Initialisation des centres

Sommaire

4.1 Introduction 81
4.2 État de l'art 82
4.2.1 Les méthodes ayant une complexité linéaire en N 82
4.2.2 Les méthodes ayant une complexité log-linéaire en N 85
4.2.3 Les méthodes ayant une complexité quadratique en N 85
4.3 Contribution 86
4.3.1 K-means++R [70] 87
4.3.2 Méthodes basées sur la variance : Rocchio-And-Split et S-Bisecting [11, 60] 89
4.4 Protocole expérimental 93
4.5 Cas où le nombre de clusters (K) est égal au nombre de classes (J) 95
4.6 Cas où le nombre de clusters (K) est supérieur au nombre de classes (J) 99
4.6.1 Évaluation de la prédiction 99
4.6.2 Évaluation de la compacité 100
4.6.3 Évaluation du compromis 101
4.7 Bilan et synthèse 104

Ce chapitre a fait l'objet des publications suivantes :

[11] Oumaima Alaoui Ismaili, Vincent Lemaire, and Antoine Cornuéjols. Une initialisation des K-moyennes à l'aide d'une décomposition supervisée des classes. *Congrès de la Société Française de Classification (SFC)*, Nantes, 2015.

[60] Oumaima Alaoui Ismaili, Vincent Lemaire, and Antoine Cornuéjols. Une méthode supervisée pour initialiser les centres des k-moyennes. *Extraction et Gestion des Connaissances (EGC), Reims*, 2016.

[70] Vincent Lemaire, Oumaima Alaoui Ismaili, and Antoine Cornuéjols. An initialization scheme for supervized k-means. In *2015 International Joint Conference on Neural Networks, IJCNN 2015, Killarney, Ireland*, July 12-17, 2015, pages 1–8, 2015.

4.1 Introduction

L'algorithme des K-moyennes est l'un des algorithmes de clustering le plus répandu dans la littérature. Il doit sa popularité essentiellement à sa rapidité et à sa simplicité [61]. Cet algorithme consiste à construire une partition initiale des données de cardinalité K. A partir de cette étape d'initialisation, l'algorithme des K-moyennes cherche à améliorer itérativement le partitionnement en déplaçant les objets d'un groupe à un autre jusqu'à atteindre un critère terminal de stabilité. Pour plus de détail sur cet algorithme, le lecteur pourra se référer à [72, 61].

Afin d'atteindre notre objectif décrit dans la section 2.6.1 du chapitre 2, à savoir "la recherche d'un algorithme permettant de prédire et de décrire d'une manière simultanée", nous avons commencé par définir dans le chapitre 3, un prétraitement supervisé (interprétable) basé sur une estimation des distributions uni-variées conditionnellement aux classes. Ce prétraitement permet d'obtenir une distance dépendante de la classe qui aide l'algorithme des K-moyennes standard à avoir de bonnes performances au sens du clustering prédictif (i.e. le compromis entre la prédiction et la description). La deuxième étape qui suit l'étape de prétraitement est l'initialisation des centres (voir Algorithme 3 de la section 2.6.2 du Chapitre 2). Cette dernière a une grande influence sur les résultats fournis par l'algorithme des K-moyennes. Ce chapitre a donc pour but de discuter l'impact des méthodes d'initialisation supervisées sur la qualité des résultats de l'algorithme des K-moyennes classique au sens du clustering prédictif (ou la classification à base de clustering).

De par sa nature, l'algorithme standard des K-moyennes converge rarement vers un optimum global. La qualité [6] de cet optimum local et le temps requis par l'algorithme pour converger dépendent entre autre du choix des centres initiaux. Une mauvaise initialisation peut produire une solution (optimum local, «Figure 4.1 a») qui peut être très différente (ou loin) de la solution optimale «Figure 4.1 b». A titre d'exemple, la figure 4.1 présente une configuration dans laquelle l'algorithme des K-moyennes (K est fixé à 3) converge vers un minimum local qui ne reflète pas la vraie structure interne des données.

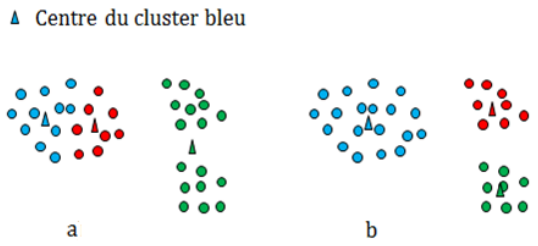


FIGURE 4.1 – Exemple de configuration dans lequel la solution générée par les K-moyennes (a) est très différente de la solution optimale (b)

Le choix d'une méthode d'initialisation appropriée est alors une étape très importante. L'utilisation d'une mauvaise méthode d'initialisation peut générer plusieurs effets indésirables tels que : *i*) des clusters vides, *ii*) une convergence plus lente, et *iii*) une grande probabilité de tomber sur un mauvais optimum local et donc la nécessité d'exécuter l'algorithme plusieurs fois [30]. Les centres choisis au départ doivent donc fournir une bonne couverture [7] de l'espace de données.

6. Une forte similarité à l'intérieur de chaque groupe et une forte dissimilarité entre les membres de différents groupes

7. Dans le but de minimiser la MSE, les centres initiaux doivent appartenir aux régions denses de l'espace d'entrée représentant des zones d'intérêt.

Ceci permet à l'algorithme d'obtenir un bon résultat sans avoir à l'exécuter de nombreuses fois, voire même en une seule fois si la méthode est déterministe.

Dans le cadre du clustering prédictif ou plus précisément des K-moyennes prédictives, le choix d'une méthode d'initialisation est également une problématique à résoudre. La seule différence réside dans le fait que l'utilisateur dispose dans ce cas d'une information supplémentaire à savoir : l'appartenance des instances à une des classes à prédire. Une solution optimale au sens des K-moyennes prédictives est donc celle qui réalise un bon compromis entre la compacité des groupes et leur pureté en termes de classes (pour plus de détails, voir Chapitre 2 Section 2.6).

Il est donc naturel de se demander si l'utilisation d'une méthode d'initialisation supervisée peut aider l'algorithme des K-moyennes standard à obtenir de bons résultats au sens du clustering prédictif. Une bonne méthode d'initialisation supervisée devrait réussir à capter les points appartenant aux régions denses et pures en termes de classes. L'obtention de ces points candidats peut faciliter la tâche de l'algorithme des K-moyennes standard puisqu'il va débuter avec une "bonne" partition initiale. Le but du travail présenté dans ce chapitre est donc de répondre à la question :

"À quel point une méthode d'initialisation supervisée pourrait aider l'algorithme des K-moyennes standard à former des groupes compacts, homogènes et purs au sens du clustering prédictif ? "

Pour répondre à cette question, nous allons tout d'abord présenté dans la section 4.2 un bref état de l'art des méthodes d'initialisation supervisées et non supervisées existantes dans la littérature. Ensuite, nous proposons dans la section 4.3 trois méthodes d'initialisation supervisées. Les résultats générés par l'algorithme des K-moyennes en utilisant ces méthodes supervisées et les méthodes issues de la littérature seront présentés et discutés dans les deux sections 4.5 et 4.6. Finalement, une conclusion générale contenant la réponse à la question posée ci-dessus sera présentée dans la section 4.7.

4.2 État de l'art

La recherche d'une méthode appropriée pour initialiser les centres des K-moyennes est un domaine de recherche très actif. Jusqu'à présent, une grande variété de méthodes existe dans la littérature. Dans cette section, nous allons présenter une brève description des méthodes les plus répandues. Ces méthodes peuvent être classées selon leur complexité en N (nombre d'instances du jeu de données) : linéaire, log-linéaire, quadratique, etc. Des articles de revue plus détaillés existent dans la littérature, le lecteur souhaitant une description plus détaillée de ces méthodes pourra se référer à [29], [28], [31], [66].

4.2.1 Les méthodes ayant une complexité linéaire en N

Les méthodes d'initialisation (quelle que soit leur complexité algorithmique), peuvent être catégorisées en deux grandes familles : déterministes et non déterministes. La première famille regroupe les méthodes capables de fournir un résultat unique quel que soit le nombre d'exécution de l'algorithme. Les résultats générés par l'algorithme des K-moyennes sont dans ce cas reproductibles. La deuxième famille, quant-à-elle, regroupe les méthodes basées sur l'aléatoire. Elles fournissent à chaque fois une solution différente de la précédente.

A. Les méthodes non déterministes

La première méthode d'initialisation proposée dans la littérature est celle de ***Forgy*** en 1965 [49]. Cette méthode consiste à choisir aléatoirement les K centres initiaux parmi les N instances de l'ensemble de données. La motivation derrière cette proposition réside dans le fait que la sélection aléatoire est susceptible de capter des points qui sont de bons candidats (par exemple, les points appartenant aux régions denses). Cependant, des points aberrants ou des points proches les uns des autres peuvent être choisis comme centres initiaux ce qui est clairement sous-optimal.

Bradley et Fayyad [24] ont proposé une méthode d'initialisation qui commence par une division aléatoire du jeu de données en B groupes. Ensuite, l'algorithme des K-moyennes est appliqué sur chacun de ces groupes en sélectionnant à chaque fois les centres aléatoirement. Les $B \times K$ centres obtenus sont alors regroupés et considérés comme une entrée de l'algorithme des K-moyennes. Ce dernier est ensuite exécuté B fois et est initialisé à chaque fois par des centres différents. Finalement, les centres qui forment la partition ayant une valeur minimale de MSE (*i.e.,* l'erreur quadratique moyenne) sont considérés comme les centres finaux. Le principal avantage de cette méthode est qu'elle augmente l'efficacité du résultat par le fait que les centres initiaux sont obtenus après des exécutions multiples de l'algorithme des K-moyennes. Cependant, l'inconvénient majeur de cette méthode est qu'elle nécessite beaucoup d'effort de calcul.

Sample [76] est une simple méthode d'initialisation qui consiste à appliquer un algorithme de partitionnement sur un échantillon de l'ensemble de données (souvent 10%). Les centres résultant sont alors considérés comme les centres initiaux. L'inconvénient majeur de cette méthode est qu'il se peut que l'échantillon sélectionné ne soit pas vraiment un échantillon représentatif de l'ensemble des données.

La méthode ***MaxiMin*** [52] choisie le premier centre aléatoirement. Puis le *ième* centre c_i ($i \in \{2, 3, \ldots, K\}$) est défini comme étant le point X_t qui vérifie :

$$t = argmax_{j \in \{1,2,\ldots,N\}}(min_{k \in \{1,2,\ldots,i-1\}} \parallel X_j - c_k \parallel_2^2)$$

Ce processus est répété $(K-1)$ fois.

La méthode ***K-means++*** [15] combine les deux méthodes ***Forgy*** et ***MaxiMin***. Cette méthode commence par choisir le premier centre aléatoirement. Ensuite le *i-ème* centre ($i \in \{2, \ldots, K\}$) est choisi de la manière suivante : *i*) calculer pour chaque point X' qui n'est pas un centre, la probabilité $\frac{dist(X')^2}{\sum_{X \in \mathcal{D}} dist(X)^2}$, où $dist(X)$ est la distance entre un point $X \in \mathcal{D}$ et son centroïde le plus proche, *ii*) tirer un centre c_i parmi les X' suivant cette probabilité, et *iii*) répéter les deux étapes *i*) et *ii*) jusqu'à ce que l'on ait placé tous les centres.

B. Les méthodes déterministes

MacQueen [72] a proposé une méthode simple d'initialisation des centres. Cette méthode consiste à prendre les K premiers points du jeu de données comme étant les centres initiaux. L'inconvénient majeur de cette méthode réside dans sa sensibilité envers l'ordre des données. De plus, les centres sélectionnés peuvent être proches les uns des autres.

Ball et Hall [17] proposent, quant à eux, une méthode d'initialisation qui consiste tout d'abord à choisir le centre de gravité de l'ensemble des données comme étant le premier centre. Ensuite, la distance entre ce centre et le premier point du jeu de données est alors calculée. Si cette distance est supérieure à un certain seuil T, alors ce point est choisi comme étant le deuxième centre. Sinon, le point suivant du jeu de données est alors testé. Ce processus est répété jusqu'à atteindre le nombre de clusters désiré. Le point fort de cette méthode est qu'elle

permet à l'utilisateur de contrôler la distance entre les centres de différents clusters. Cependant, ce procédé souffre de quelques inconvénients : *i*) la dépendance de la méthode à l'ordre des points dans le jeu de données, *ii*) la distance entre les centres dépend du seuil T qui doit être connu *a priori*. La complexité algorithmique de cette méthode est $\mathcal{O}(NKd)$.

La méthode de recherche des clusters simples [100] (*ou en anglais* ***Simple Cluster Seeking Method***) est similaire à la méthode proposée par Ball et Hall. La seule différence est dans le choix du premier centre. Cette méthode le considère comme étant le premier point dans le jeu de données. La complexité algorithmique de cette approche est $\mathcal{O}(NKd)$.

PCA-Part [97] utilise une approche de division hiérarchique basée sur l'analyse en composantes principales (ACP) pour déterminer les centres initiaux. La méthode part d'un seul cluster qui contient l'ensemble des données. Ensuite, elle sélectionne successivement le cluster ayant une valeur maximale de SSE (*i.e.,* Sum Squared Error) et le divise en deux sous-clusters à l'aide d'un hyperplan. Ce dernier passe par le centroïde du cluster sélectionné qui est orthogonal au vecteur propre principal de sa matrice de covariance. Cette procédure itérative est répétée $K-1$ fois. Finalement, les centres initiaux sont considérés comme étant les centroïdes des K groupes obtenus. Les différentes étapes de cette méthode sont comme suit :

1. Soit le cluster ayant la plus grande valeur de SSE [8] et μ est le centre de gravité du cluster C. Le premier cluster C_1 est le cluster contenant tous les données et c_1 est son centre de gravité.
2. Soit q la projection du c_i sur le principal vecteur propre $v_i (q = c_i.v_i)$
3. Diviser C_i en deux sous-clusters C_{i1} et C_{i2} en respectant la règle suivante :

$$\forall X_l \in C_i, \textbf{ Si } X_l.v_i \leq q \text{ alors } X_l \in C_{i1}$$
$$\textbf{Sinon } X_l \in C_{i2}$$

4. Répéter les étapes 1.-3.$(K-1)$ fois

Pour déterminer le vecteur propre principal à partir du cluster sélectionné, plusieurs méthodes peuvent être utilisées. A titre d'exemple, on trouve la méthode de la puissance [89] et la méthode Lanczos [22]. La complexité de cette méthode dépend alors de la méthode utilisée. Par exemple, la complexité de ***PCA-Part*** en utilisant la méthode puissance est $\mathcal{O}(Nd^2K)$.

Var-Part [97] est une approximation de la méthode PCA-Part. La seule différence entre les deux méthodes réside dans le choix de l'axe de projection. Dans Var-Part, à chaque itération, la matrice de covariance du cluster à diviser est supposée être diagonale. Dans ce cas, l'hyperplan de séparation est perpendiculaire à l'axe ayant la plus grande variance. Les différentes étapes de cette méthode sont :

- Soit C_1 le cluster contenant l'ensemble de données et c_1 le centre de gravité associé à ce cluster.

1. Sélectionner le cluster ayant la plus grande valeur de SSE.
2. Diviser le cluster C_i sélectionné en deux de la manière suivante :
   — Calculer la variance de chaque variable
   — Trouver la variable qui a une grande variance notée X^p avec $p \in \{1, \ldots, d\}$
   — Soit X_i^p la valeur de la variable X^p pour l'instance i et μ_i^p la moyenne du cluster C_j pour la variable p.
   Diviser le cluster C_j en deux sous-clusters C_{j1} et C_{j2} selon la règle suivante :

$$\textbf{Si } X_i^p \leq \mu_i^p \text{ alors } X_i \in C_{j1}$$
$$\textbf{Sinon } X_i \in C_{j2}$$

8. Soit pour un cluster C, $SSE = \sum_{X_j \in C} \parallel X_j - \mu \parallel_2^2$ avec $\parallel X_j \parallel_2 = (\sum_{l=1}^{d} X_{jl}^2)^{1/2}$

3. Répéter les deux étapes 1. et 2. $(K-1)$ fois.

La méthode ***KKZ*** est une méthode d'initialisation proposée par Katsavounidis et *al.* [64]. L'idée est de se focaliser sur les points les plus éloignés les uns des autres. Ces points sont les plus susceptibles d'appartenir à des clusters différents. La démarche suivie par la méthode KKZ est comme suit :

1. Choisir le point ayant la maximale norme ℓ_2 comme le premier centre.
2. Chaque centre c_j $(j \in \{2, \ldots, K\})$ est défini de la manière suivante : pour chaque point X_i du jeu de données qui n'est pas un centre, calculer la distance *dist* entre ce point et le centre le plus proche. Ensuite, le point ayant la plus grande valeur *dist* est choisi comme le centre c_j.

La méthode KKZ est connue par sa simplicité. Cependant, cette méthode est sensible à l'existence des outliers dans les données. La complexité algorithmique de cet algorithme est $\mathcal{O}(NKd)$.

4.2.2 Les méthodes ayant une complexité log-linéaire en N

La méthode de ***Hartigan*** [56] commence par trier les points du jeu de données en fonction de leurs distances au centre de gravité de l'ensemble de données. Le i*ème* centre $(i \in \{1, \ldots, K\})$, est défini comme étant le $(1 + (i-1)N/K)$*ème* point. Cette méthode est une amélioration de la méthode proposée par MacQueen [72]. Elle est invariante à l'ordre des données et semble générer des centres bien séparés.

La méthode ***ROBIN*** (ROBust INitialisation) [7] utilise le facteur local des outliers (LOF) [26] pour éviter la sélection des outliers comme des centres. A l'itération $i \in \{2, \ldots, K\}$, la méthode ROBIN trie les points du jeu de données dans un ordre décroissant en fonction de leur distance minimale aux centres déjà sélectionnés. Suivant ce tri ***ROBIN*** sélectionne le premier point rencontré ayant une valeur de LOF proche de 1 comme le i*ème* centre. Ce procédé est répété jusqu'à atteindre le nombre de clusters désiré.

Al-Daoud [5] trie tout d'abord les points en fonction de l'attribut ayant la plus grande variance et les partitionne ensuite en K groupes de même dimension. Les centres initiaux sont alors définis comme étant les points qui correspondent aux médianes de ces groupes. Cette méthode ne prend en compte qu'un seul attribut. Elle est susceptible d'être efficace seulement pour les données dont la variabilité est principalement contenue sur une seule dimension.

4.2.3 Les méthodes ayant une complexité quadratique en N

Kaufman et Rousseeuw [65] prennent le premier centre comme étant le centre de gravité de l'ensemble des données. Le i*ème* $i \in \{2, \ldots, K\}$ centre est choisi comme étant le point qui réduit le plus la valeur de SSE.

K. A. Abdul Nazeer *et al.* [78] proposent une méthode d'initialisation qui commence par calculer les distances entre chaque point de l'ensemble de données $\mathcal{D}$ et tous les autres points. Ensuite, le couple de points ayant la distance minimale est alors sélectionné et retiré du jeu de données en formant ainsi un nouveau sous-ensemble A_1. A chaque fois, le point le plus proche des points de ce sous-ensemble est alors ajouté à A_1 et retiré du jeu de données $\mathcal{D}$. Ce processus est répété jusqu'à ce que le nombre d'éléments dans A_1 atteint un certain seuil. De la même façon les sous-ensembles $A_2, \ldots, A_K$ sont construit à partir des points restant dans l'ensemble $\mathcal{D}$. Finalement, les centres initiaux sont obtenus en calculant les centres de gravité de chaque sous-ensemble $A_1, \ldots, A_K$.

Le tableau 4.1 présente quelques points résumant les méthodes d'initialisation qui seront utilisées dans les deux sections 4.5 et 4.6 lors des expérimentations.

Nom	**Type**	**Avantages et limites**	**Complexité**
Foggy (Random)	Non supervisée	- Simple et rapide - La probabilité de choisir des points proches ou des points aberrants est grande	$\mathcal{O}(K)$
Sample	Non supervisée	- Simple et rapide - L'échantillon sélectionné peut ne pas être un échantillon représentatif de l'ensemble de données	$\mathcal{O}(N'Kdt)$ N' est le nombre d'instances dans l'échantillon
Kmeans++	Non supervisée	- Génère des centres bien séparés les uns des autres - Deux sous-groupes proches peuvent partager le même centre même s'ils ont des classes différentes.	$\mathcal{O}(NKd)$
Var-Part	Non supervisée	- Déterministe, simple et rapide : sa complexité algorithmique est égale à la complexité des K-moyennes pour une seule itération - Elle se focalise sur la diminution de la MSE au sein de chaque cluster lors de la division sans prendre en considération qu'un groupe compact peut contenir des instances de classes différentes. Ceci dégrade la qualité au sens du clustering prédictif	$\mathcal{O}(NKd)$

TABLE 4.1 – Propriétés de quelques méthodes d'initialisation utilisées dans la partie d'expérimentation

4.3 Contribution

L'intérêt de l'utilisation d'une méthode supervisée pour initialiser les centres dans le cadre des K-moyennes prédictives peut être vu clairement dans le cas de déséquilibre des classes à prédire. Par exemple, dans l'exemple illustratif de la figure 4.2, si on tire aléatoirement les centres initiaux alors la probabilité de choisir plus d'un centre dans la classe majoritaire (*e.g.,* la classe rouge dans la figure 4.2) et de ne choisir aucun centre dans la classe minoritaire (*e.g.,* la classe magenta dans la figure 4.2) est élevée. Par conséquent, une détérioration au niveau de la pureté, en termes de classe à prédire, des clusters serait introduite. De ce fait, l'idée d'intégrer l'information contenue dans la variable cible dans le processus d'initialisation peut s'avérer nécessaire vis à vis du compromis entre description des données et prédiction des classes.

Cette section est consacrée à la présentation de trois nouvelles méthodes d'initialisation (K-means++R, Rocchio-and-split et S-Bisecting) qui se servent de l'information cible pour sélec-

tionner les centres initiaux.

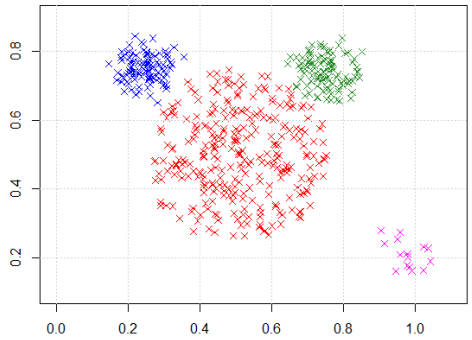


FIGURE 4.2 – Jeu de données Mouse [9][1]

4.3.1 K-means++R [70]

La première méthode proposée dans cette section est nommée K-means++R (ou K++**R**) [70]. Cette méthode suit un mécanisme dit : "d'*exploitation* et d'*exploration*". En effet, elle exploite dans un premier temps l'information donnée par la variable cible. Puis, elle explore dans un deuxième temps la densité de la distribution des données. Dans la première phase dite d'exploitation, K++R se sert de l'information donnée par la variable cible pour fournir les J premiers centres initiaux. Ces derniers sont obtenus de telle sorte que chacune des J classes contient un seul centre (*i.e.,* l'approche Rocchio [73]). Cette phase d'exploitation permet de résoudre la problématique citée ci-dessus : la probabilité de ne choisir aucun centre dans les classes minoritaires est dans ce cas nulle (voir Figure 4.2). Chaque centre est défini comme étant le centre de gravité des instances de la même classe. Par exemple, le $j ème$ centre associé à la classe C_j ($j \in \{1, \ldots, J\}$) est donnée par l'équation (4.1).

$$c_j = \frac{1}{N_j} \sum_{i \in C_j} X_i \tag{4.1}$$

La phase d'exploration, quant à elle, est consacrée à la sélection des $K - J$ centres initiaux restants. Cette phase cherche à explorer la densité de la distribution des données afin de sélectionner les points candidats appartenant aux régions denses. Cette phase est donc une étape non supervisée qui consiste à sélectionner à chaque itération le point le plus éloigné des centres déjà choisis. Il s'agit de chercher à assurer plus de diversité en explorant l'ensemble de données. La méthode K++R utilise à ce stade l'algorithme d'initialisation communément utilisé à savoir K-means++ [15]. Cet algorithme est débuté par les J centres trouvés dans la première phase (*i.e.*, la phase d'exploitation de la variable cible). Il est à noter que dans notre cadre d'étude, on ne s'intéresse pas au cas où le nombre de clusters K est inférieur au nombre de classes J puisqu'on cherche à découvrir la distribution sous-jacente de chaque classe à prédire.

La figure 4.3 présente un exemple illustratif de l'emplacement des centres initiaux dans l'espace des données (Figure 4.2) en utilisant la méthode K++R pour différent nombres de clusters $K \in \{4, 5, 6, 7\}$. Cette figure montre que : *i*) lorsque $K = J = 4$ (partie gauche de la figure), chaque classe (majoritaire et minoritaire) contient un centre, ce qui résout la problématique de

9. Le jeu de données Mouse est caractérisé par la présence de 500 instances, 2 variables descriptives et une variable à prédire contenant 4 classes.

déséquilibre des classes, *ii*) plus le nombre de clusters augmente ($K \in \{5, 6, 7\}$), plus la méthode cherche à trouver les points les plus éloignés les uns des autres en explorant l'ensemble des données.

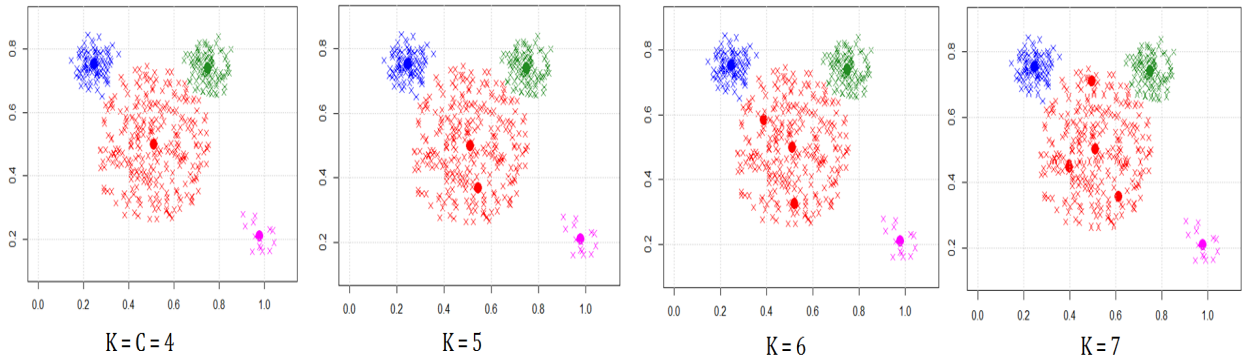


FIGURE 4.3 – l'emplacement des centres initiaux dans l'espace d'entrée en utilisant la méthode K-means++R pour $K \in \{4, 5, 6, 7\}$

Les différentes étapes de l'approche K++R sont présentées par des lignes de code présentées dans l'algorithme 4.

Entrée : $\mathcal{D} = \{(X_i, Y_i)\}_{i=1}^{N}$ *: Le jeu de données d'apprentissage.*
K : Le nombre de clusters.
J : Le nombre de classes.

- Calculer le centre de gravité de chaque classe (équation 4.1)
Si $(J = K)$ **Alors**
  Sortie : les J centres de gravité
Fin Si
Si $(K > J)$ **Alors**
  On pose *dist* la distance entre un point X et son centroïde le plus proche
  1. Calculer pour chaque point X' qui n'est pas un centre la probabilité $\frac{dist(X')^2}{\sum_{X \in \mathcal{D}} dist(X)^2}$
  2. Tirer un centre c_i parmi les X' suivant cette probabilité
  3. Répéter 2. et 3. Jusqu'à ce que l'on ait placé tous les centres

  Sortie : Les K centres initiaux
Fin Si

Algorithme 4 – K-means++R

Dans le cadre du clustering standard, il est clair que cette méthode d'initialisation n'est pas une méthode appropriée pour former de bons clusters. En effet, le fait de dédier à chaque classe un centre peut générer une détérioration au niveau de la qualité des partitions générées en termes d'erreur de construction MSE (*ou* erreur quadratique moyenne). À titre d'exemple voir la figure 4.2 pour le cas où le nombre de clusters est égal au nombre de classes ($K = J = 4$). Cependant, dans le cadre du clustering prédictif, cette méthode est intéressante puisqu'on cherche plutôt à réaliser un compromis entre la compacité (*e.g.,* en termes de MSE) et la pureté en termes de

taux de bonnes classifications.

A note connaissance, il n'existe pas de méthode dans la littérature qui utilise une étape d'initialisation supervisée. Cependant, certains travaux dans le domaine semi-supervisé, intègrent l'information supplémentaire donnée par la variable cible pour initialiser les centres. La méthode d'initialisation proposée dans [19] dans le contexte semi-supervisé initialise de la même façon les centres initiaux que la méthode K++R. La principale différence réside dans l'étiquetage de la partition finale. Dans notre proposition, après la convergence de l'algorithme, on attribue à chaque cluster, la classe majoritaire correspondant aux instances qui le forme (i.e., le vote majoritaire). Par conséquent, l'étiquetage initial associé à la partition initiale change au cours du processus. Par contre dans [19], l'étiquetage des clusters reste inchangé : l'étiquetage initial et final est le même.

4.3.2 Méthodes basées sur la variance : Rocchio-And-Split et S-Bisecting [11, 60]

Lorsque le nombre de clusters (K) est supérieur au nombre de classes (J), la méthode K++R est partiellement supervisée. Les $K - J$ centres initiaux restants sont sélectionnés d'une manière non supervisée à l'aide de l'algorithme $(K - J)$Means++.Il est donc naturel de se demander si l'intégration de l'information contenue dans la variable cible lors de la sélection de ces $K - J$ centres initiaux restants pourrait garantir une meilleure performance que celle obtenue par la méthode K++R. Pour être en mesure de répondre à cette question, nous allons dans ce qui suit proposer deux nouvelles méthodes d'initialisation supervisées.

Ces deux méthodes d'initialisation sont appelées "Rocchio-And-Split" (RS) et "S-Bisecting" (SB). Dans le cas où le nombre de clusters (K) est égal au nombre de classes (J), ces deux méthodes fonctionnent de la même manière que K++R : elles dédient un seul centre à chaque classe. Chaque centre est définit comme étant le centre de gravité des instances de même classe (*i.e.,* la méthode Rocchio [73]). Dans le cas contraire, *i.e.,* lorsque le nombre de clusters est supérieur au nombre de classes, ces deux méthodes suivent une division hiérarchique descendante. Elles partent des J groupes où chaque groupe représente une classe. Ensuite, à chaque itération, le groupe qui vérifie un critère déterminé est alors divisé en deux. Ce processus est répété jusqu'à ce que le nombre de groupes formés soit égal au nombre de centres désiré. Au final, les centres initiaux sont obtenus en calculant les centres de gravité de chacun de ces groupes résultants. Les deux points clefs à déterminer sont alors : *comment sélectionner le groupe candidat à diviser ?*, et *comment le diviser ?*

- **Comment choisir le groupe candidat à diviser** ?

Dans la littérature, il existe plusieurs façons pour sélectionner le groupe candidat. Cette sélection dépend essentiellement du critère choisi et bien entendu du résultat attendu. Par exemple, on peut choisir de diviser le cluster i ayant la plus grande taille (*i.e.,* $i = argmin_k(1/n_k)$). Cette condition permet de produire des clusters de tailles équilibrées. On peut également choisir le groupe candidat suivant la similarité moyenne ou bien la cohésion [44]. Ces deux conditions permettent de produire des groupes compacts en se déplaçant d'un niveau de la hiérarchie à un autre (du haut vers le bas). Au lieu de se focaliser seulement sur les caractéristiques de chaque groupe, on peut également se baser sur la fonction objectif du clustering pour choisir le groupe candidat. Il s'agit de choisir le groupe k telle que la division de celui-ci conduit à une faible augmentation de la fonction objectif globale [44].

Dans notre cadre d'étude, le clustering prédictif cherche à «*discerner des groupes compacts,*

homogènes et purs en termes de classe». Les deux points essentiels qu'on peut retirer de cette définition sont la pureté et la compacité des clusters appris. La pureté peut être assurée en se basant sur le principe de la décomposition des classes (*e.g.,* [101]). C'est-à-dire, traiter chaque classe individuellement. En ce qui concerne, la compacité, elle peut être assurée en se basant sur les caractéristiques de chaque groupe. Il s'agit de chercher à diminuer la dispersion des données dans chaque groupe. En combinant les deux points, le but sera alors de diviser à chaque itération le groupe k_j (le groupe k ayant comme classe j) le plus dispersé. Ce processus est répété jusqu'à atteindre le nombre de groupes désiré. Dans notre proposition, on choisit de mesurer la dispersion par l'inertie intra-cluster.

- **Comment diviser le groupe candidat** ?

Après avoir sélectionné le groupe candidat, on cherche à le diviser en deux. Il est à noter que les deux méthodes proposées dans cette section (RS et SB) différent uniquement au niveau de la méthode de division du groupe candidat. Dans le reste de cette section, nous allons tout d'abord présenté l'approche Rocchio-And-Split tout en décrivant ses avantages et ses limites. Ensuite, nous allons présenter l'approche S-Bisecting.

A. L'approche "Rocchio-And-Split" (RS)

La méthode RS cherche à identifier les régions denses dans la classe ayant une variance intra-classe maximale. Pour ce faire, la méthode commence par sélectionner le groupe ayant une variance intra-classe élevée [10]. Ce groupe est considéré comme étant le groupe candidat à diviser. Pour le diviser, RS commence par sélectionner l'instance la plus éloignée du centre de gravité de ce groupe, notée $X_{i_{max}}$. Notons d_1 cette distance maximale et d_2 la distance entre $X_{i_{max}}$ et chaque instance du groupe à diviser. Ensuite, toutes les instances ayant une distance d_2 plus petite que la distance d_1 sont regroupées ensemble. Cela correspond tout simplement à diviser le cercle de rayon d_1 en deux. La figure 4.4 présente un exemple illustratif de la démarche de l'approche RS. Les différentes étapes de l'approche SB sont présentées par des lignes de code de l'algorithme 5.

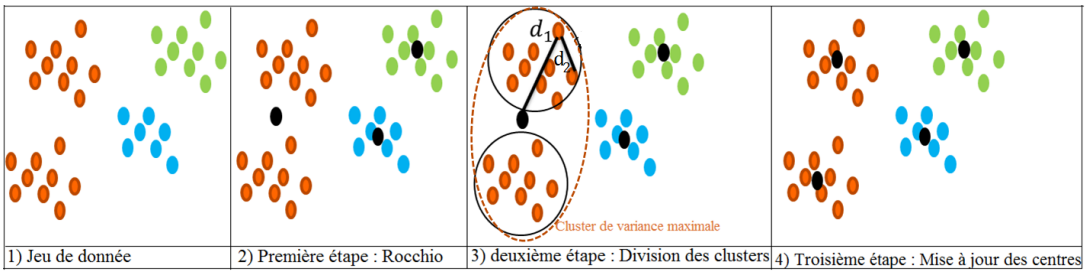


FIGURE 4.4 – Le fonctionnement de la méthode Rocchio-And-Split pour le cas où $K = 4$

10. Dans cette thèse, la variance intra-cluster est calculée à l'aide de l'inertie intra normalisée, présentée comme suit :

$$Intra(k) = \frac{1}{N_k} \sum_{X_i \in C_k} \| X_i - \mu_k \|^2$$

avec N_k est le nombre d'instances dans le cluster k ayant μ_k comme centre de gravité.

Entrée : $\mathcal{D} = \{(X_i, Y_i)\}_{i=1}^{N}$ *: Le jeu de données d'apprentissage.*
K : Le nombre de clusters.
J : Le nombre de classes.

- Calculer le centre de gravité de chaque classe
Si $(J = K)$ **Alors**
  Sortie : les J centres de gravité
Fin Si
Si $(K > J)$ **Alors**
  Tant que (*le nombre de clusters n'est pas atteint*) **faire**
    - Calculer la dispersion dans chaque cluster en sens de l'inertie intra
    - Diviser le cluster le plus dispersé C_k en deux sous-clusters C_{k1} et C_{k2} de la manière suivante : Sélectionner l'instance $X_{i_{max}}$ de C_k telle que $i_{max} = argmax_{\{i \in C_k\}} dist(X_i, \mu_k)$ et $d_1 = dist(X_{i_{max}}, \mu_k)$ avec μ_k est le centre de gravité du cluster k
    Pour (j=1 ; j $\leq N_k$; j++) **faire**
      [N_k est le nombre d'instances dans C_k]
      Si $(d_1 \geq dist(X_{i_{max}}, X_j))$ **Alors**
        $X_j \in C_{k1}$
      Sinon
        $X_j \in C_{k2}$
      Fin Si
    Fin Pour
    - Supprimer le centre du groupe sélectionné et calculer les centres de gravité des deux clusters C_{k1} et C_{k2}
  Fait
  Sortie : Les K centres initiaux
Fin Si

Algorithme 5 – Rochio-And-Split

Avantages : L'un des points fort de l'approche RS est qu'elle est une approche déterministe. Cet avantage permet bien entendu de réduire le temps d'exécution. En effet, il n'est pas nécessaire d'exécuter l'algorithme des K-moyennes plusieurs fois afin de choisir le meilleur résultat (comme dans le cas des approches basées sur l'aléatoire). La complexité de cette approche est linéaire en N : $\mathcal{O}(CN_jd + KN_kd(K - J))$. De plus, la technique suivie pour diviser les groupes dispersés a pour but de capter les points appartenant aux régions denses et pures en termes de classe. L'approche RS est plus adaptée pour les données sphériques.

Limites : L'inconvénient majeur de cette approche est qu'elle est sensible à la présence des bruits, ou "outliers", en raison de l'utilisation de la distance maximale lors de la division. Néanmoins, cet inconvénient peut être atténué en utilisant par exemple un bon prétraitement des données (*e.g,* Conditional Info décrit dans la discussion de la Section 3.2.3 du Chapitre 3). Pour remédier à ce problème, l'approche S-Bisecting est donc proposée.

B. L'approche "S-Bisecting" (SB)

Pour l'approche S-Bisecting, la division du cluster le plus dispersé est réalisée en appliquant un algorithme des K-moyennes. Le nombre de clusters K est fixé ici à 2. La méthode d'initialisation utilisée au cours de la division est alors nécessairement une méthode non supervisée (car le cluster à diviser est pur). Pour S-Bisecting, on a choisi d'utiliser l'algorithme d'initialisation K-means++

(avec $K = 2$) [11] Les différentes étapes de l'approche SB sont présentées par des lignes de code de l'algorithme 6.

Entrée : $\mathcal{D} = \{(X_i, Y_i)\}_{i=1}^{N}$ *: Le jeu de données d'apprentissage.*
K : Le nombre de clusters.
J : Le nombre de classes.

- Calculer le centre de gravité de chaque classe
Si $(J = K)$ **Alors**
  Sortie : les J centres de gravité
Fin Si
Si $(K > J)$ **Alors**
  Tant que (*le nombre de clusters n'est pas atteint*) **faire**
    - Calculer la dispersion dans chaque cluster en sens de l'inertie intra
    - Diviser le cluster le plus dispersé C_k en deux sous-clusters C_{k1} et C_{k2} de la manière suivante :
      1. Initialiser les centres à l'aide de 2-means++
      2. Appliquer l'algorithme 2-moyennes (initialisation K-means++) R fois
      3. Fixer les centres de gravité des deux clusters C_{k1} et C_{k2} à l'aide du résultat du meilleur des R 2-moyennes au sens de la SSE

  Fait
  Sortie : Les K centres initiaux
Fin Si

Algorithme 6 – S-Bisecting

Avantages : L'approche SB est une approche non déterministe. Elle contient une partie d'aléas lors de la division du cluster le plus dispersé. La complexité de la méthode SB est de l'ordre de $\mathcal{O}(R(CN_jd + 2N_kd(K - J)t))$ (t est le nombre d'itérations du 2-means et R est le nombre d'exécutions de l'algorithme (ou de réplicates)) qui est linéaire en N. Le principal avantage de cette approche est d'augmenter l'efficacité du résultat grâce au fait que les centres initiaux sont obtenus après des exécutions multiples de l'algorithme des K-moyennes ($K = 2$).

Limite : L'inconvénient majeur de cette méthode est qu'elle nécessite beaucoup d'efforts de calcul.

À notre connaissance, il n'existe pas dans la littérature de méthode d'initialisation supervisée pour les K-moyennes. Cependant, une méthode d'initialisation supervisée peut être construite en se basant sur le principe de la décomposition des classes (*e.g.,* [101]). Cette dernière consiste à attribuer un nombre égal de centres pour toutes les classes en utilisant l'algorithme K-means++. Le nombre de clusters doit être donc un multiplicateur du nombre de classes ($K = \beta \times J$). Par exemple, dans le cas où $K = (\beta \times J) + \alpha$ avec $(1 \leq \alpha < J)$, le nombre de clusters considéré restera $K = \beta \times J$. La méthode S-Bisecting (SB) proposée est plus parcimonieuse par rapport à la méthode de la décomposition de classes.

Si on revient maintenant à la question posée au début de ce chapitre, à savoir "*les méthodes d'initialisation supervisées peuvent-elles aider l'algorithme des K-moyennes standard à fournir de bons résultats au sens du clustering prédictif ?*". Pour vérifier sa validité, nous allons comparer les

11. Le choix de la méthode K-means++ n'est pas une obligation. D'autres méthodes d'initialisation peuvent également être utilisées (e.g., Variance Partitionning).

performances de l'algorithme des K-moyennes, précédé par différentes méthodes d'initialisation (supervisées ou non supervisées), en termes de compacité et de pureté. Le protocole expérimental suivi dans cette étude est présenté dans la section 4.4. Cette étude expérimentale est divisée en deux parties principales. La première partie est dédiée au cas où le nombre de clusters K est égal au nombre de classes J (Section4.5). Dans ce cas, on suppose que la variable à prédire ne dispose pas d'une structure interne à découvrir (*i.e.,* chaque classe à prédire est très compacte). Le problème est alors limité dans ce cas, au problème de la classification supervisée : on cherche à savoir si l'algorithme des K-moyennes, précédé par une étape d'initialisation supervisée, a la capacité de bien prédire la classe des nouvelles instances. La deuxième partie, quant à elle, est dédiée au cas où le nombre de clusters est supérieur au nombre de classes (Section 4.6). Dans ce cas, le problème devient plus complexe : on suppose que chaque classe (ou quelques-unes) a une structure qui la caractérise. Il s'agit donc de tester si l'algorithme des K-moyennes précédé par une méthode d'initialisation supervisée a la capacité de découvrir la structure interne de la variable cible.

4.4 Protocole expérimental

- Les méthodes d'initialisation : L'algorithme des K-moyennes doit sa popularité à l'une de ses propriétés, à savoir, sa rapidité : sa complexité est linéaire de l'ordre $\mathcal{O}(NKdt)$ (N : nombre d'instances, K : nombre de clusters, d : nombre de variables explicatives, t : nombre d'itérations). Pour préserver cet avantage, on s'intéresse aux méthodes d'initialisation ayant également une complexité linéaire en N [29]. Le tableau 4.2 présente l'ensemble des méthodes d'initialisation supervisées et non supervisées utilisé dans cette étude expérimentale.

Les méthodes non supervisées	**Les méthodes supervisées**
Forgy (**Random**)	K-means++R (**K++R**)
Sample (**Sample**)	Rocchio-and-Split (**RS**)
K-means++ (**K++**)	S-Bisecting (**SB**)
MaxiMin non déterministe (**MM(Rand)**)	Décomposition des classes (**CD**)
MaxiMin déterministe (**MM**)	
Var-Part (**Var-Part**)	

TABLE 4.2 – l'ensemble des méthodes d'initialisation utilisé

1. ***Les méthodes non supervisées*** : selon les études comparatives effectuées par Celebi et al. dans [31], *Var-Part* (Variance Partitioning) est l'une des méthodes qui fournit les meilleurs résultats. À côté de cette méthode, on utilise également les méthodes les plus répandues dans la littérature, à savoir : *K-means++* (K++), *MaxiMin* (déterministe (MM) et non déterministe (MM-Rand)), *Sample* et la méthode de *Forgy* (Random). Pour plus de détails sur ces méthodes, voir la section 4.2.
2. ***Les méthodes supervisées*** : En dehors de la méthode présentée dans [19], proche de K++R uniquement dans le cas où $K = J$ et qui diffère dans le processus d'étiquetage, il n'existe pas de méthodes d'initialisation supervisées. Nous décidons alors de prendre dans cette étude les méthodes proposées dans la section 4.3 (voir aussi la deuxième colonne du tableau 4.2). À côté de ces méthodes, nous avons ajouté dans la deuxième partie expérimentale, une autre méthode basée sur le principe de la décomposition des classes (CD). Pour plus de détails, voir la Section 4.3.2 b).

- Le prétraitement : dans cette étude expérimentale, nous avons choisi d'utiliser deux prétraitements. Le premier est un prétraitement supervisé nommé : "*Conditional Info*". Ce choix

fait suite à l'étude menée dans [10] et dans le chapitre 2 de ce mémoire où l'on a pu montrer que l'utilisation de ce prétraitement aide l'algorithme des K-moyennes standard à atteindre une bonne performance prédictive (le processus de prédiction est expliqué ci-dessous). Le deuxième prétraitement, quant à lui, est un prétraitement non supervisé. Parmi les prétraitements non supervisés, nous avons décidé d'utiliser celui qui fournit de bons résultats au sens du clustering prédictif (voir chapitre 3 section 3.4). Il s'agit de "*Rank Normalization*" (voir Chapitre 3 Section 3.3) pour les variables continues et de "*Basic-grouping*" (voir Chapitre 3 Section 3.3) pour les variables catégorielles. On prend ici deux types de prétraitements (supervisé et non supervisé) afin de savoir si la réponse à la question posée dans ce chapitre n'est pas dépendante du prétraitement utilisé.

- Nombre de clusters : Il varie de J jusqu'à K_i pour un prétraitement i utilisé. Pour chaque jeu de données, K_i a été déterminé au préalable de manière à ce que la partition obtenue, avec $K{=}K_i$ permette d'obtenir un ratio (inertie inter / inertie totale) de 80%. Pour plus de détails sur cette démarche voir l'annexe A. La valeur de K_i ($i \in \{1, 2\}$) pour chacune des jeux de données est indiquée dans le tableau 4.3 où $i = 1$ et $i = 2$ correspondent respectivement au Conditional Info et au Rank Normalization. Il est à noter que dans cette étude, le nombre de clusters K ne doit pas être inférieur à J puisqu'on suppose que la variable cible a une structure interne à découvrir.

ID	**Données**	M_n	M_c	N	J	K_1	K_2	J_{maj}
1	Iris	4	0	150	3	4	4	33
2	Hepatitis	6	13	155	2	9	66	79
3	Wine	13	0	178	3	12	38	40
4	Glass	10	0	214	6	15	25	36
5	Heart	10	3	270	2	23	90	56
6	Horsecolic	7	20	368	2	6	200	63
7	Soybean	0	35	376	19	20	49	14
8	Breast	9	0	683	2	4	12	65
9	Australian	14	0	690	2	22	210	56
10	Pima	8	0	768	2	10	74	65
11	Vehicle	18	0	846	4	11	24	26
12	Tictactoe	0	9	958	2	12	64	65
13	LED	7	0	1000	10	17	19	11
14	German	24	0	1000	2	7	363	70
15	Segmentation	19	0	2310	7	23	64	14
16	Abalone	7	1	4177	28	29	29	16
17	Waveform	21	0	5000	3	86	64	34
18	Adult	7	8	48842	2	12	64	76
19	Mushroom	0	22	8416	2	8	64	53
20	PenDigits	16	0	110992	10	64	64	10
21	Phoneme	256	0	2254	5	64	64	26

TABLE 4.3 – Liste des jeux de données utilisés - (J_{maj} représente $\approx$ pourcentage classe majoritaire)

- Les jeux de données : Pour évaluer et comparer les différentes méthodes d'initialisation en fonction de leur capacité à aider l'algorithme standard des K-moyennes à atteindre l'objectif du clustering prédictif, nous allons effectuer des tests sur différents jeux de données de l'UCI [1]. Ces jeux de données ont été choisis afin d'avoir des bases de données diverses en termes de nombre de classes J, de variables (continues M_n et/ou catégorielles M_c) et d'instances N (voir Tableau 4.3).

- Les critères d'évaluation : À notre connaissance, il n'existe pas dans la littérature un

critère global intégrant une partie interne qui mesure la compacité des clusters et une partie externe qui mesure la pureté des clusters en termes de classes. Pour cette raison, nous allons utiliser deux types de critères d'évaluation : supervisé (critère externe) et non supervisé (critère interne). Pour le critère supervisé, nous avons choisi un critère d'évaluation communément utilisé à savoir *Adjusted Rand Index* (ARI) [57]. En ce qui concerne le critère non supervisé, on a choisi d'utiliser *l'erreur quadratique moyenne* (MSE). La formule mathématique utilisée pour la MSE est donnée comme suit :

$$MSE = \frac{1}{N}\frac{1}{Z}\frac{1}{K}\sum_{i=1}^{N}\sum_{z=1}^{Z}\sum_{t=1}^{K}(XR_i^z - k_t^z)^2 \tag{4.2}$$

— N est le nombre d'instances dans l'ensemble de données.
— Z est le nombre de variable après le processus de prétraitement. Par exemple, pour Conditional Info, $Z = (M_n + M_c) \times J$.
— XR est le nouveau vecteur d'instance obtenu après le processus de prétraitement utilisé. Par exemple, pour Conditional Info, ce nouveau vecteur XR est de dimension $(M_n + M_c) \times J$.

- Nombre de réplicates (ou d'exécutions) R : De par sa nature, l'algorithme des K-moyennes standard converge rarement vers un optimum global. En utilisant une méthode d'initialisation basée sur l'aléatoire et en n'exécutant l'algorithme qu'une seule fois, ce dernier est susceptible de tomber sur un mauvais minima local. Afin d'éviter ce risque, ce dernier doit alors être exécuté plusieurs fois tout en changeant les conditions initiales. Dans cette étude, on prend $R \in \{1, 10, 100\}$.

- Le choix de la bonne partition : Lorsque l'algorithme des K-moyennes est exécuté R ($R > 1$) fois, la partition optimale (parmi les R partitions) est alors choisie suivant le critère MSE (voir équation 4.2 de la section 4.4).

- Affectation des classes aux clusters : À la fin du processus d'apprentissage, chaque groupe appris prend j comme étiquette si la majorité des exemples qui le forme sont de la classe j (*i.e.,* l'utilisation du vote majoritaire).

- La prédiction : à la présence d'une nouvelle instance, l'algorithme lui affecte l'étiquette du cluster qui lui est plus proche [12] (*i.e.,* l'utilisation du 1 plus proche voisin).

- Folds cross validation : pour pouvoir comparer les résultats obtenus, un 2×5 folds cross validation a été effectuée sur chaque jeu de données. De ce fait, les résultats sont présentés comme une moyenne de 10 tests.

4.5 Cas où le nombre de clusters (K) est égal au nombre de classes (J)

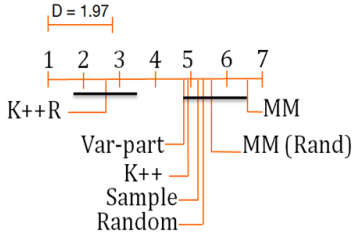
Dans le cas où le nombre de clusters est égal au nombre de classes ($K = J$), on suppose que la variable cible ne dispose pas d'une structure interne à découvrir (*i.e.,* chaque classe à prédire est compacte). Le problème du clustering prédictif devient tout simplement un problème de classification supervisée. Les groupes appris par l'algorithme doivent assurer une bonne performance en termes de pureté afin de pouvoir prédire correctement par la suite la classe des nouvelles instances. Dans cette étude, nous cherchons à savoir si les méthodes d'initialisation

12. Une instance i est plus proche au cluster C_1 que au cluster C_2 si et seulement $dist(i, g_1) < dist(i, g_2)$ avec g_1 (respectivement g_2) est le centre de gravité du cluster C_1 (respectivement C_2).

supervisées ont un impact sur les performances *'prédictives'* des K-moyennes. Il est à noter que les trois méthodes d'initialisation supervisées (*Rocchio-and-Split*, *S-Bisecting* et *K-means++R*) fonctionnent de la même façon lorsque $K = J$ (*i.e.,* l'utilisation du *Rocchio*). Pour cette raison, nous ne détaillons ci-dessous que les résultats de la méthode *K-means++R* (K++R) parmi ces trois méthodes.

Pour comparer les performances prédictives de plusieurs méthodes d'initialisation sur plusieurs bases de données, nous utilisons le test de Friedman couplé au test post-hoc de Nemenyi [41] pour un seuil de significativité $\alpha = 0.05$. Pour plus de détails sur ces deux tests voir Annexe B Section B.1.

• **Avec le prétraitement supervisé** : la figure 4.5 présente les résultats des comparaisons des performances prédictives en termes d'ARI de l'algorithme des K-moyennes en utilisant à chaque fois "Contidional Info" et l'une des méthodes d'initialisation. Ces résultats sont obtenus dans le cas où l'algorithme des K-moyennes n'est exécuté qu'une seule fois (*i.e.,* $R = 1$). Les méthodes dans la figure à gauche sont classées en ordre décroissant selon leurs performances prédictives en se basant sur la moyenne des rangs : plus le rang moyen de la méthode est proche de 1 meilleure elle est en prédiction. D'après le résultat du test de Friedman, il existe une différence significative entre les 7 méthodes d'initialisation ($p_{value} < 1.336e^{-07} \ll 0.05$) avec une grande préférence pour la méthode K++R. Ce résultat est confirmé par le test de Nemenyi (voir le tableau des p_{values} présenté dans la partie droite de la figure 4.5). Celui-ci partitionne les méthodes en deux groupes distincts $\{K++R\}$ et $\{Random, Var-Part, K++, Sample, MM(Rand), MM\}$ de tel sorte que le premier groupe (K++R) est celui qui fournit de bons résultats en termes de prédiction.



	Sample	Random	K++	Var-Part	MM (Rand)	MM
Random	1.000	-	-	-	-	-
K++	0.98935	0.99537	-	-	-	-
Var-Part	0.96807	0.98284	1.000	-	-	-
MM (Rand)	0.99996	1.000	0.99926	0.99537	-	-
MM	0.58212	0.50912	0.15911	0.10365	0.39190	-
K++R	0.00019	0.00031	0.00440	0.00845	0.00068	1.4e-08

FIGURE 4.5 – Comparaison des méthodes d'initialisation (précédées par le prétraitement "CI") en utilisant le test de Friedman couplé au test post-hoc de Nemenyi (ARI en test) pour R=1

Il faut noter que l'algorithme des K-moyennes converge très rarement vers un optimum global. Les méthodes d'initialisation basées sur l'aléatoire (*e.g.,* MM (Rand), Sample et Forgy) ont donc une grande chance d'échapper aux mauvais minima locaux lorsqu'on exécute l'algorithme de nombreuses fois. Pour cette raison, nous augmentons le nombre de réplicates $R \in \{10, 100\}$. Pour $R = 10$ (Figure 4.6 a)), le test de Friedman montre qu'il existe une différence significative ($p_{value} = 3.203e^{-09} \ll 0.05$) entre les méthodes, tandis que le test de Nemenyi partitionne les méthodes en deux groupes distincts : $\{K++R\}$ et $\{Sample, Random, Var-Part, MM, K++, MM(Rand)\}$. La figure (Figure 4.6 a)) montre que la méthode K++R est la meilleure en termes de prédiction. Pour $R = 100$ (Figure 4.6 b)), les mêmes conclusions peuvent être observées ($p_{value} = 1.526e^{-10}$ suivant le test de Friedman).

• **Avec le prétraitement non supervisé :** En utilisant Rank Normalization (RN) pour les variables continues et Basic Gouping (BGB) pour les variables catégorielles, la figure 4.7 présente les résultats des comparaisons des performances prédictives de l'algorithme des K-moyennes

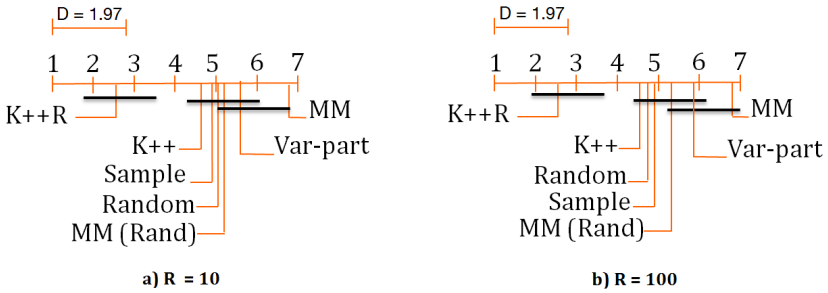


FIGURE 4.6 – Comparaison des méthodes d'initialisation (précédées par le prétraitement "CI") en utilisant le test de Friedman couplé au test post-hoc de Nemenyi (ARI en test) pour $R \in \{10, 100\}$

précédé à chaque fois par une méthode d'initialisation. Les résultats du test Friedman montrent que quel que soit le nombre de réplicates utilisé ($R \in \{1, 10, 100\}$), la méthode K++R reste celle qui fournit les meilleurs résultats en termes d'ARI. Tandis que le test de Nemenyi partitionne à chaque fois les méthodes en deux groupes. En appliquant le test bilatéral (Wilcoxon signé) pour les deux premières méthodes du classement (selon Friedman), on trouve que : 1) pour $R = 1$, $p_{value} = 10^{-4}$, 2) pour $R = 10$, $p_{value} = 0.007$ et 3) pour $R = 100$, $p_{value} = 10^{-4}$. On constate donc que quel que soit le nombre de réplicates utilisé, la méthode K++R est celle qui fournit les meilleurs résultats en termes de prédiction et elle est différente significativement de la deuxième méthode de classement.

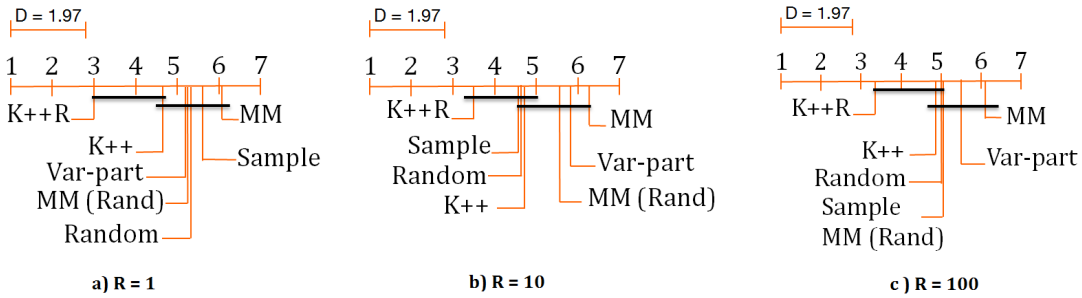


FIGURE 4.7 – Comparaison des méthodes d'initialisation (précédées par le prétraitement "RN" ou/et "Basic-Grouping") en utilisant le test de Friedman couplé au test post-hoc de Nemenyi (ARI en test)

Bilan

Selon les résultats des tests statistiques effectués dans cette première partie d'expérimentation, nous constatons que quel que soit le prétraitement utilisé, la méthode K++R fournit de meilleurs résultats en termes de prédiction par rapport aux autres méthodes. Le tableau 4.4 présente les résultats des performances prédictives moyennes (en termes d'ARI) en utilisant Conditional Info et Rank Normalization. Les résultats présentés pour la méthode K++R sont obtenus lorsque l'algorithme est exécuté une seule fois (*i.e.,* $R = 1$), tandis que les résultats présentés pour les autres méthodes sont obtenus lorsque l'algorithme est exécuté 100 fois (*i.e.,* $R = 100$). A partir de ces résultats, nous observons que la méthode K++R arrive à partir d'une seule exécution à fournir des résultats largement meilleurs pour quelques jeux de données ainsi que des résultats compétitifs pour le reste des jeux de données.

		R=100						R=1
	Données	**Sample**	**Random**	**K++**	**Var-Part**	**MM (Rand)**	**MM**	**K++R**
Conditional Info ($K = J$)	German	0.04 ± 0.04	0.04 ± 0.04	0.04 ± 0.04	0.03 ± 0.04	0.05 ± 0.04	0.07 ± 0.02	$\mathbf{0.12 \pm 0.02}$
	Australian	$\mathbf{0.5 \pm 0.06}$	$\mathbf{0.5 \pm 0.06}$	$\mathbf{0.5 \pm 0.06}$	0.49 ± 0.07	$\mathbf{0.5 \pm 0.06}$	0.35 ± 0.09	$\mathbf{0.5 \pm 0.06}$
	LED	0.47 ± 0.04	0.48 ± 0.03	0.48 ± 0.03	0.41 ± 0.04	0.44 ± 0.02	0.27 ± 0.02	$\mathbf{0.53 \pm 0.03}$
	Hepatitis	0.05 ± 0.06	0.05 ± 0.06	0.05 ± 0.06	0.03 ± 0.07	0.08 ± 0.1	0.27 ± 0.15	$\mathbf{0.20 \pm 0.11}$
	Heart	0.30 ± 0.11	0.30 ± 0.11	0.30 ± 0.11	0.17 ± 0.09	0.30 ± 0.11	0.22 ± 0.11	$\mathbf{0.36 \pm 0.09}$
	Glass	0.52 ± 0.12	0.52 ± 0.12	0.53 ± 0.11	0.79 ± 0.06	0.49 ± 0.09	0.30 ± 0.01	$\mathbf{0.82 \pm 0.06}$
	Breast	0.87 ± 0.06	0.87 ± 0.06	0.87 ± 0.06	0.87 ± 0.06	0.87 ± 0.06	0.8 ± 0.14	$\mathbf{0.88 \pm 0.04}$
	Iris	0.62 ± 0.08	0.62 ± 0.08	0.62 ± 0.08	0.61 ± 0.09	0.62 ± 0.08	0.60 ± 0.07	$\mathbf{0.72 \pm 0.07}$
	Pima	-0.02 ± 0.02	-0.02 ± 0.02	-0.02 ± 0.02	-0.02 ± 0.01	-0.02 ± 0.02	0.07 ± 0.08	$\mathbf{0.10 \pm 0.07}$
	Wine	0.91 ± 0.06	0.91 ± 0.06	0.91 ± 0.06	0.86 ± 0.06	0.86 ± 0.06	0.66 ± 0.14	$\mathbf{0.92 \pm 0.06}$
	Tictactoe	0.11 ± 0.03	0.11 ± 0.03	0.11 ± 0.03	0.11 ± 0.03	0.10 ± 0.02	0.05 ± 0.05	$\mathbf{0.14 \pm 0.02}$
	Vehicle	0.17 ± 0.02	0.17 ± 0.02	0.17 ± 0.02	$\mathbf{0.19 \pm 0.02}$	0.14 ± 0.02	0.1 ± 0.04	$\mathbf{0.19 \pm 0.04}$
	Horsecolic	$\mathbf{0.37 \pm 0.06}$	0.37 ± 0.06	$\mathbf{0.37 \pm 0.06}$	0.29 ± 0.1	0.14 ± 0.1	0.11 ± 0.07	$\mathbf{0.37 \pm 0.06}$
	Abalone	0.05 ± 0.01	0.05 ± 0.01	0.05 ± 0.01	$\mathbf{0.06 \pm 0.01}$	$\mathbf{0.06 \pm 0.01}$	$\mathbf{0.06 \pm 0.01}$	$\mathbf{0.06 \pm 0.01}$
	Segmentation	$\mathbf{0.70 \pm 0.04}$	$\mathbf{0.70 \pm 0.04}$	$\mathbf{0.70 \pm 0.04}$	0.67 ± 0.03	0.7 ± 0.03	0.63 ± 0.03	0.68 ± 0.02
	Soybean	0.52 ± 0.05	0.53 ± 0.04	0.53 ± 0.06	0.45 ± 0.02	0.51 ± 0.04	0.42 ± 0.03	$\mathbf{0.61 \pm 0.04}$
	Waveform	0.22 ± 0.02	0.22 ± 0.02	0.22 ± 0.02	0.22 ± 0.02	0.22 ± 0.02	0.14 ± 0.07	$\mathbf{0.23 \pm 0.04}$
	Adult	0.15 ± 0.02	0.15 ± 0.02	0.15 ± 0.02	0.15 ± 0.02	0.15 ± 0.02	0.04 ± 0.07	$\mathbf{0.17 \pm 0.02}$
	Mushroom	$\mathbf{0.94 \pm 0.01}$	$\mathbf{0.94 \pm 0.01}$	$\mathbf{0.94 \pm 0.01}$	$\mathbf{0.94 \pm 0.01}$	$\mathbf{0.94 \pm 0.01}$	0.06 ± 0.01	$\mathbf{0.94 \pm 0.01}$
	PenDigits	0.56 ± 0.01	0.56 ± 0.02	0.56 ± 0.02	0.51 ± 0.02	0.56 ± 0.02	0.48 ± 0.02	$\mathbf{0.62 \pm 0.01}$
	Phoneme	$\mathbf{0.73 \pm 0.01}$	$\mathbf{0.73 \pm 0.01}$	$\mathbf{0.73 \pm 0.01}$	0.72 ± 0.01	$\mathbf{0.73 \pm 0.01}$	$\mathbf{0.73 \pm 0.01}$	$\mathbf{0.73 \pm 0.01}$

		R=100						R=1
	Données	**Sample**	**Random**	**K++**	**Var-Part**	**MM (Rand)**	**MM**	**K++R**
Rank Normalization + BGB ($K = J$)	German	0.01 ± 0.01	0.01 ± 0.01	0.01 ± 0.01	0.01 ± 0.01	0.01 ± 0.01	$\mathbf{0.02 \pm 0.02}$	0.01 ± 0.01
	Australian	$\mathbf{0.22 \pm 0.02}$	$\mathbf{0.22 \pm 0.02}$	$\mathbf{0.22 \pm 0.02}$	$\mathbf{0.22 \pm 0.02}$	$\mathbf{0.22 \pm 0.02}$	0.04 ± 0.06	$\mathbf{0.22 \pm 0.02}$
	LED	0.47 ± 0.02	0.47 ± 0.02	0.48 ± 0.02	0.45 ± 0.01	0.47 ± 0.01	0.36 ± 0.02	$\mathbf{0.51 \pm 0.01}$
	Hepatitis	0.17 ± 0.11	0.17 ± 0.11	0.17 ± 0.11	0.17 ± 0.11	0.17 ± 0.11	-0.01 ± 0.03	$\mathbf{0.19 \pm 0.12}$
	Heart	$\mathbf{0.40 \pm 0.06}$	$\mathbf{0.40 \pm 0.06}$	$\mathbf{0.40 \pm 0.06}$	0.39 ± 0.05	$\mathbf{0.40 \pm 0.06}$	0.25 ± 0.14	$\mathbf{0.40 \pm 0.06}$
	Glass	0.29 ± 0.06	0.29 ± 0.05	$\mathbf{0.3 \pm 0.04}$	0.26 ± 0.05	0.24 ± 0.04	0.23 ± 0.04	$\mathbf{0.3 \pm 0.07}$
	Breast	$\mathbf{0.90 \pm 0.03}$	$\mathbf{0.90 \pm 0.03}$	$\mathbf{0.90 \pm 0.03}$	$\mathbf{0.90 \pm 0.03}$	$\mathbf{0.90 \pm 0.03}$	$\mathbf{0.90 \pm 0.03}$	$\mathbf{0.90 \pm 0.03}$
	Iris	$\mathbf{0.66 \pm 0.09}$	$\mathbf{0.66 \pm 0.09}$	$\mathbf{0.66 \pm 0.09}$	0.64 ± 0.08	$\mathbf{0.66 \pm 0.09}$	0.65 ± 0.09	0.64 ± 0.09
	Pima	0.08 ± 0.02	0.08 ± 0.02	0.08 ± 0.02	0.09 ± 0.02	0.08 ± 0.02	0.10 ± 0.03	$\mathbf{0.11 \pm 0.02}$
	Wine	$\mathbf{0.90 \pm 0.05}$	$\mathbf{0.90 \pm 0.05}$	$\mathbf{0.90 \pm 0.05}$	0.87 ± 0.05	0.88 ± 0.06	0.87 ± 0.04	$\mathbf{0.9 \pm 0.06}$
	Tictactoe	0.01 ± 0.01	0.01 ± 0.01	0.01 ± 0.01	$\mathbf{0.07 \pm 0.01}$	0.01 ± 0.01	0.01 ± 0.01	$\mathbf{0.07 \pm 0.02}$
	Vehicle	0.08 ± 0.01	0.08 ± 0.01	0.08 ± 0.01	0.08 ± 0.01	0.08 ± 0.01	0.08 ± 0.01	$\mathbf{0.12 \pm 0.01}$
	Horsecolic	0.08 ± 0.04	0.09 ± 0.05	0.09 ± 0.04	0.08 ± 0.04	0.08 ± 0.04	0.02 ± 0.02	$\mathbf{0.12 \pm 0.05}$
	Abalone	$\mathbf{0.05 \pm 0.0}$	$\mathbf{0.05 \pm 0.0}$	$\mathbf{0.05 \pm 0.01}$	$\mathbf{0.05 \pm 0.0}$	$\mathbf{0.05 \pm 0.0}$	$\mathbf{0.05 \pm 0.0}$	$\mathbf{0.05 \pm 0.01}$
	Segmentation	0.52 ± 0.01	0.52 ± 0.01	0.52 ± 0.01	0.50 ± 0.01	0.52 ± 0.01	0.52 ± 0.01	$\mathbf{0.54 \pm 0.01}$
	Soybean	0.52 ± 0.04	0.51 ± 0.05	0.5 ± 0.04	0.4 ± 0.03	0.52 ± 0.03	0.39 ± 0.02	$\mathbf{0.64 \pm 0.04}$
	Waveform	0.26 ± 0.0	0.26 ± 0.0	0.26 ± 0.0	0.26 ± 0.0	0.26 ± 0.0	0.26 ± 0.0	0.28 ± 0.04
	Adult	0.14 ± 0.05	0.14 ± 0.05	0.14 ± 0.05	0.14 ± 0.05	0.14 ± 0.05	0.14 ± 0.05	$\mathbf{0.18 \pm 0.01}$
	Mushroom	$\mathbf{0.62 \pm 0.01}$	$\mathbf{0.62 \pm 0.01}$	$\mathbf{0.62 \pm 0.01}$	$\mathbf{0.62 \pm 0.01}$	$\mathbf{0.62 \pm 0.01}$	0.01 ± 0.01	$\mathbf{0.62 \pm 0.0}$
	PenDigits	0.57 ± 0.02	0.57 ± 0.01	0.57 ± 0.01	0.59 ± 0.02	0.57 ± 0.01	0.57 ± 0.02	$\mathbf{0.63 \pm 0.04}$
	Phoneme	$\mathbf{0.61 \pm 0.01}$	$\mathbf{0.61 \pm 0.01}$	$\mathbf{0.61 \pm 0.01}$	$\mathbf{0.61 \pm 0.01}$	$\mathbf{0.61 \pm 0.01}$	0.46 ± 0.03	$\mathbf{0.61 \pm 0.01}$

TABLE 4.4 – Performance prédictive en terme d'ARI lorsque $K = J$ en utilisant conditional Info et Rank Normalization

4.6 Cas où le nombre de clusters (K) est supérieur au nombre de classes (J)

Lorsque le nombre de clusters est supérieur au nombre de classes, les approches de clustering prédictif cherchent à décrire et à prédire d'une manière simultanée. L'objectif est alors de trouver au cours de la phase d'apprentissage, le meilleur compromis entre la compacité et la pureté des groupes appris. Il s'agit de découvrir la structure interne de la variable cible. La prédiction de la classe des nouvelles instances est réalisée en se basant sur cette structure. Puisqu'il n'existe pas de critère global permettant de mesurer ce compromis, dans ce qui suit les performances prédictives des méthodes seront évaluées en utilisant l'ARI et la compacité des groupes formés sera évaluée en utilisant la MSE. Finalement, une discussion sera mener sur le compromis prédiction\compacité. L'ensemble des tableaux qui servent à obtenir les résultats synthétiques présentés dans cette deuxième partie d'expérimentation sont détaillés dans l'Annexe C.

4.6.1 Évaluation de la prédiction

Pour cet axe d'évaluation, quel que soit le prétraitement utilisé, nous commençons par tracer pour chaque jeu de données et pour chaque méthode d'initialisation la courbe d'ARI en fonction du nombre de clusters (de $K = J$ jusqu'à K_1 pour Conditional Info et de $K = J$ jusqu'à K_2 pour Rank Normalization). Pour plus de détails sur la manière d'obtention de K_1 et de K_2 , voir Section 4.4 et\ou l'Annexe A de ce mémoire. L'aire sous cette courbe est ensuite calculée (ALC-ARI : Area under the Learning Curve de l'ARI). Dans ce cas, plus la valeur de l'aire est grande plus la méthode est bonne. La figure 4.8 présente un exemple illustratif de l'aire sous la courbe d'ARI calculée pour la méthode SB (précédée par Conditional Info) pour le jeu de données Heart. Le nombre de clusters K varie dans ce cas de $J = 2$ jusqu'à $K_1 = 23$.

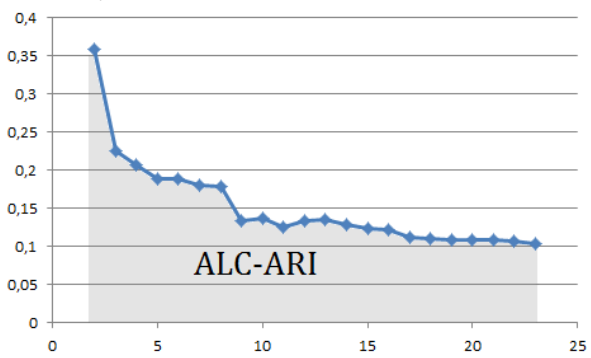


FIGURE 4.8 – L'aire sous la courbe d'ARI pour la méthode SB pour le jeu de données Heart en utilisant CI ($K_1 = 23$)

Muni des valeurs d'ALC-ARI, qui synthétisent les résultats de chaque méthode d'initialisation (voir tableau 4.2 de la section 4.4), nous appliquons le test de Friedman couplé au test post-hoc de Nemenyi sur les 21 jeux de données et pour un seuil de significativité $\alpha = 0.05$. Les valeurs d'aires qui ont servi à l'obtention de ces tests statistiques sont disponibles dans les tableaux C.4 et C.5 de l'Annexe C.

Avec le prétraitement supervisé : suivant les résultats du test statistique de Friedman, on observe que les méthodes d'initialisation sont différentes significativement quel que soit le nombre

de réplicates (R) utilisé. En outre, en s'appuyant sur les résultats du test de Nemenyi présentés dans la figure 4.9, on constate que ces méthodes peuvent être partitionnées en 4 groupes de telle sorte que la méthode RS suivie par les deux méthodes SB et CD sont celles qui fournissent de bons résultats en termes d'ARI. On observe également que lorsque le nombre de réplicates (R) augmente, les trois méthodes RS, SB et CD s'écartent des autres méthodes et deviennent plus performantes en termes de prédiction.

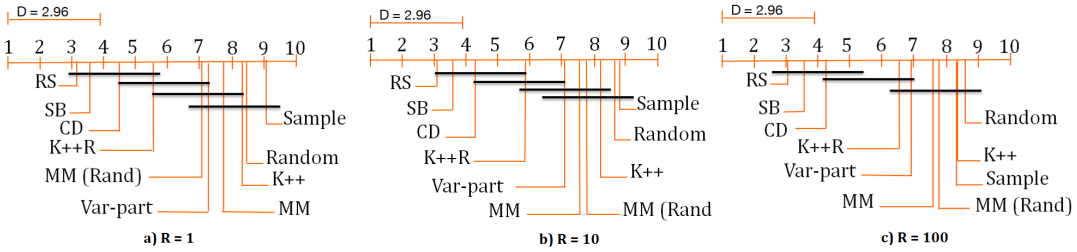


FIGURE 4.9 – Comparaison des méthodes d'initialisation (précédées par Conditional Info) en utilisant le test de Friedman couplé au test post-hoc de Nemenyi (ALC-ARI en test)

Avec le prétraitement non supervisé : même lorsqu'on change le prétraitement supervisé (CI) par le prétraitement non supervisé (RN et/ou BG), nous trouvons les mêmes résultats : la méthode RS suivie par les deux méthodes SB et CD sont les méthodes les plus performantes en termes d'ARI (voir la figure 4.10).

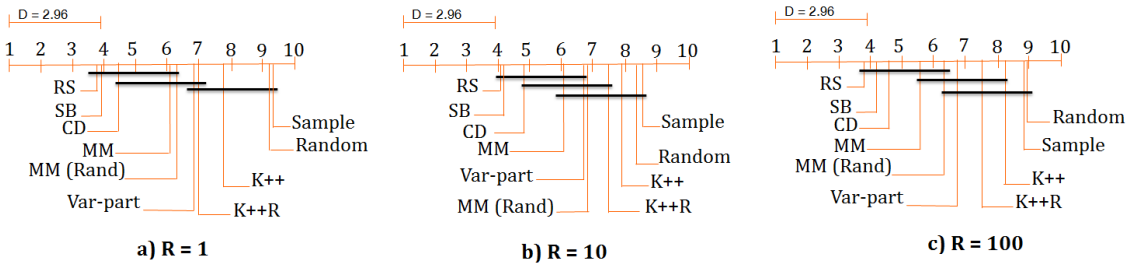


FIGURE 4.10 – Comparaison des méthodes d'initialisation (précédées par RN-BGB) en utilisant le test de Friedman couplé au test post-hoc de Nemenyi (ALC-ARI en test)

4.6.2 Évaluation de la compacité

Pour l'axe de la description, nous procédons de la même manière que pour la prédiction : pour chaque méthode d'initialisation et pour chaque jeu de données, nous traçons la courbe de l'erreur quadratique moyenne MSE (voir l'équation 4.2 de la Section 4.4) en fonction du nombre de clusters puis nous calculons l'aire sous la courbe (ALC-MSE). Dans ce cas, plus la valeur de l'aire est petite meilleure est la méthode. Les valeurs d'aires qui ont servi à l'obtention de ces tests statistiques sont disponibles dans les tableaux C.6 et C.7 de l'Annexe C.

Avec le prétraitement supervisé : Lorsque $R = 1$, on remarque que les quatre méthodes Var-Part, SB, K++ et K++R sont meilleures que les autres méthodes en termes de compacité des groupes appris (la figure 4.11 a)). Lorsqu'on augmente R (*i.e.,* $R \in \{10, 100\}$), on trouve que les trois méthodes K++, K++R et Sample deviennent plus performantes que les deux méthodes Var-Part et SB.

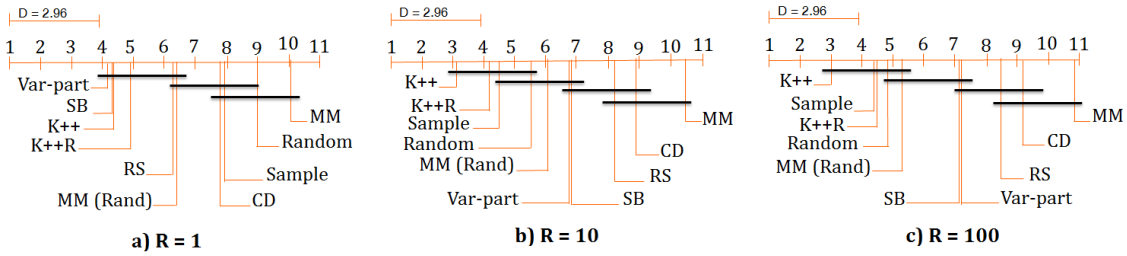


FIGURE 4.11 – Comparaison des méthodes d'initialisation (précédées par Conditional Info) en utilisant le test de Friedman couplé au test post-hoc de Nemenyi (ALC-MSE en test)

Avec le prétraitement non supervisé : A partir de la figure 4.12, on observe que quel que soit le nombre de réplicates R, les méthodes d'initialisation se rapprochent en termes de performance (la constitution d'un seul groupe suivant le test de Nemenyi). Néanmoins, les deux méthodes SB et K++R restent les premières en classement.

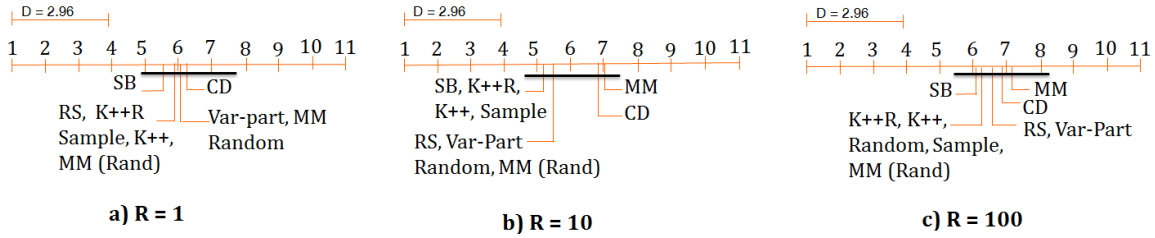


FIGURE 4.12 – Comparaison des méthodes d'initialisation (précédées par RN-BGB) en utilisant le test de Friedman couplé au test post-hoc de Nemenyi (ALC-MSE en test)

4.6.3 Évaluation du compromis

Une bonne méthode d'initialisation suivant le principe des K-moyennes prédictives est celle qui cherche à trouver le bon compromis entre la prédiction et la compacité. Puisqu'il n'existe pas dans la littérature un critère global permettant d'évaluer ce compromis, on s'est basé alors sur le principe du Front de Pareto. Les figures 4.13, 4.14 et 4.15 présentent respectivement, pour les 21 jeux de données, le rang des quatre premières méthodes obtenues pour l'ARI vis-à-vis de la MSE (en utilisant Conditional Info a) et Rank Normalization b)) dans le cas où $R = 1, 10$, et 100. Sur ces figures, plus les points approchent de l'origine des axes, plus la méthode utilisée sur ces jeux de données arrive à réaliser un bon compromis entre les deux critères. Lorsque on utilise Conditional Info comme un prétraitement, nous observons que les deux méthodes qui arrivent à atteindre un bon compromis entre la MSE et l'ARI sont : SB et RS pour $R = 1$, SB et K++R suivie par RS pour $R = 10$ et SB, RS et K++ pour $R = 100$ (voir le graphiques gauche de chaque figure). Lorsqu'on utilise Rank Normalization et Basic Grouping comme un prétraitement nous observons que quel que soit le nombre de réplicates R utilisé, les méthodes d'initialisation ont presque la même performance en termes de MSE. Cependant, les deux méthodes RS et SB ont une meilleure performance en termes d'ARI.

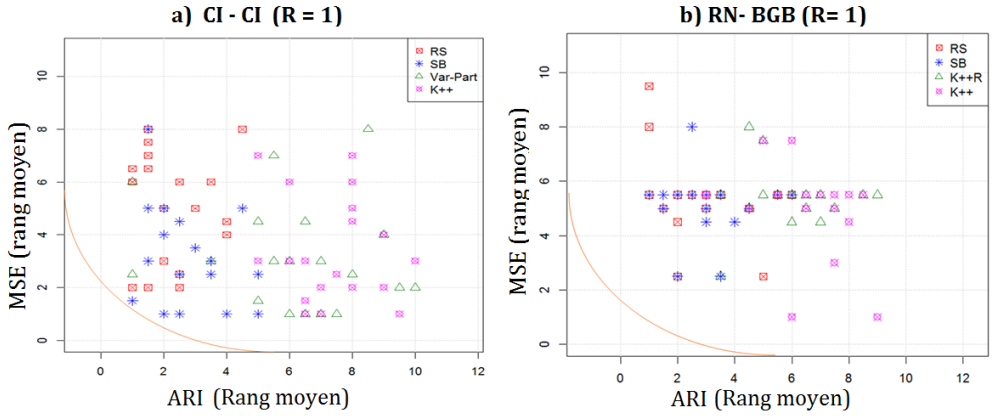


FIGURE 4.13 – Le rang des quatre premières méthodes en termes d'ARI vis-à-vis de la MSE pour $R = 1$ (la ligne orange représente un guide de lecture)

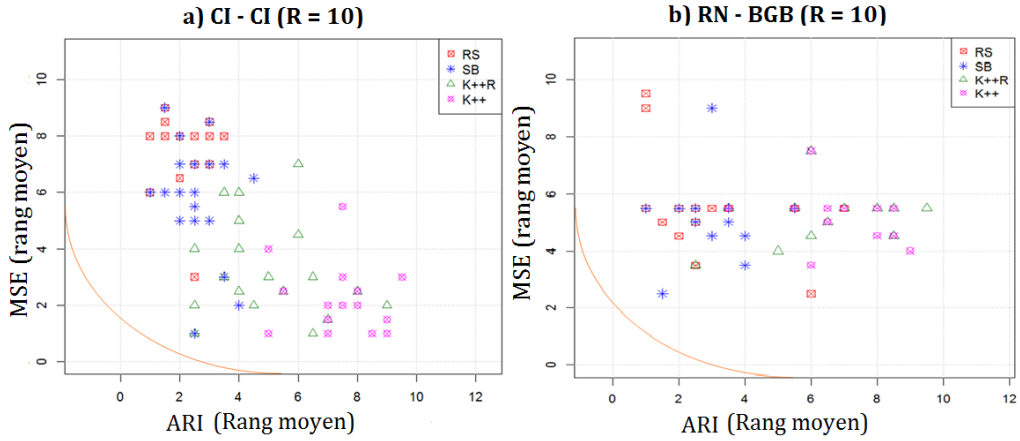


FIGURE 4.14 – Le rang des quatre premières méthodes en termes d'ARI vis-à-vis de la MSE pour $R = 10$ (la ligne orange représente un guide de lecture)

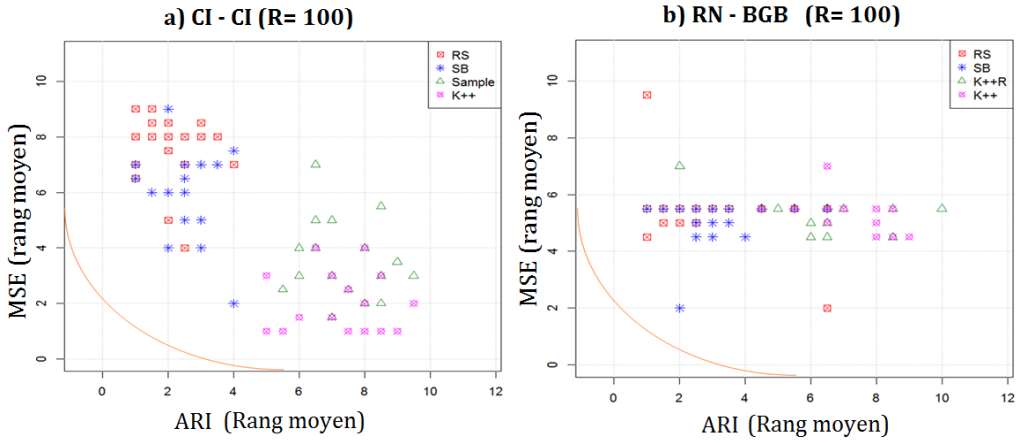


FIGURE 4.15 – Le rang des quatre premières méthodes en termes d'ARI vis-à-vis de la MSE pour $R = 100$ (la ligne orange représente un guide de lecture)

Une autre façon de comparer les méthodes d'initialisation proposées en termes de performance est de mesurer les écarts de performances (en termes de valeurs) entre les méthodes proposées et une méthode prise en référence. Par exemple, pour le critère à maximiser ARI et pour une méthode A, la formule utilisée est la suivante : $(ARI(A) - ARI(ref)/ARI(ref)) \times 100$. Une valeur positive signifie que la méthode A a de meilleure performance en termes d'ARI vis à vis de la méthode ref et vice versa. La méthode de référence choisie dans cette étude comparative est la méthode qui fournit une bonne performance en termes de description à savoir, KMean++. La figure 4.16 présente la comparaison en pourcentage de l'ALC-ARI et de l'ALC-MSE des méthodes d'initialisation proposées vis-à-vis de la méthode de référence KMean++ lorsque l'algorithme est exécuté 1 fois (partie gauche de la figure), 10 fois (partie milieu de la figure) et 100 fois (partie droite de la figure). Dans cette figure, les valeurs de l'ALC-ARI et de l'ALC-MSE représentent une moyenne sur les 21 jeux de données. Nous constatons ici que lorsque l'algorithme des K-moyennes n'est exécuté qu'une seule fois, les trois méthodes d'initialisation proposées ont de meilleures performances en termes d'ARI par rapport à la méthode de référence et une performance en termes de MSE presque similaire à la méthode KMean++ (par exemple, pour la méthode RS, on trouve 19% en ALC-ARI et −3% en ALC-MSE). Lorsque l'algorithme est exécuté plusieurs fois (*i.e.*, $R \in \{10, 100\}$), la méthode de référence KMean++ devient meilleure en termes de MSE (e.g, -11% de l'ALC-MSE pour S-Bisecting lorsque $R = 100$) mais elle reste moins performante que les méthodes proposées en termes d'ARI (e.g, 19% de l'ALC-ARI pour S-Bisecting lorsque $R = 100$).

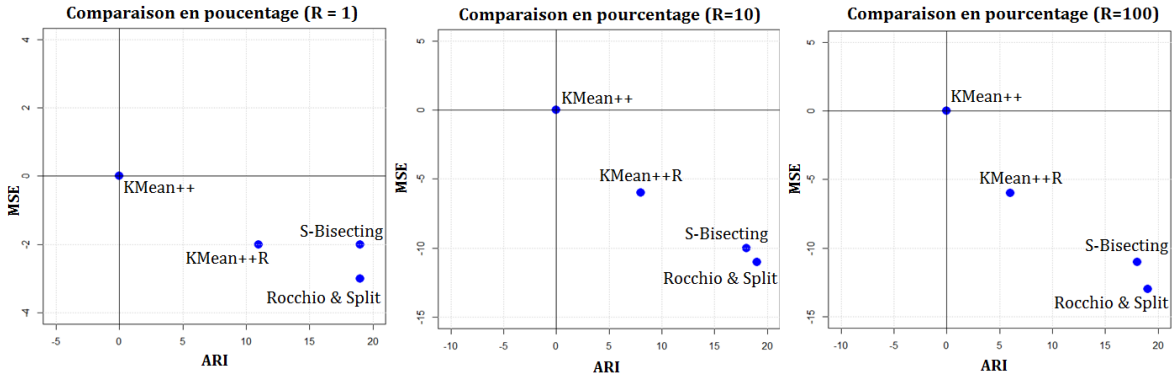


FIGURE 4.16 – Comparaison (en pourcentage) de la méthode KMean++ avec les trois méthodes d'initialisation proposées

Discussion : En se basant sur l'ensemble des résultats obtenus dans cette deuxième partie d'expérimentation, nous constatons que :

- *Pour l'axe de prédiction* : quel que soit le prétraitement utilisé (Conditional Info ou Rank Normalization et/ou basic Grouping), nous remarquons que l'algorithme des K-moyennes standard précédé par la méthode Rocchio-and-Split (RS) fournit de meilleurs résultats en termes de prédiction par rapport à sa performance en utilisant les autres méthodes d'initialisation.
- *Pour l'axe de description* : Lorsque Conditional Info est utilisé, la méthode Var-Part fournit de meilleurs résultats en termes de MSE quand l'algorithme des K-moyennes est exécuté qu'une seule fois ($R = 1$). Quand on augmente la valeur du nombre de réplicates R ($R \in \{10, 100\}$), les trois méthodes K++, K++R et Sample deviennent celles qui fournissent de bons résultats en termes de MSE. Lorsque Rank Normalization et/ou Basic

Grouping [13] est utilisé, les performances en termes de la MSE des méthodes d'initialisation deviennent plus proches les unes des autres.

— *Pour le compromis* : quel que soit le prétraitement utilisé et le nombre de réplicates R, la méthode SB suivie par la méthode RS sont les deux méthodes qui parviennent à établir un certain compromis entre la description et la prédiction. A titre d'exemple, la méthode S-Bisecting perd 10% de la MSE mais gagne en parallèle 20% en ARI vis-à-vis de la méthode KMeans++. Cependant, lorsque le nombre de clusters est élevé la méthode Rocchio-and-split (SB) reste préférable à la méthode S-Bisecting en raison de sa complexité (SB nécessite beaucoup d'effort de calcul).

4.7 Bilan et synthèse

Ce chapitre a présenté l'influence d'une étape d'initialisation supervisée sur la qualité des résultats générés par l'algorithme des K-moyennes standard. Nous avons pu montrer qu'une bonne méthode d'initialisation supervisée a la capacité d'aider cet algorithme à atteindre l'objectif des K-moyennes prédictives (i.e., le compromis entre la prédiction et la compacité). Ce résultat reste inchangé quel que soit le prétraitement utilisé (supervisé ou non supervisé). Dans le cas où le nombre de clusters est égal au nombre de classes (problème de classification supervisée), la méthode K-Mean++R parvient à obtenir de meilleurs résultats en termes de prédiction en une seule exécution (méthode déterministe). Dans le cas où le nombre de clusters est supérieur au nombre de classes, SB et RS sont les meilleures méthodes (parmi les 10 méthodes) arrivant à atteindre un certain compromis entre la prédiction et la compacité. La figure 4.17 présente l'évolution de l'ALC-ARI (moyenne sur les 21 jeux de données) suivant le prétraitement et la méthode d'initialisation utilisé. A partir de cette figure nous remarquons que l'utilisation d'un prétraitement supervisé et d'une méthode d'initialisation supervisée améliore davantage la performance prédictive de l'algorithme des K-moyennes classique (en passant de 0.17 pour RN-BGB et K++ à 0.38 pour CI-CI et RS).

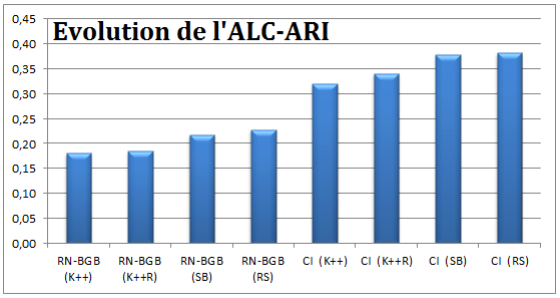


FIGURE 4.17 – Évolution de l'ALC-ARI (en test) suivant le prétraitement et la méthode d'initialisation utilisée

La figure 4.18, quant à elle, présente l'évolution de l'ALC-MSE (moyenne sur les 21 jeux de données) suivant la méthode d'initialisation utilisée en utilisant Conditional Info (partie gauche de la figure) et Rank Normalization\Basic Grouging (partie droite de la figure) comme prétraitement. Ces deux graphiques montrent que les méthodes d'initialisation supervisées ont une

13. Il est à rappeler que le choix du prétraitement utilisé dépend de la nature des variables existant dans le jeu de données : on utilise Rank Normalization pour les variables continues et Basic Grouping pour les variables catégorielles.

performance similaire en termes de MSE à la méthode non supervisée KMean++ lorsque le prétraitement non supervisé est utilisé et une performance moins bonne que celle de la méthode non supervisée.

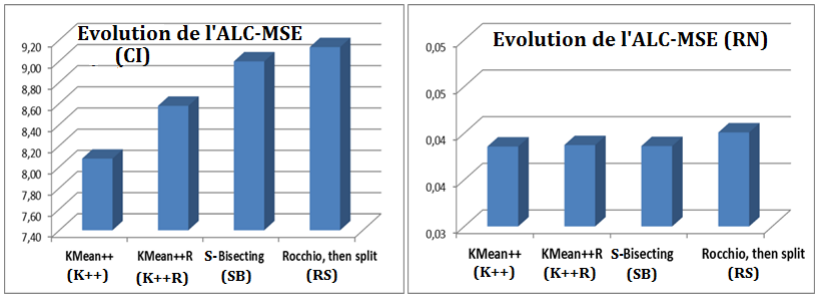


FIGURE 4.18 – Évolution de l'ALC-MSE (en test) suivant la méthode d'initialisation utilisée pour CI (partie à gauche) et pour RN\BGB (partie à droite).

Dans le domaine d'apprentissage automatique, l'évaluation de la qualité des résultats fournis par un algorithme d'apprentissage est une tâche cruciale. Cependant, dans le cadre du clustering prédictif, il n'existe pas de critère global permettant de mesurer la qualité des résultats. Seul le principe du Front de Pareto peut être utilisé pour sélectionner la ou les méthodes qui réalisent le meilleur compromis entre la prédiction et la description (voir Section 4.6.3). De ce fait, la recherche d'un critère d'évaluation global qui permet de mesurer ce compromis s'avère nécessaire. Le chapitre qui suit sera consacré à la recherche de ce critère.