VII Algorithme des K moyennes

9 février 2025

1 Partitionnement et apprentissage non supervisé

Nous nous intéressons dans ce chapitre à un deuxième problème classique d'apprentissage : celui du *partitionnement de données* (en anglais : *clustering*). Il s'agit, au vu d'un certain nombre de données, de répartir ces données en un certain nombre de classes, de sorte que chaque classe contienne des données « semblables ».

Plus formellement, nous allons considérer des vecteurs $x_0, \dots, x_{n-1} \in \mathbb{R}^d$ et un nombre de classes $K \in [\![2, n]\!]$. Considérons une partition $G_0, \dots, G_{K-1}$ de $[\![0, n]\!]$ (*i.e.* une affectation des vecteurs x_i aux différentes classes), c'est-à-dire des parties $G_0, \dots, G_{K-1} \subset [\![0, n[\![$ telles que

$$\bigcup_{i=0}^{K-1} G_i = [\![0, n[\![\quad \text{et} \quad \forall 0 \leqslant i \neq j < K,\ G_i \cap G_j = \varnothing.$$

Pour une classe G_i, on peut définir son isobarycentre (ou moyenne, ou centroïde)

$$\mu_i = \frac{1}{\mathrm{Card}(G_i)} \sum_{j \in G_i} x_j$$

ainsi que sa *variance intra-classe*

$$V(G_i) = \frac{1}{\mathrm{Card}(G_i)} \sum_{j \in G_i} \|x_j - \mu_i\|^2 .$$

Remarque 1.0.1.
La variance intra-classe est un indicateur de dispersion : plus les points de la classe G_i seront proches du barycentre μ_i, plus cette variance sera faible.

Nous allons nous prendre comme critère d'évaluation la moyenne de ces variances intra-classe, pondérées par la taille de ces classes. Nous allons donc essayer de minimiser la quantité

$$\sum_{i=0}^{K-1} \mathrm{Card}(G_i) V(G_i) = \sum_{i=0}^{K-1} \sum_{j \in G_i} \|x_j - \mu_i\|^2 ,$$

les inconnues étant les classes $G_0, \dots, G_{K-1}$.

Remarque 1.0.2.
On parle ici d'*apprentissage non supervisé*. Le caractère non supervisé de l'apprentissage vient du fait que l'on ne travaille avec aucune donnée étiquetée, *i.e.* avec aucune indication initiale sur les classes $G_0, \dots, G_{K-1}$.

2 Algorithme des K-moyennes

De manière générale, la minimisation de la fonction précédente demande une recherche exhaustive sur toutes les classes possibles, ce qui n'est pas possible du point de vue de la complexité temporelle. Nous allons donc voir un algorithme classique permettant d'en donner une solution approchée : l'algorithme des K-moyennes.

Cet algorithme est très simple. On commence par initialiser les centroïdes.

Étape d'initialisation : choisir des centroïdes $\mu_0^0, \dots, \mu_{K-1}^0$ (par exemple : au hasard parmi les données).

Ensuite, à la p^{e} étape, si l'on a des centroïdes $\mu_0^p, \dots, \mu_{K-1}^p$ qui ont été construits, on enchaîne les deux étapes suivantes.

Étape d'affectation : affecter chaque vecteur x_j au centroïde le plus proche, ce qui donne des nouvelles classes $G_0^{p+1}, \dots, G_{K-1}^{p+1}$.

Étape de mise à jour : calculer les centroïdes $\mu_0^{p+1}, \dots, \mu_{K-1}^{p+1}$ de ces nouvelles classes $G_0^{p+1}, \dots, G_{K-1}^{p+1}$.

Il convient alors de se fixer un critère d'arrêt. Pour cela, il est important d'avoir conscience du théorème suivant.

Théorème 2.0.1.
La quantité

$$E(G_0^p, \ldots, G_{K-1}^p) = \sum_{i=0}^{K-1} \sum_{j \in G_i^p} \|x_j - \mu_i^p\|^2$$

décroît en fonction de p.

L'algorithme des K-moyennes converge vers un minimum local de la fonction E.

Les deux critères d'arrêts classiques sont les suivants :

— arrêt dès que deux itérations successives donnent les mêmes affectations (ou les mêmes centroïdes) ;
— arrêt dès que la quantité E a décru de moins qu'un seuil $\varepsilon > 0$ fixé au préalable.

Voici un premier exemple simpliste (voir figure 1), dans $\mathbb{R}^2$ et avec $n = 200$ points. Il semble que l'on ait ici trois paquets, on choisit donc de prendre $K = 3$, et l'on fait tourner l'algorithme des K-moyennes (voir figure 2).

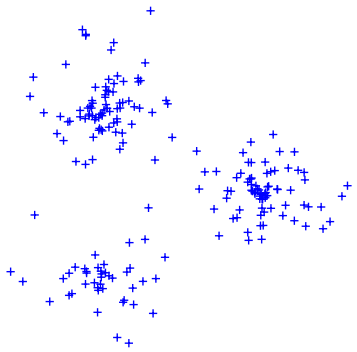
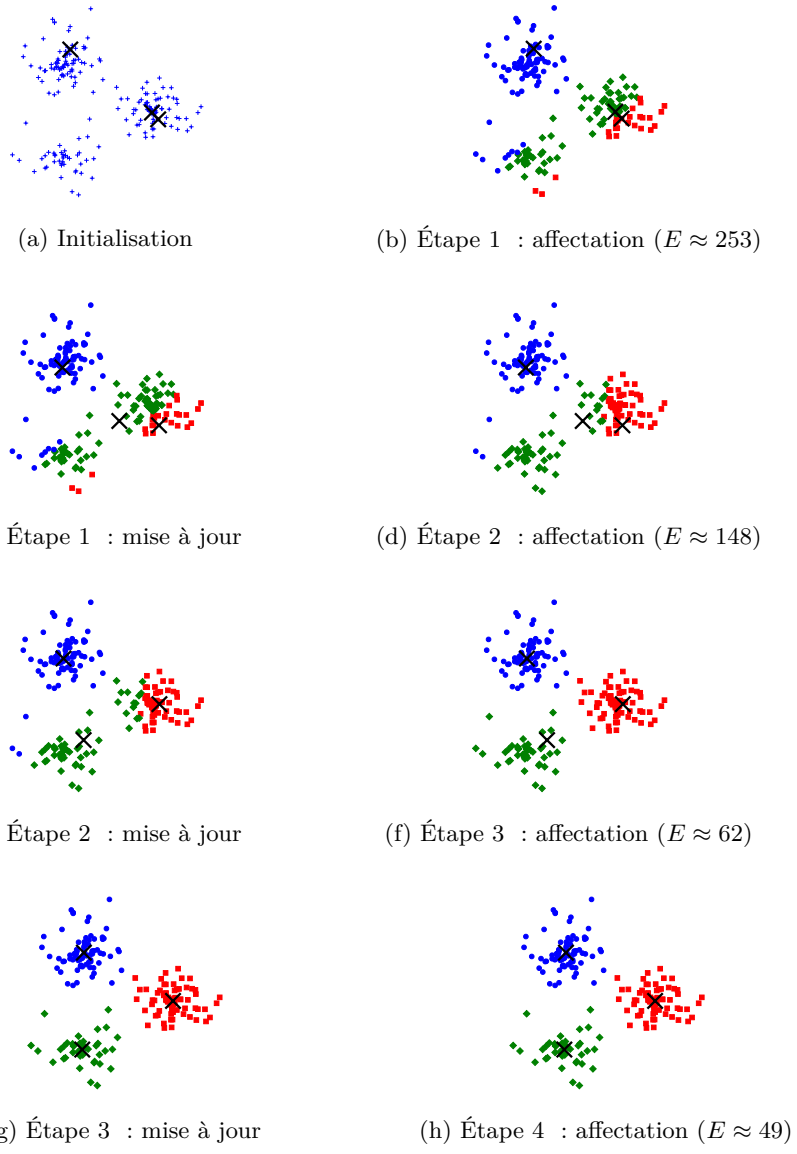


Figure 1 – Notre échantillon

On a représenté par des croix noires les centroïdes, et par trois types de marqueurs différents les points d'une même classe. Les centroïdes ont été initialisés aléatoirement. On a aussi calculé les valeurs de $E(G_1, G_2, G_3)$ après chaque étape d'affectation. On s'apperçoit qu'aux étapes 3 et 4, les affectations ne changent pas : on s'arrête donc là.

On a aussi représenté une situation semblable, mais beaucoup moins facile à séparer (voir figure 3). On a ici $n = 100$, les points sont tirés aléatoire de trois centroïdes prédéfinis, mais de manière bien plus dispersée que dans l'exemple précédent. On a fait tourner trois fois l'algorithme, à partir de trois distributions de centroïdes initiales différentes (à chaque fois tirées aléatoirement). À chaque fois, on a attendu la convergence complète de l'algorithme.

On remarquera que les trois situations sont fort différentes, et que la troisième est meilleure que les deux autres selon notre critère d'évaluation. Les deux premières configurations correspondent toutefois bien à une situation stable par notre transformation.



(a) Initialisation

(b) Étape 1 : affectation ($E \approx 253$)

(c) Étape 1 : mise à jour

(d) Étape 2 : affectation ($E \approx 148$)

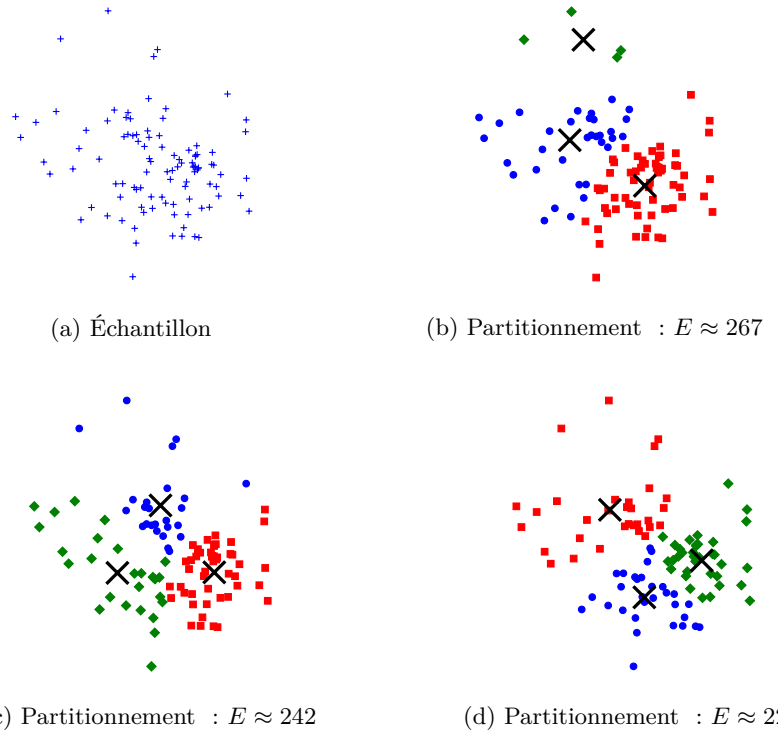
(e) Étape 2 : mise à jour

(f) Étape 3 : affectation ($E \approx 62$)

(g) Étape 3 : mise à jour

(h) Étape 4 : affectation ($E \approx 49$)

Figure 2 – Évolution des 3-moyennes sur notre exemple.



(a) Échantillon

(b) Partitionnement : $E \approx 267$

(c) Partitionnement : $E \approx 242$

(d) Partitionnement : $E \approx 222$

Figure 3 – Trois réalisations différentes de l'algorithme sur un même exemple.