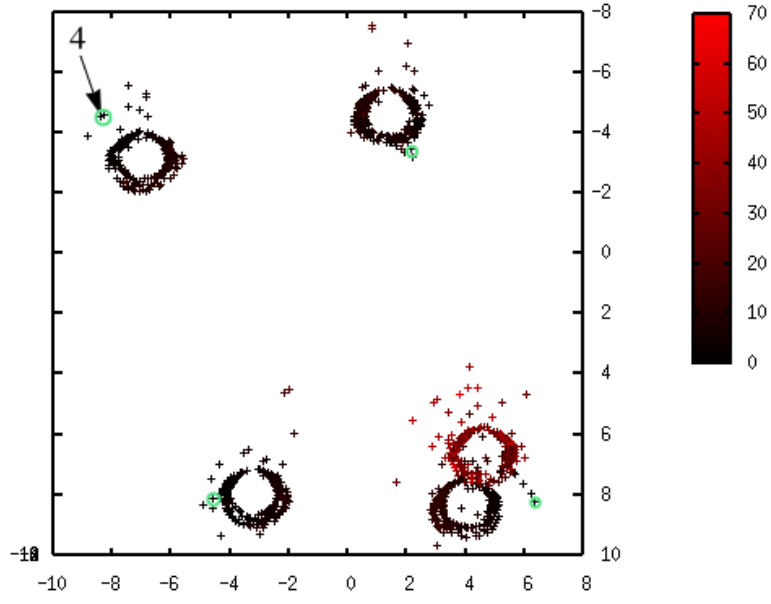
# *K-means++* : évolution des probabilités (2)

(probabilité de sélection proportionnelle à $d^2(\mathbf{x}, \mathcal{C})$, représentée par la couleur rouge)




Après la sélection de 3 points

$$\mathcal{C} = \left\{ \begin{pmatrix} -4,6 & 2,15 & 6,32 \\ 8,0 & -3,45 & 8,22 \end{pmatrix} \right\}$$




Après la sélection de 4 points

$$\mathcal{C} = \left\{ \begin{pmatrix} -4,6 & 2,15 & 6,32 & -8,37 \\ 8,0 & -3,45 & 8,22 & -4,54 \end{pmatrix} \right\}$$

# Initialisation *K-means* parallélisable : *K-means||*

- *K-means||* [2] proposé comme variante parallélisable de *K-means++*
- Idée : choisir plus qu'un centre à chaque itération, mais suivant une loi non uniforme

**Data :** Ensemble $\mathcal{E}$ de $N$ données de $\mathbb{R}^p$ ; nombre souhaité de centres $k$ ; degré de sur-échantillonnage $l \sim \Omega(k)$ ($l$ augmente au moins aussi vite que $k$)

**Result :** $\mathcal{C} = \{\mathbf{c}_j, 1 \leq j \leq k\}$

$\mathcal{C} \leftarrow$ un $\mathbf{x}$ de $\mathcal{E}$ choisi au hasard ;

$\psi \leftarrow \phi_{\mathcal{E}}(\mathcal{C})$ ;

**for** $O(\log \psi)$ *fois* **do**

  $\mathcal{C}' \leftarrow$ sélectionner chaque $\mathbf{x} \in \mathcal{E}$ indépendamment avec la probabilité $\frac{l \cdot d^2(\mathbf{x}, \mathcal{C})}{\psi}$ ;

  $\mathcal{C} \leftarrow \mathcal{C} \cup \mathcal{C}'$ ; $\psi \leftarrow \phi_{\mathcal{E}}(\mathcal{C})$ ;

**end**

**for** $\mathbf{m}_j \in \mathcal{C}$ **do**

  $w_m \leftarrow$ nombre de points de $\mathcal{E}$ plus proches de $\mathbf{m}_j$ que de tout autre point de $\mathcal{C}$ ;

**end**

Classification en $k$ groupes des données $\mathcal{C}$ pondérées par leurs poids $w_m$ ;

# *K-means||* : caractéristiques

- A l'étape (4), $\sum_{\mathbf{x}\in\mathcal{E}} \frac{l\cdot d^2(\mathbf{x},\mathcal{C})}{\psi} = l$, donc à chaque itération env. $l$ nouveaux points sont choisis
- L'étape finale traite un nombre réduit de données, $O(l \cdot \log\psi)$, elle peut donc se dérouler sur un seul nœud de calcul et employer *K-means++*
- Comme *K-means++*, *K-means||* donne des garanties de qualité de la solution obtenue (voir [2]) : $E[\phi_{\mathcal{E}}(\mathcal{C})] \leq O(\log k) \cdot \phi_{\mathcal{E}}(\mathcal{C}^*)$, $E$ étant l'espérance et $\mathcal{C}^*$ la solution optimale
- D'après [2], après 5 itérations on atteint déjà une très bonne solution, il n'est pas nécessaire de faire $O(\log\psi)$ itérations

# *K-means||* : implémentation MapReduce

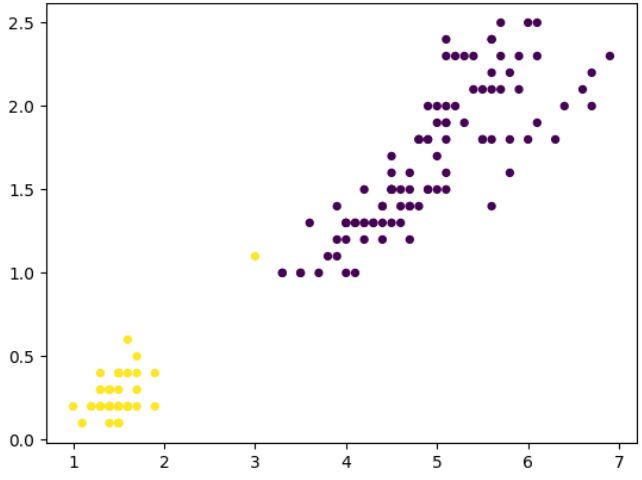
- Comme pour l'implémentation de *K-means*, après sa mise à jour dans l'étape (5), l'ensemble $\mathcal{C}$ des centres est transmis à tous les nœuds de calcul
- Le calcul de $\psi$ dans l'étape (5) est réalisé comme pour l'implémentation de *K-means*, le résultat est transmis à tous les nœuds de calcul pour l'étape (4)
- Le calcul des $d^2(\mathbf{x}, \mathcal{C})$ dans l'étape (4) peut être fait par chaque nœud de calcul pour les $\mathbf{x}$ de son fragment
- Les tirages de l'étape (4) sont réalisés de façon indépendante par les nœuds de calcul
- Le calcul des poids $w_m$ dans l'étape (8) est fait comme le calcul des $n_j$ dans l'implémentation de *K-means*
- La classification des données de $\mathcal{C}$ dans l'étape (10) peut être faite sur un seul nœud de calcul avec *K-means++*

# Classification hiérarchique

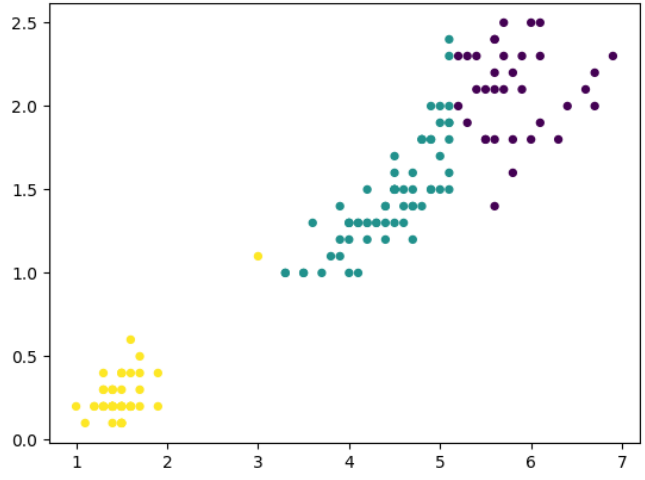
- Objectifs :
  - Obtenir une hiérarchie de groupes, structure plus riche qu'un simple partitionnement
  - Obtenir une solution moins dépendante de l'initialisation
- Permet d'examiner l'ordre des agrégations ou divisions de groupes, les rapports de similarités entre groupes, etc.
- Approches :
  - Classification ascendante : procède par agrégation des données et des groupes, complexité algorithmique $> O(N^2)$, parallélisation inefficace
  - Classification descendante : procède par division des groupes, complexité algorithmique $O(N)$, parallélisation efficace possible
    - Exemple : *bisecting k-means* (implémentée dans Spark)

# *Bisecting k-means* : principe

- Ensemble $\mathcal{E}$ de $N$ données décrites par $p$ variables à valeurs dans $\mathbb{R}$
- Objectif : répartir les $N$ données en $k$ groupes disjoints $\mathcal{E}_1, \ldots, \mathcal{E}_k$ (inconnus *a priori*) en optimisant la somme des inerties intra-classe (1)
- Principe : découpage récursif du groupe le moins compact en deux (sous-)groupes obtenus par application de *k-means* avec $k = 2$
- Illustration :




Itération 1 : données $\rightarrow$ 2 groupes




Itération 2 : groupe dispersé $\rightarrow$ 2 groupes

# *Bisecting k-means* : algorithme

**Data :** Ensemble $\mathcal{E}$ de $N$ données de $\mathbb{R}^p$

**Result :** $k$ groupes (*clusters*) disjoints $\mathcal{E}_1, \ldots, \mathcal{E}_k$ et ensemble $\mathcal{C}$ de leurs centres

Initialization : liste avec un seul groupe qui regroupe toutes les données ;
**while** *nombre groupes* $< k$ **do**
    Retirer de la liste le groupe qui a la plus grande inertie intra-classe ;
    **for** *i=1 à nb_essais* **do**
        appliquer *K-means* pour diviser ce groupe en 2 sous-groupes ;
    **end**
    Ajouter à la liste les 2 sous-groupes avec la plus faible somme des inerties intra-classe ;
**end**

- $\phi_{\mathcal{E}}(\mathcal{C})$ diminue à chaque itération
- Le découpage fait à une itération est affiné ultérieurement mais pas remis en cause !

# Comparaison entre *K-means* et *Bisecting k-means*

- Avantages *Bisecting k-means* :
  - Meilleure stabilité : à chaque itération *K-means* appliqué avec $k = 2$ et plusieurs essais
  - Coût inférieur car à chaque itération sont traitées les données d'un seul groupe
- Avantage *K-means* :
  - Potentiellement meilleur ajustement des groupes présents dans les données car le découpage en 2 groupes lors des premières itérations peut être trop grossier et n'est pas remis en cause ultérieurement (voir le point jaune isolé dans l'illustration initiale)

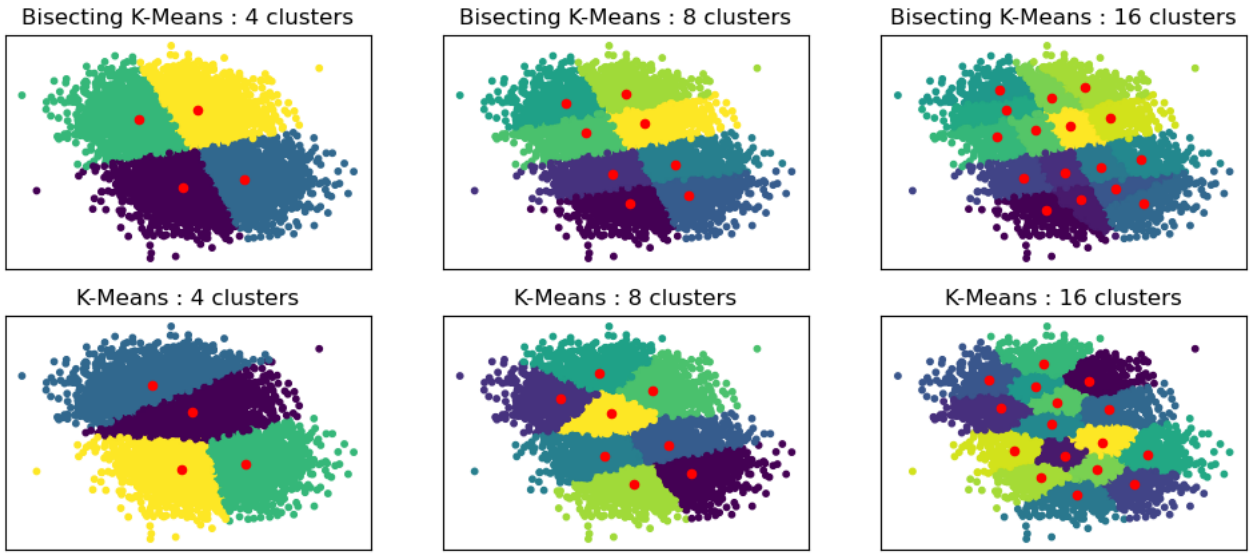



FIG. – Illustration issue de la documentation de Scikit-learn. L'objectif ici est la quantification vectorielle plutôt que le *clustering*, vu qu'il n'y a pas vraiment de groupes dans les données.

# La classification automatique dans Spark

- *K-means* : initialisation par *K-means*|| [2]
- Estimation de mélanges gaussiens (méthode d'estimation de densité) :
  - Ajout de variables auxiliaires et estimation par espérance-maximisation (*Expectation-Maximization*, EM)
  - EM : algorithme itératif, à chaque itération étape de calcul de l'espérance de la vraisemblance suivie d'étape de calcul des paramètres pour maximiser la vraisemblance
  - On obtient des équations de mise à jour facilement parallélisables
- *Bisecting k-means* : méthode de classification descendante hiérarchique
- *Power Iteration Clustering* (PIC [3]) : peut être vu comme une version plus efficace de la classification spectrale ; travaille sur la matrice des similarités entre données

# Références I



D. Arthur and S. Vassilvitskii. K-means++ : The advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, pages 1027–1035, Philadelphia, PA, USA, 2007. Society for Industrial and Applied Mathematics.


B. Bahmani, B. Moseley, A. Vattani, R. Kumar, and S. Vassilvitskii. Scalable k-means++. *Proc. VLDB Endow.*, 5(7) :622–633, 2012.


F. Lin and W. W. Cohen. Power iteration clustering. In J. Fürnkranz and T. Joachims, editors, *ICML*, pages 655–662. Omnipress, 2010.


J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In L. M. L. Cam and J. Neyman, editors, *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. University of California Press, 1967.