
TRAITEMENT DES DONNÉES : K-MEANS

Introduction

Le clustering est un type d'apprentissage non supervisé, et K-means est un algorithme non supervisé du clustering. Ce type d'apprentissage vise à trouver des patterns dans les données afin de les regrouper.

Les données sont représentées comme suit :

$$X = \begin{pmatrix} x_{(1,1)} & \cdots & x_{(1,n)} \\ \vdots & \ddots & \vdots \\ x_{(m,1)} & \cdots & x_{(m,n)} \end{pmatrix}$$

Chaque ligne représente un individu (et donc une observation). Et suivant le principe de clustering, on pourra regrouper en plusieurs familles (clusters) les individus/objets en fonction de leurs caractéristiques/ressemblances.

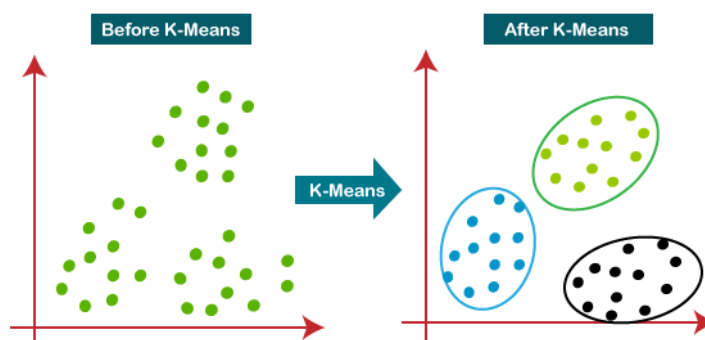
Ainsi, les individus se trouvant dans un même cluster seront similaires et les données se trouvant dans un autre cluster ne le seront pas.

Il existe deux types de clustering :

- Le clustering hiérarchique
- Le clustering non-hiérarchique (partitionnement)

L'exemple utilisé ci-dessus est un clustering non hiérarchique.

K-means tombe aussi dans cette catégorie, il permet de regrouper K clusters distincts à partir des observations de notre data set. Il existe aussi une exclusivité d'observations dans cet algorithme, c'est-à-dire qu'une observation ne peut se retrouver que dans un cluster à la fois (exclusivité d'appartenance) ...Une même observation, ne pourra donc, appartenir à deux clusters différents.



Théorie

Afin de regrouper un jeu de données en K cluster distincts, l'algorithme K-Means aura bien évidemment besoin d'un moyen de comparer le degré de similarité entre les différentes observations. Pour ce faire, on fait appel au concept de distance de dissimilarité.

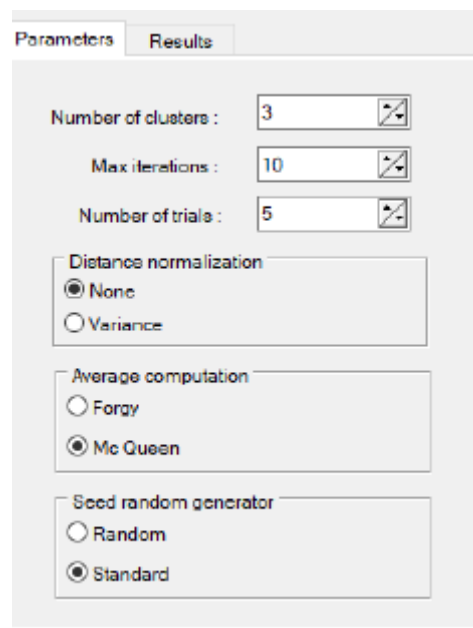
Ainsi, deux données qui se ressemblent, auront une distance de dissimilarité réduite, alors que deux objets différents auront une distance de séparation plus grande. Les plus connues sont la distance euclidienne et la distance Manhattan, mais on se concentrera sur le premier.

Sa définition est assez simple :

- Soit une matrice X à n variables quantitatives. Dans l'espace vectoriel E^n . La distance euclidienne d entre deux observations x_1 et x_2 se calcule comme suit :

$$d(x_1, x_2) = \sqrt{\sum_{j=1}^n (x_{1j} - x_{2j})^2}$$

Comme vu pendant notre TP utilisant Tanagra, choisir un nombre de cluster K n'est pas forcément intuitif. Encore moins quand le jeu de données est grand et qu'on n'ait pas un a priori ou des hypothèses sur les données. Un K trop grand peut conduire à un partitionnement trop fragmenté des données, qui à son tour empêchera de découvrir des patterns intéressants dans les données.



The screenshot shows the 'Parameters' tab of the Tanagra software. The 'Number of clusters' is set to 3, 'Max iterations' to 10, and 'Number of trials' to 5. Under 'Distance normalization', the 'None' radio button is selected. Under 'Average computation', the 'Mc Queen' radio button is selected. Under 'Seed random generator', the 'Standard' radio button is selected.

De même, un nombre de clusters trop petit est tout autant problématique car on aura des clusters trop généralistes contenant beaucoup de données. Dans ce cas, on n'aura pas de patterns "fins" à découvrir.

Il n'existe pas de procédé automatisé pour trouver le bon nombre K de clusters, la méthode habituelle étant de tester différentes valeurs de K et de calculer la variance des différents clusters.

La variance étant la somme des distances entre chaque centroid d'un cluster et les différentes observations incluses dans le même cluster.

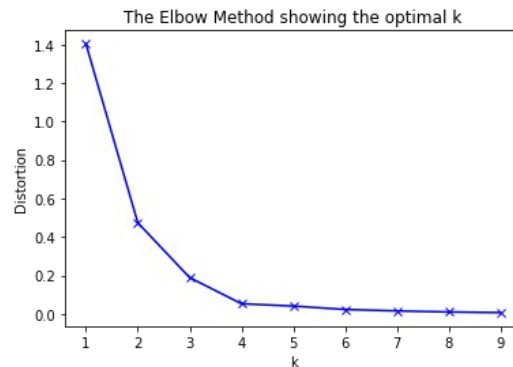
On cherche donc un K tel que les clusters retenus minimisent la distance entre leurs centres (centroids) et les observations dans le même cluster. On parle de minimisation de la distance intra-classe.

La variance des clusters se calcule comme suit :

$$V = \sum_j \sum_{x_i \rightarrow c_j} D(c_j, x_i)^2$$

Avec c_j centre du cluster, x_i la i ème observation dans le cluster ayant pour centroid c_j , et $D(c_j, x_i)$ la distance euclidienne entre le centre du cluster et le point x_i .

Et en schématisant la relation entre les deux, on peut retrouver un graphique similaire à :

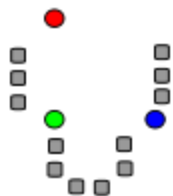


Ici le point optimal serait le point représentant le coude, donc $K=(2 \text{ ou } 3)$ car ce point représente généralement le point à partir du quel la variance ne se réduit plus aussi grossièrement (et donc une « chute » moins significative).

Avec le procédé maintenant plus clair, on peut passer à la description de l'algorithme de K-means, qui est tout compte fait, assez direct.

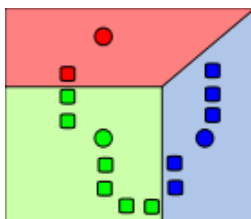
Initialisation

On commence par choisir, au hasard, k (ici $k=3$) centroïdes qui seront les centres des clusters de départ.

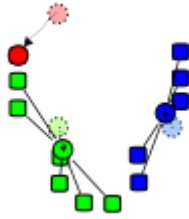


Boucle

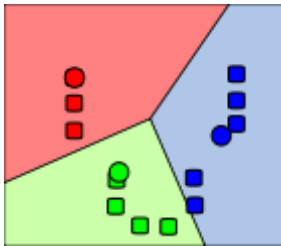
- **On construit k clusters** : Chaque point est dans le cluster du centroïde qui lui est le plus proche (en calculant la distance entre chaque point centroïde avec les autres).



- **On calcule les nouveaux centroïdes** : Pour chacun des clusters qu'on vient de former, on calcule la moyenne. Celle-ci devient le nouveau centroïde.



- **On recommence jusqu'à ce qu'à ce qu'il y ait convergence ou stabilisation de l'inertie totale du groupe** : La convergence correspond au fait que les centroïdes ne changent pas après une mise à jour.



Code Python

Le code ipynb **commenté** est joint au rapport pdf.