

Chapitre 1

État de l’art du clustering

Le terme clustering fait référence au concept de classification non supervisée. Ce concept fait lui-même partie d’une plus grande famille qu’est l’apprentissage non supervisé. À l’inverse de l’apprentissage supervisé qui nécessite des données déjà segmentées, le clustering vise quant à lui à déterminer une segmentation du jeu de données étudié. Dans ce cas, l’intervention humaine n’est pas nécessaire étant donné que l’ordinateur détermine les différentes segmentations sans l’apport de variables cibles fournies à l’algorithme. L’analyse de cluster permet donc d’identifier des groupes de données relativement homogènes sur la base de leur similitude basée sur des caractéristiques données. Dans notre cas, cela peut par exemple être le type d’emploi occupé en fonction des filières suivies par d’anciens étudiants. Cependant, analyser différents profils d’individus peut représenter des difficultés techniques importantes, c’est pourquoi cette première section du document présentera les différentes solutions possibles pouvant apporter une réponse à cette difficulté de classification des parcours.

1.1 La préparation des données

Lorsqu’une segmentation basée sur des clusters est utilisée, il existe plusieurs formes de préparations de données pouvant aider à la formation des différents segments.

1.1.1 La transformation des variables

Il existe deux types de transformation :

- La modification de la portée des variables connues
- La modification de la forme de distribution

L’utilisation de la standardisation est motivée par le fait que l’analyse de cluster implique une étude implicite du poids des objets afin de pouvoir se concentrer sur ceux possédant une variance plus élevée. Les méthodes de standardisation les plus communes sont les suivantes [?] :

- Multiplication de chaque variable par une différente constante.
- Utilisation des techniques de réduction de dimensions, qui est un processus visant à réduire le nombre de variables aléatoires afin d’obtenir un jeu de

variables principal.

- Multiplier chaque variable par une différente constante afin que chacune d'entre elles ait une portée commune.

La modification de la distribution quant à elle est motivée par les mêmes problématiques que dans d'autres secteurs ayant recours aux statistiques, qui sont d'extrêmes variations par rapport à ce qui est considéré «normal» dans le cas étudié. Celles-ci entraînent des analyses pouvant induire en erreur. Par conséquent lorsque le poids des variables est modifié le but est d'identifier et supprimer leur longue traine qui correspond à un nombre de variables possédant des valeurs très supérieures ou inférieures à la moyenne.

1.1.2 La détection d'outliers

Dans des secteurs comme le Data-Mining, un outlier ou données aberrantes est une observation se distinguant des autres par le fait qu'il représente une valeur extrême. À noter que cette observation peut être soit anormalement élevée ou basse selon le cas étudié. Par conséquent, une interprétation d'un jeu de données contenant un nombre important d'outliers peut amener à l'erreur. Ceci implique que la décision de considérer ou non les outliers doit être prise au moment de la construction du modèle de données. Il est à remarquer qu'en fonction du cas étudié, il n'existe pas de définition mathématique déterminant ce qui constitue un outlier. Toutefois, les causes retenues pouvant expliquer leur présence sont :

- L'erreur humaine
- Une erreur lors de la transmission/transcription des données
- Un changement de procédure

Des méthodes existent cependant afin d'aider à la détection telles que la droite de Henry ou encore la technique des boîtes à moustaches.

La droite de Henry

La droite de Henry est méthode graphique permettant d'évaluer «la normalité» d'une distribution pour une série d'observations. Cela permet à la lecture du graphique obtenu de repérer rapidement la moyenne et l'écart type du jeu de données traité.(figure 1.1)

Les points indiquent toutes les valeurs des fréquences réelles pour les différentes valeurs attribuées à une variable. Si la distribution est considérée comme «normale» alors tous les points devraient se trouver sur la droite.

la technique des boîtes à moustaches

Une boîte à moustaches ou box-plot est, en statistiques, une représentation graphique permettant pour un jeu de données de représenter la médiane, les quartiles et les centiles. Cette méthode est principalement utilisée afin de comparer un même caractère dans deux populations de tailles différentes.

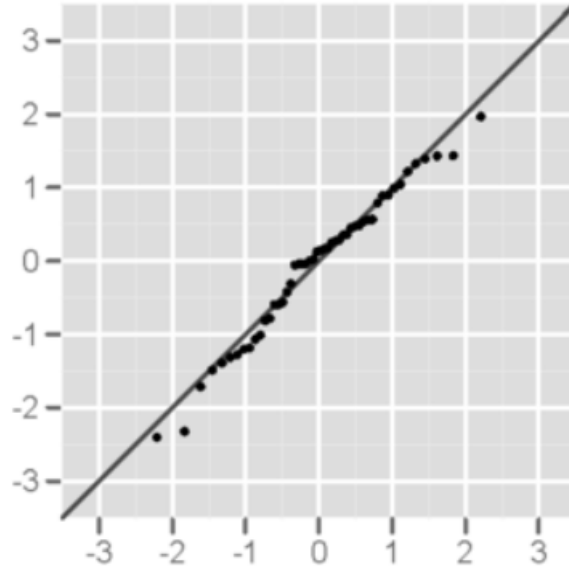


FIGURE 1.1 – Représentation d'une droite de Henry

1.2 La mesure de distance

Le choix d'une méthode de mesure de distance est une étape critique pour les méthodes de clustering, son choix ayant une très forte influence sur le résultat final. En effet, la méthodologie choisie définira comment les ressemblances entre deux éléments sont calculées et influera par conséquent sur la forme des clusters également. Les deux méthodes les plus communes de mesure sont la distance euclidienne illustrée par la formule suivante (figure1.2) :

$$d_{euc}(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

FIGURE 1.2 – Distance euclidienne

Dans cette méthode, la distance est calculée en effectuant le carré de l'ensemble des distances mises au carré pour toutes les variables satisfaisant un critère donné. La seconde méthode communément utilisée est la distance de Manhattan, appelée également «taxi-distance». Celle-ci, pour un point A et B de coordonnées respectives (X_a, Y_a) et (X_b, Y_b) est définie de la façon suivante :

$$d(A, B) = [X_b - X_a] + [Y_b - Y_a]$$

FIGURE 1.3 – Distance de Manhattan

À l'inverse de la méthode euclidienne qui pourrait être influencée par des valeurs inhabituelles, le calcul de la distance de Manhattan va s'effectuer selon la différence moyenne entre les dimensions. La présence de valeurs aberrantes impactera le résultat de façon réduite étant donné qu'elles ne seront pas élevées au carré contrairement à la méthode euclidienne. Cette méthode aura donc tendance à donner le même type de résultat.

1.3 Les types de clusters

La finalité du clustering étant de trouver des groupes d'objets présentant des similarités définies en fonction du but recherché. Il existe toutefois une multitude de types de cluster qui seront étudiés au sein de cette section. Chacun sera présenté avec ses avantages et inconvénients en fonction de notre cas avant de statuer sur le type qui sera utilisé pour le reste de ce document.

1.3.1 Well-Separated

Un cluster «well-separated» est un regroupement de points de telle façon à ce que tous les points faisant partie d'un même cluster présentent de fortes ressemblances entre eux comparées aux points d'un cluster extérieur (figure 1.6). Si la population du cluster est suffisamment bien compartimentée, ce type de cluster permet de faire fonctionner n'importe quelle méthode de clustering de façon efficace.

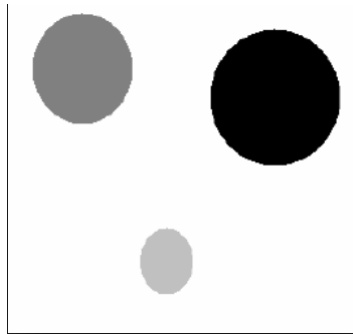


FIGURE 1.4 – Well separated clusters

1.3.2 Graph-Based

Le graph-based cluster est utilisé dans les cas où les données peuvent être représentées sous forme de graphe dont les nœuds sont des objets et les liens représentent les connexions entre ceux-ci. Dans cette situation, un cluster peut être défini comme un composant connecté. C'est-à-dire un groupe d'objets liés les uns aux autres au sein du même groupe. Ce type de classification permet de visualiser facilement d'importants jeux de données. La théorie des graphes peut être utilisée afin d'apporter des informations plus précises sur le jeu de données en ce qui concerne les clusters, cliques et outliers. Une approche hiérarchique peut être employée à travers la création d'un arbre couvrant de poids minimal ou minimum spanning tree. Dans un contexte où un graphe G est connexe, l'arbre est un sous-graphe connexe avec un poids minimal contenant tous les nœuds du graphe d'origine G et ne possédant pas de cycle. Il existe deux algorithmes permettant d'établir ce type d'arbre, l'algorithme de Prim et l'algorithme de Kruskal.

L'algorithme de Prim

L'algorithme de Prim est un algorithme dit « glouton », c'est-à-dire qu'il va pour chaque étape effectuer un choix local jugé optimum. Cet algorithme (figure ...) va s'exécuter en choisissant un axe de poids minimal jusqu'à ce que tous les sommets aient été atteints. Il existe cependant une part d'indéterminisme, il est en effet possible que deux exécutions de cet algorithme renvoient chacun une solution différente. À noter que celle-ci sera tout de même optimale.

```
Initialiser  $T$  avec
{ sommets : un sommet de  $G$  qu'on choisit
  arêtes : aucune
Répéter :
  • Trouver toutes les arêtes de  $G$  qui relient
    un sommet de  $T$  et un sommet extérieur à  $T$ 
  • Parmi celles-ci, choisir une arête de poids
    le plus petit possible
  • Ajouter à  $T$  cette arête et le sommet
    correspondant
S'arrêter dès que tous les sommets de  $G$ 
sont dans  $T$ 
Retourner  $T$ 
```

FIGURE 1.5 – Algorithme de Prim

1.3.3 Density-Based

A travers l'utilisation d'un density-based cluster, le but est de détecter les zones ou les points formant des clusters sont concentrés et celles ou les points sont séparés par des zones vides ou par des zones contenant très peu de points (figure 1.4). Les points ne faisant pas partie d'un agrégat sont ici considérés comme du bruit. Ce type de définition est utilisée lorsque les clusters s'entrecroisent ou sont irréguliers.[?] Il est également possible d'avoir recours à ce type de classification lorsque du bruit est présent, celui-ci peut former des ponts entre les clusters lorsqu'un autre type de classification est utilisée.

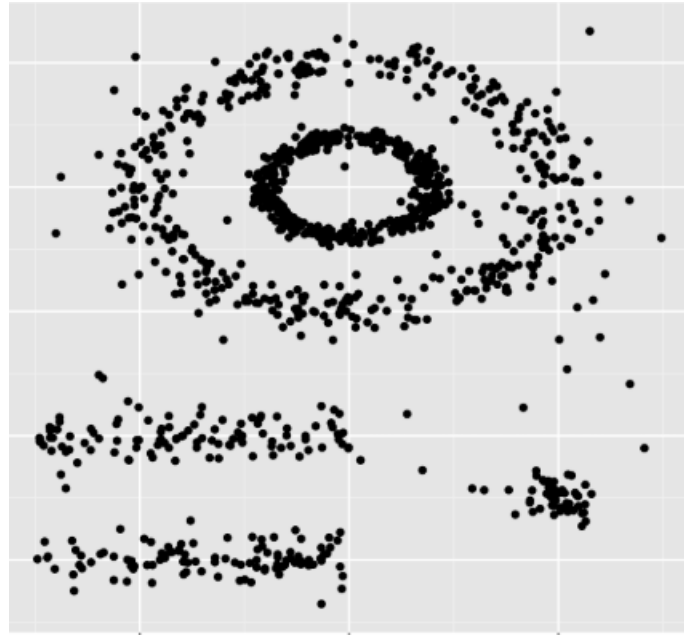


FIGURE 1.6 – Density based clusters

1.4 Les méthodes et algorithmes

Dans cette section seront décrits les principaux algorithmes utilisés lorsque des techniques de clustering sont employées. Les avantages et inconvénients de chacun seront présentés en fonction du cas présenté dans l'introduction.

1.4.1 Le clustering hiérarchique

Parmi les différents types de clustering existants [?], le premier étudié sera le clustering hiérarchique. Très utilisé comme outil d'analyse de données, l'idée principale de cette méthode est de construire un arbre binaire fusionnant de façon successive les groupes de points similaires. L'un des avantages de cette façon de procéder est tout d'abord l'apport de l'arbre qui permet d'avoir une vision globale des données traitées. De plus, cette méthodologie possède ses propres outils de visualisation qui sont le dendrogramme et la classification double. Le dendrogramme permet d'illustrer l'arrangement des clusters (figure 1.3)[?] :

- la racine de l'arbre est formée par un cluster contenant l'ensemble des objets.
- chaque nœud de l'arbre constitue un cluster.
- l'union des objets contenus par les nœuds fils correspond aux objets présents dans le nœud racine.
- les paliers sont indexés en fonction de l'ordre de construction.

Tandis que la classification double est une technique d'exploration de données non supervisées permettant de segmenter concurremment les lignes et les colonnes d'une matrice. L'autre avantage du clustering hiérarchique est sa facilité d'implémentation dans des algorithmes tels que K-Means en plus de fournir une représentation graphique comme dit précédemment. Afin d'établir un arbre hiérarchique, le processus de clustering a recours à deux méthodes qui sont la méthode agglomérative et la méthode divisive.

Un regroupement agglomératif traite chaque objet comme un seul élément qui à chaque étape de l'algorithme est fusionné avec un second objet présentant le plus de similarités en un nouveau cluster de plus grande taille. Ce processus est répété jusqu'à ce que tous les points soient membre d'un seul et même cluster. À l'inverse d'un regroupement agglomératif qui utilise une approche «bottom-up», les algorithmes divisifs utilisent une approche «top-down». Ces algorithmes débutent ainsi leur traitement à partir de la racine de l'arbre ou tous les objets sont regroupés en un seul cluster. À chaque itération, les clusters les plus hétérogènes sont divisés en deux jusqu'à ce que l'ensemble des objets fassent partie de leur propre cluster. Toutefois, sa complexité le rend inefficace sur de larges jeux de données [?]. De plus, la première injection de données ainsi que l'ordre de celles-ci à un fort impact sur le résultat final. En outre, il n'est pas possible de défaire ou modifier les étapes précédentes du traitement. Une fois une instance assignée à un cluster, il n'est plus possible de la déplacer pour effectuer d'éventuelles modifications ou corrections [?]. Dans notre cas la base de CV utilisée n'étant pas de taille importante le clustering hiérarchique reste une méthode applicable.

Cependant, la problématique à résoudre est la gestion des filières intégrant plusieurs domaines comme la filière MASHS de Nanterre qui possède une dimension mathé-

matique et une informatique. Les données étant représentées sous forme d'arbre cela entrainerait une répétition au niveau des résultats.

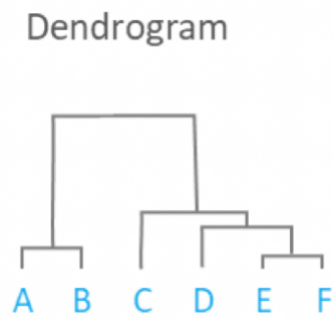


FIGURE 1.7 – Exemple de dendrogramme

1.4.2 Le clustering par partitionnement

Le clustering par partitionnement contrairement au clustering hiérarchique qui utilise un arbre afin de représenter les différents groupes de données va classer les différents objets en groupe en fonction de leurs similarités. Cependant ce mode de fonctionnement pose un problème concernant le choix de la «bonne représentation» en fonction d'un critère choisi, le but devient alors de rechercher une représentation optimale de son critère à travers plusieurs itérations.[?] L'algorithme le plus utilisé pour ce type de méthode est K-means qui sera présenté dans la suite de ce document. Dans le cas du clustering par partitionnement, il existe une sous-catégorie nommée le partitionnement flou. Celle-ci a recours à une autre version de l'algorithme k-means appelée *Fuzzy k-means* ou k-moyennes flou. Cet algorithme cherche à minimiser la fonction illustrée par la figure 1.4.

$$\sum_{j=1}^k \sum_{x_i \in C_j} u_{ij}^m (x_i - \mu_j)^2$$

FIGURE 1.8 – Fonction objectif de Fuzzy K-Means

Dans la figure ci-dessus (figure 1.5),

- u_{ij} est le degré selon lequel une observation x_i appartient à un cluster c_j .
- μ_j est le centre d'un cluster j .
- m est le «fuzzifier».

Contrairement aux méthodes de clustering dites dures, lorsque l'on a recours au clustering flou, il est possible que des données fassent partie de plusieurs clusters. De plus, l'appartenance d'un objet à un cluster est ici décidée par la proximité entre le centre du cluster traité et l'objet en question. [?] Cet algorithme reste toutefois soumis aux mêmes restrictions que l'algorithme K-Means classique dans le fait qu'il requiert de spécifier le nombre de clusters souhaités.

1.4.3 Le clustering basé sur des mélanges de modèles

Cette approche est adoptée lors de l'utilisation d'apprentissage automatique et au traitement de données manquantes ou cachées. Par rapport à d'autres approches basées sur des métriques permettant de déterminer le nombre de classes nécessaires ou encore le degré d'incertitude [?]. Les limites de cette méthode résident principalement dans les limitations entraînées par le type de données utilisées et la nécessité de formuler des hypothèses sur la distribution des observations rarement vérifiables dans la réalité.[?]

1.4.4 Le clustering conceptuel

Le clustering conceptuel est un paradigme d'apprentissage non supervisé ayant fait son apparition durant les années 1980. Cette méthode se différencie du cluste-

ring de données classique par le fait qu'elle génère une description de concept pour chaque classe générée. Un autre facteur de différenciation est le fait que les phases de clusterisation et de caractérisation des données ne sont pas indépendantes.[?] Un exemple d'algorithme se basant sur cette méthode est COBWEB(figure 1.5) qui est un algorithme utilisant les concepts du clustering hiérarchique et produira par conséquent en sortie une hiérarchie de concept. La limite de cet algorithme est le fait que celui-ci traite difficilement les attributs numériques[?], de plus, bien qu'étant utilisé pour la découverte de classes d'objets dans d'importants volumes de données [?]. Le stockage de l'ensemble des instances d'une hiérarchie peut représenter une nouvelle problématique sur des volumes importants de données.

```

Function Cobweb (object, root)
  Incorporate object into the root cluster;
  If root is a leaf then
    return expanded leaf with the object;
  else choose the operator that results in the
  best clustering:
    a) Incorporate the object into the best host;
    b) Create a new class containing the object;
    c) Merge the two best hosts;
    d) Split the best host;
  If (a) or (c) or (d) then
    call Cobweb (observation, best host);

```

FIGURE 1.9 – Algorithme Cobweb

L'algorithme de Kruskal

Créé en 1956 par Joseph Kruskal, cet algorithme utilise comme entrée un graphe connexe non orienté et pondéré (figure 1.8) et va considérer chaque arête par ordre croissant. Si l'arête sélectionnée ne crée pas de cycle, celle-ci sera utilisée dans la construction d'un arbre couvrant de poids minimum.

Les algorithmes de Prim et Kruskal possédant chacun une part d'indéterminisme, il est tout à fait possible que chacun renvoie un arbre différent tout en utilisant le même graphe de départ. Les solutions données seront toutefois jugées optimales pour les deux algorithmes.

- Initialiser T avec
 - $\left\{ \begin{array}{l} \text{sommets : tous les sommets de } G \\ \text{arêtes : aucune} \end{array} \right.$
- Traiter les arêtes de G l'une après l'autre par poids croissant :
 - Si une arête permet de connecter deux composantes connexes de T ,
 - alors l'ajouter à T
 - sinon ne rien faire
 - Passer à l'arête suivante
- S'arrêter quand il n'y a plus d'arêtes
- Retourner T

FIGURE 1.10 – Algorithme de Kruskal

1.4.5 K-means

L'algorithme K-means est l'algorithme le plus populaire, celui-ci peut être utilisé dans plusieurs domaines tels que :

- Utilisation du clustering dans un contexte de Data Mining.
- Clustering de documents qui dans notre cas seraient les CV d'anciens étudiants.
- La segmentation d'un jeu de données en fonction de critères.

Celui-ci recherche la meilleure division possible au sein d'un jeu de données [?] et possède comme avantage une certaine facilité d'implémentation. Cependant, il impose de connaître le nombre de clusters K souhaités et par conséquent une bonne connaissance des données utilisées. L'une des solutions possibles lorsqu'un grand jeu de données est utilisé afin de déterminer le nombre de clusters voulus est de lancer l'algorithme avec différentes valeurs et de calculer ensuite la variance entre les résultats obtenus. Celle-ci représente ainsi la somme des distances entre chaque *centroïde*.

K-means fonctionne de la façon suivante :

- Choix d'un nombre de clusters K
- Affectation de chaque point au groupe dont il est le plus proche.
- Itération jusqu'à ce qu'il n'y ait plus de changements au niveau des centroïdes, c'est-à-dire que ceux-ci ne bougent plus lors des itérations.

Dans notre cas qui est la clusterisation de documents en fonction des types de parcours. Ceux-ci représentant différents tags en fonction du document étudié, nous sommes par conséquent dans un problème classique de classification pour lequel K-Means représente une solution possible. Dans cette situation, un premier traitement de chaque document est requis afin de pouvoir les représenter sous forme de vecteurs. Il est ensuite possible d'utiliser la fréquence d'apparition de certains termes afin d'identifier les mots les plus fréquents et ainsi aider à la classification du document étudié. Les vecteurs de documents sont ensuite clusterisés afin d'identifier les similarités dans un groupe de documents. À noter qu'il existe d'autres applications possibles à K-means telles que :

- Segmentisation de clientèle
- Analyse d'appels enregistrés
- Prédiction de crime [?]

1.4.6 Agglomerative Hierarchical Clustering

Les techniques de clustering agglomératives partent d'un ensemble de points formant un cluster. Ensuite, les deux clusters les plus proches sont fusionnés successivement jusqu'à ce qu'il n'y est plus qu'un unique cluster. [?] Les techniques de clustering agglomératives peuvent fonctionner de deux façons, ascendante ou descendante. Ce type d'approche figure parmi les plus utilisés lorsqu'il existe un besoin de regrouper des objets au sein de clusters basés sur leur similarité. L'agglomerative clustering est également connu sous le nom d'AGNES (Agglomerative Nesting) tandis que l'agglomerative hierarchical clustering peut être également appelé HAC. Lorsque la méthode bottom-up est utilisée, cet algorithme fonctionne la façon suivante :

- Chaque objet est traité en tant que cluster sous la forme d'un singleton.
- Les paires de clusters sont successivement fusionnées jusqu'à ce que tous les clusters fassent partie d'un unique cluster contenant tous les objets.

Le résultat de ce traitement est un dendrogramme (figure 1.3) représentant les objets.[?] En utilisant la méthode descendante, une méthode séparant les clusters est nécessaire. L'algorithme dans ce cas de figure sépare les clusters de façon récursive jusqu'à ce que tous les objets fassent partie de leur propre cluster. À noter qu'également dans cette méthode, les clusters sont considérés comme des singletons. L'avantage de l'approche ascendante, est que celle-ci peut se révéler plus rapide si l'on se trouve dans un cas où il n'est pas nécessaire de générer une hiérarchie dans son intégralité.[?] L'inconvénient de ces procédés est qu'ils possèdent une complexité O_n^3 tout en nécessitant O_n^2 de mémoire ce qui le rend trop lent même sur des jeux de données de taille moyenne. Il existe toutefois des méthodes agglomératives possédant la même complexité, utilisées dans des cas particuliers qui sont le single linkage et complete linkage clustering. À l'exception de ces deux autres méthodes, il n'est pas garanti qu'une solution optimale soit trouvée en ayant recours à ces algorithmes à moins de fournir une heuristique permettant d'atteindre une solution.

1.4.7 DBSCAN

DBSCAN est un algorithme basé sur le partitionnement de données. Cet algorithme utilise deux principaux paramètres qui sont la distance minimale entre deux points et le nombre de points minimum devant se trouver dans un rayon donné afin qu'ils soient considérés comme un cluster[?]. Le choix d'une bonne mesure de la distance reste critique comme dans toute autre méthode de clustering. En effet, si la valeur choisie est trop petite, une partie des données risque d'être considérée en tant qu'*outlier*. C'est-à-dire des observations peu fréquentes sortantes de la norme. L'algorithme DBSCAN sert dans des situations où l'on cherche à déterminer des structures au sein d'un jeu de données. Celui-ci peut être exprimé en pseudo-code.(figure 1.5). Dû à sa popularité, cet algorithme est souvent directement importé à travers des libraires Python ou R. Cet algorithme offre comme avantages le fait qu'il n'a

Algorithm 1 The DBSCAN algorithm. Input: A set of points X , distance threshold eps , and the minimum number of points required to form a cluster, $minpts$. Output: A set of clusters.

```
1: procedure DBSCAN( $X, eps, minpts$ )
2:   for each unvisited point  $x \in X$  do
3:     mark  $x$  as visited
4:      $N \leftarrow \text{GETNEIGHBORS}(x, eps)$ 
5:     if  $|N| < minpts$  then
6:       mark  $x$  as noise
7:     else
8:        $C \leftarrow \{x\}$ 
9:       for each point  $x' \in N$  do
10:         $N \leftarrow N \setminus x'$ 
11:        if  $x'$  is not visited then
12:          mark  $x'$  as visited
13:           $N' \leftarrow \text{GETNEIGHBORS}(x', eps)$ 
14:          if  $|N'| \geq minpts$  then
15:             $N \leftarrow N \cup N'$ 
16:          if  $x'$  is not yet member of any cluster then
17:             $C \leftarrow C \cup \{x'\}$ 
```

FIGURE 1.11 – Algorithme DBSCAN

pas besoin de plus de paramètres que la distance, ici marquée eps et le nombre de moins minimum afin de constituer un cluster représenté ici par $minpts$. De plus, celui-ci n'est pas très sensible à l'ordre des données pour son traitement. Cependant, la qualité du résultat est grandement liée au choix d'une bonne mesure de distance. En outre, un degré de compréhension de l'échelle appliquée et des données étudiées est nécessaire afin de pouvoir choisir une mesure de distance pertinente.

Une étude a montré qu'il est possible de combiner son propre algorithme avec DBSCAN et d'utiliser celui-ci pour tout ce qui concerne la détection du bruit dans un jeu de données. Cet algorithme étant capable de détecter dans un rayon donné tous les points de donnée. Dans un cas où plusieurs exécutions de DBSCAN montrent qu'un point est isolé, c'est-à-dire que celui-ci est dans une zone à faible densité. Ce point peut être réellement considéré comme du bruit et il est possible de le supprimer afin de perfectionner le jeu de données étudié.[?]

1.5 Analyse et conclusion

Nous avons relevé dans un premier temps dans la figure ci-dessous les principaux avantages et inconvénients de chacune des méthodes étudiées précédemment.

| Méthode de clustering | Avantages | Inconvénients |
|-----------------------|--|--|
| Hiérarchique | Ne nécessite pas d'informations sur le nombre de clusters requis Facilité d'implémentation | Qualité du résultat dépendant de la méthode utilisée (Bottom-up ou Top-down) Scalabilité Lisibilité des résultats sur d'important jeu de données |
| Par partitionnement | Facilité d'implémentation Scalable | Modifications non possibles après la fin du traitement. Pauvre interprétabilité des résultats |
| Conceptuel | Capable de détecter les variables représentant du bruit Fonctionne sur une grande variété de données. | Traitement difficile de données numériques. Le stockage des résultats sur d'important jeu de données |
| Mélange de modèles | Reste fonctionnel même avec la présence de données cachées ou manquantes | Sensible à la qualité des hypothèses formulées Dépendant du type de données utilisées. |

FIGURE 1.12 – Comparatif des méthodes de clustering

Avec ce premier tableau, nous pouvons éliminer des méthodes potentiellement utilisables, la méthode conceptuelle. Celle-ci traitant difficilement les données numériques, n'est par conséquent pas adaptée à notre cas. La méthode par mélange de modèles est également non utilisable, la qualité des résultats étant très fortement liée aux hypothèses formulées par un utilisateur en début de traitement. Afin de pouvoir choisir entre les deux méthodes restantes, la plus adaptée à notre cas, nous nous baserons sur les mêmes critères que ceux énoncés dans l'introduction . C'est-à-dire :

- Capacité de la méthode à traiter un jeu de données numérique
- La difficulté d'implémentation
- La sensibilité au bruit
- La scalabilité de la méthode

La figure ci-dessous présente la méthode hiérarchique et par partitionnement au vu de ces critères.

| Méthode de clustering | Difficulté d'implémentation | Scalabilité | Sensibilité au bruit | Capacité à traiter des données numériques |
|-----------------------|-----------------------------|--|---|---|
| Hiérarchique | Facilité d'implémentation | Dépendante de l'approche choisie (Top-Down ou Bottom-up) | Sensibilité dépendante de la méthode de linkage choisie | Fonctionne sur des données numériques |
| Par partitionnement | Facilité d'implémentation | K-means applicable sur de large jeu de données | Tous les objets sont assignés à des clusters | Ne fonctionne uniquement que sur des données numériques |

FIGURE 1.13 – Évaluation des méthodes de clustering hiérarchique et par partitionnement

Comme on peut l'observer dans ce tableau, les deux méthodes possèdent toutes deux une implémentation simple. Toutefois, l'approche hiérarchique est fortement dépendante de la méthode utilisée. Tandis que la méthode par partitionnement est applicable sur de larges jeux de données sans être sensible à une méthodologie extérieure particulière en plus de posséder la même facilité d'implémentation. Par conséquent, la méthode choisie pour notre implémentation sera l'approche par partitionnement.