République Algérienne Démocratique et Populaire
Ministère de l'enseignement Supérieur et de la Recherche Scientifique
Université de Mohamed El Bachir El Ibrahimi de Bordj Bou Arréridj
Faculté des Mathématiques et d'Informatique
Département d'informatique

MEMOIRE
Présenté en vue de l'obtention du diplôme
Master en informatique
Spécialité : Technologies de l'information et des communications

THEME

Approche Multifacette pour la Maladie du Foie : Prédiction, Méta-Classification et Simulation de la Migration entre Stades

Présenté par :

Nouioua Imene

Nouioua Ratiba

Soutenu publiquement le : 22/06/2024

Devant le jury composé de :

Président : Dr. Attia Abdelouahab. MCA université de BBA

Examinateur : Dr. Benaouda Nadjib. MCB université de BBA

Encadrante : Dr. Boutouhami Sara. MCB université de BBA

2023/2024

Dédicace

Nous tenons à dédier cet humble travail comme preuve de respect, de gratitude et de reconnaissance : À nos chers parents, nos frères, nos sœurs, nos belles-sœurs, nos beaux-frères et amies pour leurs encouragements, soutiens, patiences et prières.

Remerciement

C'est avec un immense plaisir que nous réservons ces quelques lignes en signe de gratitude et de reconnaissance à tous ceux qui ont contribué de près ou de loin à l'élaboration de ce travail. Nous rendons grand merci à Dieu Tout-Puissant qui nous a donné une grande volonté et le courage pour faire cet humble travail.

À notre encadrante Dr. Boutouhami Sara, pour sa compréhension, sa disponibilité, son aide et ses précieux conseils qui nous ont été très utiles pour l'achèvement de ce projet.

Nous exprimons également notre gratitude envers les membres du jury Dr. Attia Abdelouahab et Dr. Benaouda Nadjib pour l'intérêt qu'ils ont porté à notre projet de fin d'études et d'avoir accepté d'examiner notre travail.

Nos remerciements s'étendent à tous nos enseignants du département d'Informatique de l'Université BBA, en particulier à Pr. Nouioua Farid pour son aide et ses orientations. Nous tenons à remercier aussi Dr. Nouioua Mourad pour son aide et ses conseils.

Résumé

Les maladies chroniques, notamment celles affectant le foie, représentent un défi majeur pour les systèmes de santé mondiaux. Dans ce mémoire, nous avons exploré différentes facettes de la prédiction à la simulation de migration entre les stades de ces affections chroniques. En utilisant des techniques avancées d'analyse de données et d'apprentissage automatique, notre étude se concentre sur quatre aspects clés : l'amélioration de la prédiction, la sélection des caractéristiques, l'optimisation des modèles et la méta-classification, ainsi que la simulation de la migration entre les stades de la maladie dans un objectif de prévention. À chaque étape, des expérimentations rigoureuses ont été menées pour valider notre méthodologie. Les résultats obtenus confirment l'importance cruciale de la prédiction pour anticiper l'évolution de la maladie, ainsi que l'efficacité de la sélection des caractéristiques et de l'optimisation des modèles pour améliorer les performances de prédiction. La méta-classification, en combinant les prédictions de différents modèles, renforce la fiabilité des résultats. De plus, la simulation de la migration entre les stades offre une meilleure compréhension des dynamiques de progression de la maladie.

Abstract

Chronic diseases, especially those affecting the liver, pose a major challenge to global healthcare systems. In this dissertation, we have explored various aspects from prediction to simulating the migration between stages of these chronic conditions. Utilizing advanced data analysis and machine learning techniques, our study focuses on four key aspects : improving prediction, feature selection, model optimization, and meta-classification, along with simulating the migration between disease stages for preventive purposes. At each stage, rigorous experiments were conducted to validate our methodology. The results confirm the crucial importance of prediction in anticipating disease progression, as well as the effectiveness of feature selection and model optimization in enhancing prediction performance. Meta-classification, by combining predictions from different models, enhances result reliability. Furthermore, simulating the migration between stages provides a better understanding of disease progression dynamics.

ملخص

الأمراض المزمنة، وخاصة تلك التي تؤثر على الكبد، تشكل تحدياًا كبيراً لأنظمة الرعاية الصحية العالمية. في هذا البحث استكشفنا جوانب مختلفة من التنبؤ إلى محاكاة الانتقال بين مراحل هذه الحالات المزمنة. باستخدام تقنيات تحليل البيانات المتقدمة والتعلم الآلي، تركز دراستنا على أربعة جوانب رئيسية: تحسين التنبؤ، واختيار السمات، وتحسين النماذج، والتصنيف الشامل، بالإضافة إلى تحاكي الانتقال بين مراحل المرض لأغراض الوقاية. في كل مرحلة، تم إجراء تجارب دقيقة لتحقيق صحة منهجيتنا. تؤكد النتائج على الأهمية الحاسمة للتنبؤ في توقع تقدم المرض، فضلاً عن فعالية اختيار السمات وتحسين النماذج في تعزيز أداء التنبؤ. التصنيف الشامل، من خلال دمج التنبؤات من النماذج المختلفة، يعزز موثوقية النتائج. علاوة على ذلك، فإن محاكاة الانتقال بين المراحل توفر فهماًا أفضل لديناميكيات تقدم المرض.

Table des matières

Table des figures

Liste des tableaux

Introduction Générale

1. Contexte et problématique

Les avancées dans le domaine de l'apprentissage automatique ont ouvert de nouvelles perspectives dans le domaine de la santé, notamment en ce qui concerne le traitement des maladies chroniques. Ces maladies, caractérisées par leur évolution sur le long terme et leur impact significatif sur la qualité de vie des individus, représentent un fardeau croissant pour les systèmes de santé à l'échelle mondiale. La cirrhose du foie, résultant d'une cicatrisation progressive du tissu hépatique, et les maladies cardiovasculaires figurent parmi les principales causes de morbidité et de mortalité dans de nombreuses régions du monde. Avec l'accumulation massive de données sur les patients, l'utilisation de techniques d'apprentissage automatique se présente comme une opportunité prometteuse pour améliorer la prédiction de ces maladies. Prédire et prévenir efficacement l'évolution de ces maladies, ainsi que mettre en place des stratégies de gestion efficaces, sont donc essentielles pour réduire leur impact sur la santé publique.

Cependant, l'analyse de ces données médicales massives pose des défis quant à la sélection des caractéristiques les plus pertinentes pour élaborer des modèles prédictifs précis et efficaces. Les ensembles de données peuvent être extrêmement complexes, contenant une multitude de variables, ce qui complique l'identification des facteurs les plus significatifs pour la prédiction de ces maladies. Ainsi, la sélection de caractéristiques (feature selection) devient un enjeu crucial pour développer des modèles de prédiction robustes et fiables dans le domaine de la santé.

L'état de santé des patients atteints de maladies chroniques peut évoluer d'un stade à un autre au fil du temps, ce qui nécessite une adaptation constante des stratégies de prévention et de traitement. Comprendre et prédire ces transitions entre les différents stades de la maladie est

essentiel pour améliorer la prise en charge des patients et optimiser les résultats cliniques.

2. Objectifs et contributions

Dans le cadre de notre travail de master, nous visons à développer des modèles prédictifs performants et robustes pour traiter efficacement des ensembles de données complexes. Pour atteindre cet objectif, nous intégrons des techniques avancées telles que l'apprentissage automatique, la sélection d'attributs et la méta-classification. Voici pourquoi ces techniques sont essentielles et comment elles se complètent pour améliorer les performances des modèles prédictifs.

L'apprentissage automatique permet la généralisation en construisant des modèles à partir de données historiques et en prédisant sur de nouvelles données. Dans notre étude, nous utilisons les techniques du Plus Proche Voisin (KNN), des Arbres de Décision (AD) et du Classifieur Bayésien (CB) pour construire des modèles prédictifs basés sur les données médicales "Mayo Clinic Primary Biliary Cirrhosis Data" en exploitant toutes les caractéristiques disponibles.

La sélection d'attributs réduit la dimensionnalité en identifiant les caractéristiques les plus pertinentes, améliorant ainsi la précision du modèle et prévenant le surapprentissage. Nous utilisons des techniques avancées de sélection de caractéristiques pour identifier les variables les plus informatives et réévaluer les modèles en n'utilisant que ces caractéristiques sélectionnées.

La méta-classification combine les forces de plusieurs classificateurs pour améliorer la précision des prédictions finales. Nous mettons en œuvre la méta-classification avec une validation croisée rigoureuse pour intégrer les prédictions des modèles individuels.

En combinant l'apprentissage automatique, la sélection d'attributs et la méta-classification, notre travail de master apportera une contribution significative dans le domaine de la prédiction en démontrant l'efficacité de ces techniques dans des scénarios réels.

La présence de différents stades de ces maladies nous a conduit à considérer la possibilité de passage d'un stade à un autre. C'est ce que nous avons appelé simulation de migration entre les stades de la maladie. Les patients sont souvent préoccupés par les changements qu'ils doivent apporter à leur état pour éviter de passer à un stade plus grave de leur maladie. Ce passage que

nous avons nommé (migration) peut représenter soit une amélioration ou malheureusement une détérioration de l'état de santé du patient. Souvent, ils espèrent ralentir l'évolution de la maladie vers une forme invalidante. L'idée est donc d'intervenir le plus tôt possible et de manière continue pour prévenir la progression et la gravité de la maladie. Nous avons proposé un algorithme de simulation de migration entre stades de la maladie en utilisant le méta-classifieur basé sur les attributs sélectionnés. L'algorithme s'inspire du principe du plus proche voisin, permettant de calculer de nouvelles valeurs de certains attributs (paramètres) en cherchant parmi les plus proches voisins ceux qui ont réussi la migration vers la classe souhaitée, tout en prenant en compte les paramètres que le patient souhaite maintenir ou accepter leur modification.

Notre décision de créer une application représente une réponse pratique aux besoins des patients qui cherchent à suivre de près leur état de santé et à comprendre pleinement l'impact des variations des paramètres sur leur bien-être. Cette plateforme leur offre un moyen convivial et accessible de visualiser et d'interpréter les données médicales, favorisant ainsi une prise de décision éclairée et une meilleure gestion de leur santé au quotidien.

3. Plan du mémoire

Le mémoire se structure de la manière suivante :

- Le premier chapitre expose brièvement le processus de fouille de données (Data Mining DM), ainsi que les méthodes et techniques utilisées dans ce domaine. Nous nous concentrons particulièrement sur les techniques d'apprentissage supervisé qui nous intéressent, à savoir le Plus Proche Voisin (KNN), les Arbres de Décision (AD) et le Classifieur Bayésien (CB).
- Dans le deuxième chapitre, nous nous pencherons en détail sur la sélection des attributs et explorerons les différentes techniques utilisées dans ce domaine.
- Le troisième chapitre : Conception, Réalisation et Modélisation, constitue le cœur de notre étude, où nous approfondissons la conception, la réalisation et la modélisation de nos approches pour l'amélioration de la prédiction de la maladie du foie. Nous abordons également le principe de la simulation de la migration entre classes. Nous présentons les bilans ainsi que des discussions des résultats que nous avons obtenus pour chaque étape. Ensuite, nous détaillons les différents outils et langages de programmation utilisés pour le développement de notre application, ainsi que ses interfaces.

- Enfin, nous clôturons ce mémoire avec une conclusion générale qui résume les contributions de notre travail, ainsi que quelques perspectives pour de futurs développements.

Chapitre 1

Introduction à la fouille de données et à la classification supervisée

1.1 Introduction

Dans ce chapitre, nous explorons le processus d'extraction de connaissances à partir des données, également connu sous le nom de fouille de données, en mettant particulièrement l'accent sur la classification supervisée. La classification supervisée constitue l'une des tâches centrales de la fouille de données, visant à extraire des connaissances à partir des données en construisant un modèle de classification à partir d'un ensemble d'exemples étiquetés avec leur classe. Ce modèle est ensuite utilisé pour prédire la classe de nouveaux exemples. Cette tâche revêt une grande importance car elle permet de prendre des décisions basées sur les caractéristiques des données, en les classant dans des catégories prédéfinies. Parmi les techniques d'apprentissage supervisé, nous allons nous pencher sur trois techniques à savoir le Plus Proche Voisin (KNN), les Arbres de Décision (AD) et le Classifieur Bayésien (CB). En plus de cela, nous explorerons également la technique de méta-classification, une approche avancée qui combine plusieurs modèles pour améliorer la précision des prédictions.

1.2 Définition de la fouille de données

La fouille de données est née de l'explosion des volumes de données stockées et des progrès réalisés dans le traitement et le stockage de ces données. Son objectif est d'extraire des informa-

tions utiles à partir de ces vastes ensembles de données, afin de mieux comprendre les données existantes ou de prédire leur comportement futur. Elle s'intègre dans le processus d'extraction de connaissances à partir des données, également connu sous le nom de KDD (Knowledge Discovery from Data) [1]. Ce domaine utilise des techniques de reconnaissance de formes, ainsi que des méthodes statistiques et mathématiques, pour découvrir de nouvelles connaissances dans les données stockées dans des entrepôts. Ces connaissances peuvent prendre la forme de corrélations, de schémas ou de tendances initialement inconnus. La fouille de données est essentielle dans de nombreux secteurs, tels que la finance, la santé, le commerce électronique, etc., pour optimiser la prise de décision et améliorer les performances. Elle repose sur l'analyse des données, la construction de modèles prédictifs et leur validation sur des ensembles de données distincts. Actuellement, elle utilise une variété d'outils manuels et automatiques pour extraire des informations utiles des données, et elle revêt une grande importance économique en permettant d'optimiser la gestion des ressources humaines et matérielles. En effet, elle est utilisée dans divers domaines tels que la prise de décision de crédit, l'optimisation des ressources dans les transports et les hébergements, l'organisation de rayonnages dans les magasins, la planification de campagnes publicitaires, le diagnostic médical, l'analyse génomique, la classification d'objets, le commerce électronique, l'analyse des pratiques commerciales, les moteurs de recherche sur internet, l'extraction d'informations à partir de textes et l'analyse des données évolutives dans le temps [2].

1.3 Techniques de classification supervisée

Les méthodes de la fouille de données peuvent être classées selon différents critères. Selon le traitement appliqué, elles sont généralement catégorisées en deux grandes classes : les méthodes supervisées et les méthodes non supervisées. Les méthodes supervisées sont utilisées lorsque les données sont étiquetées, ce qui signifie que chaque exemple est associé à une classe ou à une valeur de sortie. Ces méthodes visent à construire un modèle à partir de ces exemples étiquetés pour prédire la classe ou la valeur de sortie d'exemples non étiquetés. Les méthodes non supervisées sont employées lorsque les données ne sont pas étiquetées. Elles cherchent à découvrir des structures ou des régularités dans les données, sans préjuger des classes ou des valeurs de sortie [1].

Dans la démarche de classification supervisée, les classes ainsi que leur nombre sont préa-

lablement définis. L'objectif principal est d'assigner des objets à des classes spécifiques en se basant sur leurs caractéristiques distinctives [2].

Le processus de classification supervisée s'articule autour des étapes suivantes :

1. **Etape de construction du modèle** : Durant étape, l'algorithme apprend à partir des données en créant un ensemble de règles de classification, représentant ainsi le modèle d'apprentissage.
2. **Etape de prédiction** : Pendant cette étape, les données de test sont utilisées pour évaluer l'exactitude des règles de classification établies lors de l'étape précedente. Si le modèle atteint un niveau de précision acceptable, les règles peuvent être appliquées à de nouvelles données.

La construction d'un modèle prédictif suit généralement trois phases distinctes :

- **Phase d'entraînement** : Un échantillon d'entraînement est utilisé pour développer le modèle.
- **Phase de validation** : Un échantillon de validation est employé pour évaluer la performance du modèle sur des données non utilisées dans l'entraînement, afin d'éviter le sur-apprentissage. La performance du modèle peut être évaluée selon différents critères.
- **Phase de test** : Un échantillon de test est utilisé pour évaluer la performance finale du modèle. Cette étape est essentielle pour obtenir une évaluation rigoureuse de l'efficacité du modèle.

Typiquement, les données sont aléatoirement réparties en trois échantillons :

- L'échantillon d'entraînement comprend généralement entre 50% et 80% des données.
- L'échantillon de validation représente entre 20% et 40% des données.
- L'échantillon de test utilise entre 5% et 10% des données (cette phase est parfois omise en pratique).

Cette approche garantit une évaluation précise de la performance du modèle tout en évitant le sur-apprentissage.

Il existe de nombreuses méthodes de classification supervisée ; nous citons quelques-unes et nous ne détaillons que les trois techniques que nous allons utiliser dans notre système :

- K plus proches voisins « KNN »
- Régression logistique « RL »
- Arbres de décision « AD »

- Support victor machines « SVM »
- Classifieur bayésien « CB »

1.3.1 La méthode de classification « K plus proches voisins (KNN) »

La méthode des plus proches voisins (parfois notée k-PPV, k-PP ou kNN pour Nearest Neighbor) est une méthode supervisée très intuitive. Son principe peut être assimilé à l'analogie suivante : "Dis-moi qui sont tes voisins, je te dirai qui tu es". Le fonctionnement général de la méthode des kNN consiste à déterminer, pour chaque nouvel individu à classer, la liste des plus proches voisins parmi les individus déjà classés. En d'autres termes, l'objectif de l'algorithme est de classer les exemples non étiquetés en se basant sur leurs similarités avec les exemples de la base d'apprentissage. Cette méthode nécessite de choisir une mesure de distance, la plus courante étant la distance euclidienne, ainsi que le nombre de voisins à prendre en compte pour la classification [3].

1.3.1.1 Principe de la construction

Le principe de l'algorithme des k-plus proches voisins (kNN) est assez simple. Il nécessite les éléments suivants :

- Un ensemble de données d'apprentissage D ;
- Une fonction de distance d ;
- Un entier k nombre de voisins à considérer.

Pour chaque nouvel élément à classer (x), pour lequel une décision doit être prise, l'algorithme recherche dans l'ensemble de données D les k éléments les plus proches de x selon la fonction de distance d. Ensuite, il attribue à x la classe qui est la plus fréquente parmi ces k voisins [1]. Le fonctionnement de l'algorithme K-NN peut être schématisé à l'aide du pseudocode suivant [1][4] :

Algorithm 1: Pseudo-Algorithme K-NN

Data: D : un ensemble de données;
d : une fonction de définition de distance;
k : un nombre entier;
x : une nouvelle observation à prédire;
Result: La valeur prédite y pour x

foreach *observation* $d_i \in D$ **do**
    Calculer la distance $d(x, d_i)$;
end
Sélectionner les k observations les plus proches de x selon d;
Calculer le mode des valeurs y des k observations sélectionnées;
return *La valeur calculée comme prédiction pour* x;

Dans l'exemple de la figure 1.1, l'élément à classer (cercle vert) pourrait être classé soit dans la première classe de carré bleu ou la seconde classe de triangles rouges. Si $k = 3$ (cercle en ligne pleine), l'élément serait affecté à la classe des triangles rouges car il y a deux triangles et seulement un carré dans le cercle considéré. Si $k = 5$ (cercle en ligne pointillée), l'élément serait affecté à la première classe (carrés bleus) car il y a trois carrés et deux triangles dans le cercle externe.

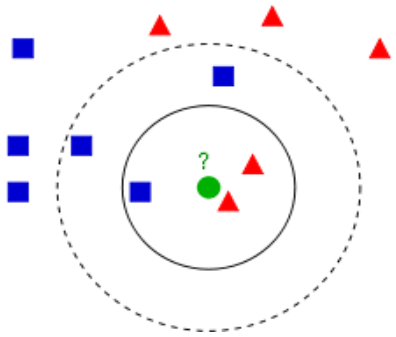


FIGURE 1.1 – Exemple du principe de la classification par KNN

1.3.1.2 Mesures de similarité

L'algorithme KNN nécessite une fonction de calcul de distance entre deux observations. Plusieurs mesures sont disponibles, telles que la distance euclidienne, la distance de Manhattan, la distance de Minkowski, la distance de Jaccard et la distance de Hamming,etc. Le choix de la fonction dépend du type de données manipulées. Par exemple, pour des données quantitatives du même type telles que le poids, les salaires, la taille ou le montant d'un panier électronique, la distance euclidienne est souvent appropriée [5]. Quant à la distance de Manhattan est préférable

lorsque les données en entrée ne sont pas du même type, comme l'âge, le sexe, la longueur et le poids. Il est généralement inutile de coder ces fonctions soi-même, car les bibliothèques de machine learning comme Scikit Learn incluent généralement des implémentations efficaces de ces calculs. Il suffit simplement d'indiquer la mesure de distance souhaitée lors de l'utilisation de ces bibliothèques [6].

Voici les définitions mathématiques de quelques distances qu'on vient d'évoquer :

- **Distance euclidienne :** Distance qui calcule la racine carrée de la somme des différences carrées entre les coordonnées de deux points x et y :

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

- **Distance Manhattan :** Calcule la somme des valeurs absolues des différences entre les coordonnées de deux points :

$$Dm(x,y) = \sum_{i=1}^{n} |x_i - y_i|$$

- **Distance Minkowski :** La distance entre deux points donnés est la différence maximale entre leurs coordonnées sur une dimension :

$$d(x,y) = (\sum_{i=1}^{n} |x_i - y_i|^c)^{\frac{1}{c}}$$

  Notez bien qu'il existe d'autres distances selon le cas d'utilisation de l'algorithme, mais la distance euclidienne reste la plus utilisée.

1.3.1.3 Comment choisir la valeur K ?

Le choix de la valeur de K dans l'algorithme des K plus proches voisins (KNN) dépend du jeu de données spécifique. En général, utiliser un petit nombre de voisins (un K faible) peut conduire à un sous-apprentissage, ce qui signifie que le modèle peut ne pas capturer suffisamment la complexité des données. Par ailleurs plus en utiliser un grand nombre de voisins (un K élevé) peut rendre les prédictions plus fiables. Cependant, si K est trop grand, par exemple égal au nombre total d'observations dans le jeu de données (K=N), cela peut conduire à un sur-apprentissage, où le modèle s'adapte trop étroitement aux données d'entraînement et généralise

mal sur de nouvelles observations [1].

1.3.1.4 Avantages [1][7]

- Simplicité et apprentissage rapide.
- Bonnes performances en général.
- Méthode facile à comprendre.

1.3.1.5 Inconvénients[1][7]

- Prédiction lente : nécessité de revoir tous les exemples à chaque fois.
- Gourmande en place mémoire : peut poser des problèmes avec de grands ensembles.
- Sensible aux attributs non pertinents et corrélés : peut affecter la qualité des prédictions.

1.3.2 La méthode de classification « arbre de décision (AD)»

Les arbres de décision sont une méthode efficace d'apprentissage supervisé utilisée pour classifier des données en fonction de leurs caractéristiques. Cette technique consiste à construire un arbre à partir d'un ensemble de données d'entraînement, où chaque nœud interne représente un test sur une caractéristique des données et chaque feuille représente une décision de classification. Ce modèle offre une grande facilité d'interprétation et d'explication grâce à sa représentation graphique : chaque chemin de la racine à une feuille indique les décisions prises. En outre, les arbres de décision peuvent être utilisés pour évaluer des actions potentielles en fonction de coûts, probabilités et bénéfices, permettant ainsi de déterminer des choix optimaux soit de manière visuelle, soit à travers des algorithmes formels [8].

1.3.2.1 Principe de la construction [1] [2]

Les arbres de décision sont des structures où chaque feuille représente une valeur de la variable-cible, et chaque embranchement correspond à une combinaison de variables d'entrée. Le processus de construction commence en plaçant tous les points de la base d'apprentissage dans le nœud racine, puis divise récursivement chaque nœud en fonction de la valeur d'un attribut testé à chaque étape. L'objectif est d'obtenir des sous-ensembles d'exemples contenant principalement des exemples appartenant à la même classe. Cela conduit à une construction top-down de l'arbre, de la racine vers les feuilles.

Algorithm 2: Pseudo-Algorithme de Construction d'un Arbre de Décision

1 **Début :**
- Initialiser l'arbre courant à l'arbre vide : la racine est le nœud courant;

Répéter :
- Décider si le nœud courant est terminal;
  - **Si** le nœud est terminal alors lui affecter une classe;
  - **Sinon** sélectionner un test créer autant de nouveaux nœuds fils qu'il y a de réponses possibles au test;
- Passer au nœud suivant non exploré s'il en existe jusqu'à un arbre de décision;

fin

Dans cet exemple (voir figure 1.2), le nœud racine est représenté par la condition "température > 37 °C". Les feuilles de l'arbre correspondent aux différentes classes ou décisions, telles que "malade" ou "sain". Les nœuds intermédiaires, tels que "température > 37 °C" et "toux", sont appelés attributs ou variables [4].

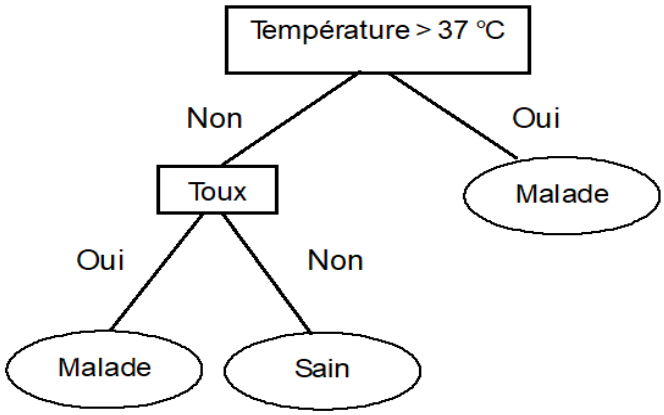


FIGURE 1.2 – Exemple d'un arbre de décision.

1.3.2.2 Les mesures de sélection d'attributs

Les mesures de sélection d'attributs sont cruciales pour choisir les attributs à chaque niveau de l'arbre. Différentes mesures sont utilisées, telles que l'indice d'impureté de Gini, le gain d'information et la réduction de la variance. Ces mesures aident à décider quelle est la meilleure façon de répartir les données à chaque niveau de l'arbre pour maximiser sa performance [1][9].

1. **L'indice de Gini** mesure la probabilité qu'un élément choisi au hasard dans un nœud soit mal classé s'il était classé aléatoirement en fonction de la distribution des classes

dans le nœud.

$$Gini\ (P) = 1 - \sum_{k=1}^{c} (P(k/P))^2$$

2. **L'entropie** mesure le désordre dans un ensemble de données. Un ensemble parfaitement homogène a une entropie de zéro, tandis qu'un ensemble avec une répartition égale de toutes les classes à une entropie maximale.

$$Entropie\,(p) = -\sum_{k=1}^{c} P(k/p)\log(k/p)$$

3. **Le gain d'information** mesure la réduction de l'entropie obtenue en divisant les données selon une caractéristique particulière. Il représente la quantité d'information gagnée en divisant les données par rapport à cette caractéristique.

$$Gain\,(p,i) = i\,(p) - \sum_{j=1}^{n} pj.i(pj)$$

1.3.2.3 Choix de la bonne taille de l'arbre

En pratique, choisir la bonne taille pour un arbre de décision est crucial. Un arbre trop complexe, avec de nombreuses branches et feuilles correspondant à des sous-ensembles parfaitement homogènes, peut-être trop spécifique aux données d'entraînement. Cela peut entraîner une mauvaise généralisation du modèle à de nouvelles données, limitant ainsi sa capacité à rendre compte de la réalité que l'on cherche à modéliser. Il est donc important de trouver un équilibre entre la complexité de l'arbre et sa capacité à généraliser les prédictions à de nouvelles observations [1].

1.3.2.4 Avantages

- Décisions aisément interprétables.
- Classification très rapide.
- Facilité à manipuler des données catégoriques.
- Traitement facile des variables d'amplitudes très différentes.

1.3.2.5 Inconvénients

- La sensibilité au bruit et aux points aberrants.
- La sensibilité au nombre de classes (plus le nombre de classes est grand plus les performances diminuent).

1.3.3 La méthode de classification « classifieur bayésien (CB) »

Parmi les nombreuses approches de classification, les méthodes bayésiennes se distinguent par leur capacité à modéliser et à quantifier l'incertitude, ce qui en fait des outils puissants et polyvalents. La classification bayésienne est une méthode statistique ancienne qui remonte au 17^{e} siècle avec les premiers travaux de la théorie de Bayes, c'est une approche probabiliste basée sur les principes du théorème de Bayes. Cette méthode permet de prédire la probabilité d'appartenance à une classe donnée en se basant sur des probabilités conditionnelles et des données observées.

Au cours des dernières décennies, cette méthode s'est répandue dans de nombreux domaines. L'apprentissage probabiliste est une approche fondamentale en intelligence artificielle, offrant une méthode pratique pour la modélisation et la prédiction des données [10].

1.3.3.1 Principe de construction

Le classifieur bayésien, est un exemple emblématique de l'apprentissage probabiliste, basée sur les probabilités conditionnelles et la règle de Bayes. Ce modèle permet de prédire les classes des données en maximisant la probabilité *a-posteriori*.
Le problème de classification peut être formulé en utilisant les probabilités *a-posteriori* :

$$P(C_k|X) = \text{probabilité que le tuple (instance) } X = \langle x_1, x_2, \ldots, x_n \rangle \text{ est dans la classe } C_k.$$

Exemple : Classer un fruit comme "Pêche" ou "Poire" (**Pêche = classe** C_1 **; Poire = classe** C_2). Soit le Fruit suivant : $X =$ (**Taille**=petite, **Couleur**=jaune, **Saison**=hiver, **Forme**=rond). La question qu'on se pose est est ce que ce fruit est une poire ou bien une pêche ? Nous devons calculer ces deux probabilités :

$$P(\text{Classe=Pêche} \mid \text{Taille=petite, Couleur=jaune, Saison=hiver, Forme=rond})$$

$$P(\text{Classe=Poire} \mid \text{Taille=petite, Couleur=jaune, Saison=hiver, Forme=rond})$$

Affecter à une instance X la classe C_k telle que $P(C_k|X)$ soit maximale.

1.3.3.2 Théorème de Bayes

Supposons que vous ayez un échantillon de données et une nouvelle donnée X dont la classe est inconnue. Vous souhaitez déterminer sa classe. Vous définissez une hypothèse C (par exemple, "X appartient à la classe C") et vous cherchez à calculer la probabilité $P(C|X)$, qui représente la probabilité que C soit vérifiée après avoir observé X [11].

$P(C|X)$ est ce qu'on appelle la probabilité *a posteriori*, c'est-à-dire la probabilité après avoir pris en compte l'observation de X, tandis que $P(C)$ est la probabilité *a priori*, qui représente la probabilité que C soit vérifiée pour n'importe quel exemple de données.
Le calcul de la probabilité *a posteriori* se fait selon le théorème de Bayes, comme suit :

$$P(C|X) = \frac{P(X|C) \cdot P(C)}{P(X)}$$

$P(C|X)$ est la probabilité conditionnelle, $P(X|C)$ est la probabilité de X sachant C, $P(C)$ est la probabilité *a priori* de C, et $P(X)$ est la probabilité marginale de X. Ce calcul permet de mettre à jour nos croyances sur C après avoir observé les données X.

1.3.3.3 Classificateur Naïve Bayes

L'estimation des probabilités *a posteriori* est cruciale. Le théorème de Bayes est une pierre angulaire de cette approche.

$$P(C_k|X) = \frac{P(X|C_k)P(C_k)}{P(X)}$$

- $P(X)$ est une constante pour toutes les classes.
- $P(C_k)$ représente la fréquence relative des instances de la classe C_k.
- Le calcul de $P(X|C_k)$ est généralement inabordable, posant ainsi un défi majeur.

Pour surmonter les limitations de calcul associées à $P(X|C_k)$, la classification bayésienne naïve propose une solution pragmatique. Cette approche repose sur l'hypothèse naïve d'indépendance des attributs, permettant ainsi de simplifier le calcul [11]

La classification Naïve Bayes est décrite par l'expression suivante :

$$P(X \mid C_k) = P(x_1, \ldots, x_n \mid C_k) = P(x_1 \mid C_k) \times \cdots \times P(x_n \mid C_k)$$

$P(x_i \mid C_k)$ est estimée en utilisant la fréquence relative des instances ayant la valeur x_i pour l'attribut correspondant dans la classe C_k.

$$P(x_i \mid C_k) = \frac{n_{X_i}}{n_{C_k}}$$

- n_{X_i} : nombre d'instances de la classe C_k qui ont comme valeur x_i pour l'attribut considéré
- n_{C_k} : nombre d'instances de la classe C_k

Pour un exemple $X = (x_1, \ldots, x_n)$ à classer, nous estimons $P(X \mid C_k) \times P(C_k)$ pour chaque classe C_k.
Enfin, nous attribuons à X la classe C_k pour laquelle cette probabilité est maximale, guidant ainsi la décision de classification.

1.3.3.4 Avantages

- Capacité à gérer l'incertitude et la variabilité des données de manière explicite.
- Possibilité d'intégrer des connaissances *a priori* sous forme de distributions sur les paramètres du modèle.
- Facilité d'interprétation des résultats grâce à l'expression des prédictions sous forme de probabilités.

1.3.3.5 Inconvénients

- La qualité et la représentativité des données d'entraînement ont un impact significatif sur la performance du modèle.
- Sensibilité aux hypothèses *a priori*.
- Complexité de modélisation.
- Limitations avec les données non indépendantes.

1.4 Meta classification

La méta-classification, également connue sous le nom de "combinaison de classifieurs", aborde le défi de fusionner les décisions de multiples classifieurs pour obtenir une prédiction finale. C'est une technique avancée de machine learning qui consiste à utiliser plusieurs modèles de classification de base pour améliorer la performance globale de la classification. L'idée principale est de combiner les prédictions de différents classificateurs pour créer un classificateur supérieur, souvent appelé "méta-classificateur". Cette approche tire parti de la diversité des classificateurs de base pour réduire les erreurs individuelles et accroître la robustesse et la précision des prédictions. Un méta-classifieur est simplement un classifieur qui génère une prédiction finale en agrégeant les prédictions des classifieurs de base. Pour ce faire, il utilise ces prédictions comme caractéristiques [12].

1.4.1 Principes de la méta-classification

La méta-classification repose sur la construction de deux niveaux de modèles :

1. **Niveau de Base (Base Level)** : Ce niveau comprend plusieurs modèles de classification de base, appelés "classificateurs de base". Ces modèles peuvent être de différents types, tels que des arbres de décision, des k-plus proches voisins KNN, des réseaux de neurones, des classifieurs bayésiens, etc.
2. **Niveau Méta (Meta Level)** : Ce niveau utilise les prédictions des classificateurs de base comme caractéristiques pour un modèle de classification supplémentaire, appelé "méta-classificateur". Le méta-classificateur apprend à combiner les prédictions des classificateurs de base pour produire la prédiction finale.

1.4.2 Avantages de la méta-classification

- **Amélioration de la précision** : En combinant plusieurs classificateurs, la méta-classification peut réduire les erreurs individuelles et augmenter la précision globale des prédictions.
- **Robustesse** : La diversité des classificateurs de base permet de créer un système plus robuste, moins susceptible de sur-apprendre ou de sous-apprendre.
- **Flexibilité** : Les méta-classificateurs peuvent utiliser n'importe quel type de classificateur de base, ce qui permet une grande flexibilité dans la conception du modèle.

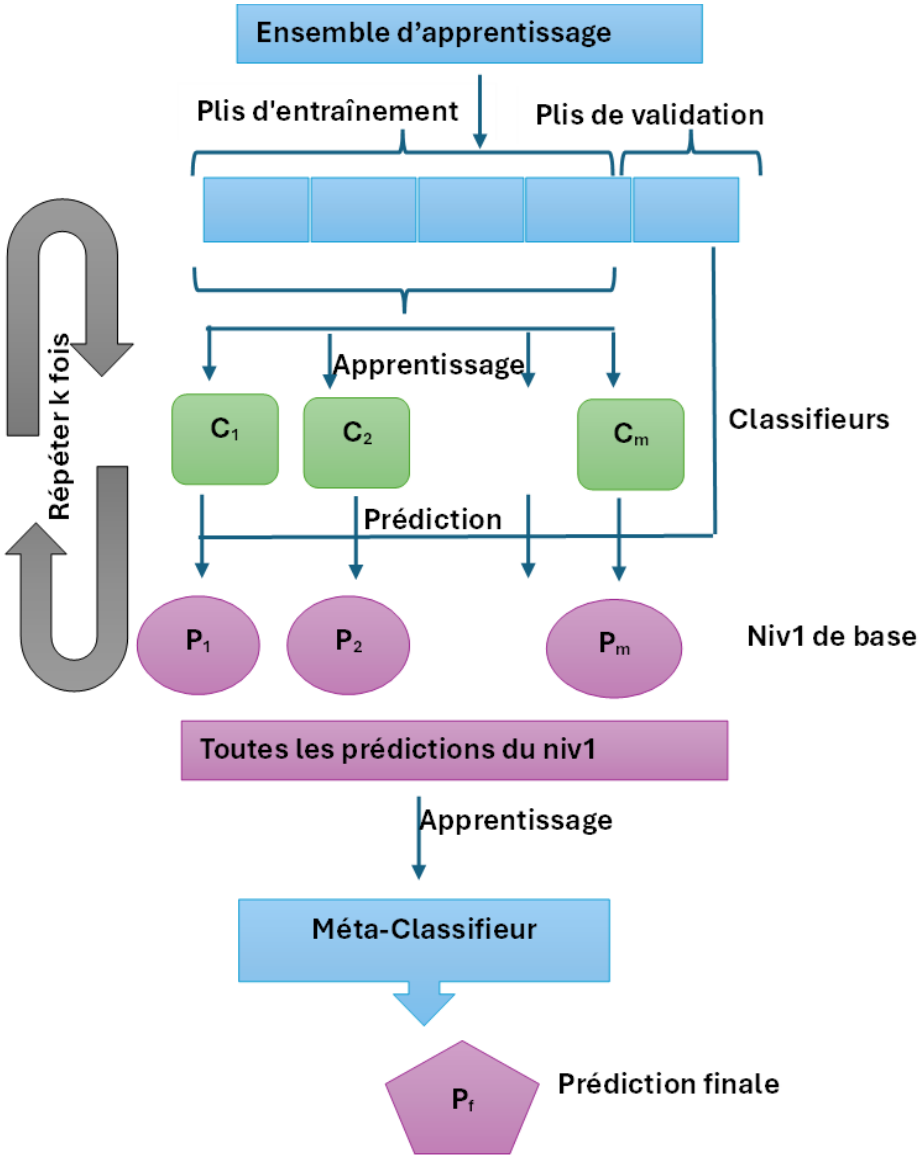


FIGURE 1.3 – Schéma de la méta-classification

Cependant, il est important de noter que la performance de la méta-classification dépend de plusieurs facteurs, tels que la diversité et la qualité des modèles de base, ainsi que la méthode de combinaison des prédictions. Parfois, une méta-classification peut ne pas améliorer la précision si les modèles de base sont trop similaires ou s'ils ont des performances médiocres. Il est donc crucial de choisir judicieusement les modèles de base et les techniques d'ensemble pour obtenir les meilleurs résultats.

1.4.3 Techniques courantes de méta-classification

1.4.3.1 Bagging (Bootstrap Aggregating)

- **Principe :** Bagging consiste à entraîner plusieurs modèles de manière parallèle et indépendante les uns des autres sur des sous-ensembles aléatoires de l'ensemble de données d'entraînement.
- **Processus :** Chaque modèle est entraîné sur un échantillon bootstrap (un échantillon aléatoire avec remplacement de la taille de l'ensemble de données d'origine).
- **Combinaison des prédictions :** Les prédictions des modèles individuels sont combinées en prenant un vote majoritaire (pour la classification) ou en moyennant (pour la régression).
- **Exemple :** Les forêts aléatoires sont un exemple célèbre de l'application de bagging, où de nombreux arbres de décision sont construits et leurs prédictions sont agrégées pour produire une prédiction finale robuste.

1.4.3.2 Boosting

- **Principe :** Le boosting entraîne des modèles de manière séquentielle, en donnant plus de poids aux exemples mal classifiés par les modèles précédents.
- **Processus :** Chaque modèle est construit de manière itérative en ajustant les poids des exemples d'entraînement. Les exemples mal classifiés par les modèles précédents reçoivent une pondération plus élevée.
- **Combinaison des prédictions :** Les prédictions des modèles individuels sont pondérées en fonction de leur performance respective lors de l'agrégation finale.
- **Exemple :** AdaBoost (Adaptive Boosting) est un exemple populaire de boosting, utilisé avec divers classificateurs faibles pour produire un modèle final plus robuste.

1.4.3.3 Stacking (Stacked Generalization)

- **Principe :** Le stacking combine les prédictions de plusieurs modèles de base en utilisant un méta-classificateur qui apprend à pondérer ces prédictions.
- **Processus :** Plusieurs modèles de base sont entraînés sur l'ensemble de données. Les prédictions de ces modèles sont ensuite utilisées comme caractéristiques d'entrée pour un méta-classificateur.

- **Combinaison des prédictions :** Le méta-classificateur prend les prédictions des modèles de base comme entrées et apprend à combiner ces prédictions pour produire une prédiction finale.
- **Exemple :** Par exemple, vous pouvez utiliser des modèles tels que SVM, réseaux de neurones, arbres de décision comme modèles de base, puis utiliser une régression logistique ou un autre modèle pour combiner leurs prédictions.

En résumé, ces techniques de méta-classification visent toutes à améliorer la performance des modèles en exploitant la diversité et la complémentarité des modèles individuels. Bagging se concentre sur la parallélisation et l'agrégation des prédictions, Boosting sur la séquentialisation et l'accentuation sur les exemples difficiles, tandis que Stacking utilise un méta-classificateur pour apprendre à combiner intelligemment les prédictions des modèles de base. En choisissant la bonne stratégie d'agrégation des prédictions, les méta-classifieurs peuvent améliorer la précision et la robustesse des prédictions finales, ce qui en fait des outils puissants en apprentissage automatique. Le choix de la stratégie dépend souvent de la nature des données, du problème à résoudre et de la performance relative des classifieurs de base [12].

1.5 Conclusion

Dans ce chapitre, nous avons exploré le processus de fouille de données, mettant en lumière l'importance de l'extraction de connaissances à partir de vastes ensembles de données. Nous nous sommes particulièrement concentrés sur la classification supervisée, une tâche centrale de la fouille de données qui vise à construire des modèles prédictifs à partir d'exemples étiquetés. Trois techniques d'apprentissage supervisé ont été examinées en détail : les K plus proches voisins (KNN), les arbres de décision (AD) et le classifieur bayésien (CB). Chacune de ces méthodes présente ses propres avantages et inconvénients en termes de facilité d'interprétation, de précision des prédictions, de performance et de sensibilité aux données.

En outre, nous avons abordé la méta-classification, une approche qui combine les prédictions de multiples classificateurs pour améliorer la robustesse et la précision des modèles prédictifs. Cette technique se révèle particulièrement utile dans des applications complexes où une seule méthode de classification pourrait ne pas suffire.

Ces techniques représentent des outils puissants pour la construction de modèles prédictifs,

notamment dans des domaines comme le diagnostic médical. Choisir la méthode appropriée dépendra des caractéristiques spécifiques des données et des objectifs visés. En combinant une compréhension approfondie de ces techniques avec des études de cas pratiques, nous pouvons exploiter pleinement le potentiel de la fouille de données pour prendre des décisions éclairées et anticiper les tendances futures dans divers domaines d'application.

Chapitre 2

La sélection d'attributs (caractéristiques)

2.1 Introduction

La sélection d'attributs, également connue sous le nom de sélection de caractéristiques (variables) ou de *feature selection* en anglais, est un processus de prétraitement de données. Elle consiste à choisir parmi un ensemble de caractéristiques de grande taille un sous-ensemble de caractéristiques intéressantes et pertinentes pour un problème donné. L'objectif principal de la sélection d'attributs est de réduire la dimensionnalité des données en éliminant les attributs redondants, non informatifs ou bruités, tout en conservant les caractéristiques les plus importantes et les plus discriminantes pour une tâche d'apprentissage spécifique. Cette approche est largement utilisée dans divers domaines, de l'apprentissage automatique à la bioinformatique, en passant par la vision par ordinateur et le traitement d'images. En effet, les attributs choisis sont ceux considérés comme pertinents pour la tâche de classification. Cette pertinence peut être évaluée selon plusieurs critères, distinguant les attributs très pertinents, les attributs peu pertinents mais potentiellement améliorants, et les attributs non pertinents [13].

La sélection d'attributs joue un rôle crucial dans le domaine médical, en particulier lorsqu'il s'agit de diagnostiquer des maladies. C'est un outil essentiel pour identifier les caractéristiques les plus pertinentes parmi les données des patients, telles que les symptômes, les antécédents médicaux et les résultats de tests, afin de permettre des diagnostics précis et une prise en charge adaptée. La sélection d'attributs permet de choisir les caractéristiques les plus pertinentes pour la classification des maladies, aidant ainsi les professionnels de la santé à établir des diagnostics

précis et à prendre des décisions éclairées sur le traitement.

Le processus général de sélection d'attributs comprend plusieurs étapes, allant de l'évaluation des attributs aux procédures utilisées pour les sélectionner. Dans ce chapitre, nous explorerons les différentes méthodes de sélection d'attributs, en nous concentrant sur les catégories principales telles que les approches “Filtre”, “Wrapper” et “Embedded”. Nous commencerons par définir la sélection d'attributs et identifier les problèmes associés, avant d'examiner en détail ses avantages et les difficultés rencontrées. Enfin, nous conclurons en présentant une vue d'ensemble des méthodes de sélection d'attributs et en soulignant leur importance dans la modélisation des données et la prise de décision.

2.2 Difficultés de la sélection

2.2.1 Dimensionnalité

Avec l'avènement des données de grande dimension, ou Big Data, la sélection d'attributs est devenue cruciale. La croissance rapide de l'espace des variables rend les données éparses et éloignées, un phénomène connu sous le nom de "fléau de la dimension". La complexité des algorithmes augmente, et même si certains attributs sont pertinents individuellement, leur combinaison peut apporter peu de gains. Les méthodes statistiques traditionnelles produisent souvent des résultats biaisés dans ces conditions. Il est donc essentiel de réduire la dimensionnalité pour permettre aux algorithmes d'apprentissage de tirer des informations utiles et compréhensibles [14, 15].

2.2.2 Pertinence d'attributs

La qualité de la classification dépend de la pertinence des informations, non de leur quantité. Les variables pertinentes contiennent la structure d'intérêt des observations. Plusieurs définitions de la pertinence existent, mais elles convergent vers l'idée que les attributs pertinents sont ceux dont les valeurs changent systématiquement selon les valeurs de classe [16, 17].

2.2.3 Redondance

La redondance des attributs réfère à la présence d'informations dupliquées ou superflues. Deux attributs sont redondants s'ils fournissent les mêmes informations. La redondance est souvent mesurée par la corrélation entre les attributs. Identifier et éliminer les attributs redondants permet de simplifier les modèles et d'améliorer leur interprétation [18, 19].

2.3 Avantages et Inconvénients de la sélection d'attributs

La sélection d'attributs présente plusieurs avantages et inconvénients [18, 19, 13] :

2.3.1 Avantages

- **Réduction du nombre d'attributs :** En éliminant les attributs non pertinents, redondants ou bruités, la sélection d'attributs simplifie le modèle et améliore la pertinence des données.
- **Amélioration de la précision et des performances :** La qualité des données d'entrée du modèle de classification est améliorée, conduisant à des modèles plus simples et plus efficaces.
- **Réduction du temps de calcul :** La complexité réduite du modèle diminue le temps de calcul nécessaire pour entraîner le modèle ou effectuer des prédictions.
- **Interprétation simplifiée :** Les attributs conservés sont directement liés aux phénomènes d'intérêt, facilitant l'interprétation des résultats du modèle.

2.3.2 Inconvénients

- **Perte d'informations :** Éliminer certains attributs peut entraîner la perte d'informations potentiellement importantes.
- **Complexité supplémentaire :** Certains algorithmes de sélection d'attributs peuvent ajouter une complexité supplémentaire au processus d'analyse des données.
- **Sensibilité aux critères de sélection :** Le choix du critère de sélection peut influencer les attributs finalement sélectionnés, rendant le processus potentiellement biaisé.
- **Surajustement :** Une sélection inappropriée peut entraîner un surajustement du modèle aux données d'apprentissage, compromettant ses performances sur de nouvelles

données.

- **Dépendance aux méthodes de sélection :** Les performances dépendent fortement des méthodes spécifiques utilisées, et aucune méthode unique ne convient à toutes les situations.

2.4 Processus de sélection d'attributs

La sélection d'attributs suit généralement quatre étapes principales [13], comme illustré dans la figure suivante :

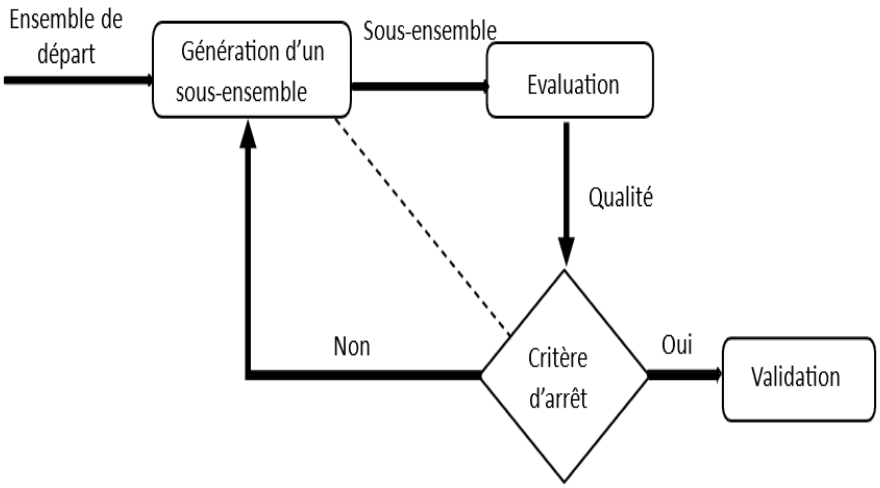


FIGURE 2.1 – Procédure générale pour la sélection d'attributs

1. **Génération du sous-ensemble :** Cette étape consiste à créer un sous-ensemble de caractéristiques à partir de l'ensemble initial d'attributs disponibles.
2. **Évaluation du sous-ensemble :** Une fois que le sous-ensemble d'attributs a été généré, il est évalué pour déterminer sa qualité par rapport à certains critères prédéfinis.
3. **Critère d'arrêt :** Cette étape définit les conditions pour arrêter le processus de sélection d'attributs.
4. **Validation des résultats :** Enfin, les résultats de la sélection d'attributs doivent être validés pour s'assurer de leur généralisation à de nouvelles données.

Cette approche en quatre étapes offre un cadre systématique pour mener efficacement la sélection d'attributs [20, 21].

2.4.1 La procédure de génération de sous-ensembles

La procédure de génération dans la sélection d'attributs est une étape critique où l'on explore l'espace des combinaisons d'attributs pour construire différents sous-ensembles. C'est un processus heuristique qui vise à identifier un sous-ensemble candidat dans l'espace de recherche, puis à l'évaluer [22].

Différentes approches sont utilisées pour cette exploration. L'approche ascendante commence avec un ensemble vide de caractéristiques et ajoute à chaque itération la variable optimale suivant un certain critère au sous-ensemble. La stratégie descendante, quant à elle, commence avec l'ensemble complet de toutes les variables et supprime à chaque itération une variable du sous-ensemble. Enfin, l'approche bidirectionnelle combine à la fois l'ajout et la suppression de variables à différentes étapes du processus de sélection, offrant ainsi une flexibilité supplémentaire [23].

En ce qui concerne la stratégie de recherche, elle peut être complète, heuristique ou aléatoire. La recherche complète effectue une exploration exhaustive du sous-ensemble optimal par rapport à la fonction d'évaluation choisie, tandis que la recherche heuristique utilise des algorithmes itératifs pour sélectionner ou rejeter des caractéristiques. La recherche aléatoire, quant à elle, sélectionne un sous-ensemble initial de manière aléatoire, puis poursuit la recherche en effectuant un nombre défini d'itérations [13, 22].

2.4.2 La procédure d'évaluation

L'évaluation des attributs est une phase cruciale du processus de sélection, où les caractéristiques sont évaluées en fonction de leur pertinence pour le modèle prédictif. Cette phase est généralement divisée en trois catégories principales : "Filter", "Wrapper" et "Embedded" [23, 20, 22, 24].

Les **méthodes de filtre (Filter methods)** évaluent la pertinence des attributs en examinant uniquement leurs propriétés intrinsèques, indépendamment du classificateur utilisé. C'est une étape de prétraitement (filtrage) avant le processus d'apprentissage. Elle calcule généralement un score de pertinence pour chaque attribut, puis élimine les attributs ayant un score faible. Le sous-ensemble optimal d'attributs obtenu par cette technique est ensuite utilisé en entrée de l'algorithme de classification.

Les **méthodes d'enveloppement (Wrapper methods)** évaluent un sous-ensemble d'attributs en utilisant directement un algorithme de classification. Elles sont généralement basées sur un algorithme de recherche pour explorer l'espace des solutions et un algorithme de classification pour évaluer les sous-ensembles sélectionnés. Cette approche conduit généralement à une précision plus élevée, mais elle est également plus coûteuse en termes de temps et de ressources.

Les **méthodes embarquées (Embedded methods)** effectuent la sélection d'attributs simultanément avec la procédure de classification. Le sous-ensemble optimal d'attributs est déterminé pendant l'apprentissage du classificateur. Tout comme les méthodes wrapper, les méthodes embarquées sont spécifiques à un algorithme d'apprentissage donné. Cependant, elles sont généralement plus rapides que les méthodes wrapper car la sélection d'attributs est intégrée dans le processus d'apprentissage du classificateur.

Chaque méthode a ses propres avantages et inconvénients, et le choix entre elles dépend souvent du contexte spécifique du problème de sélection d'attributs et des ressources disponibles.

2.4.3 Le critère d'arrêt

Le critère d'arrêt dans le processus de sélection de caractéristiques est essentiel pour déterminer quand arrêter les itérations de la sélection. Ce critère peut être influencé par la procédure de génération des sous-ensembles et par la fonction d'évaluation utilisée. Certains critères d'arrêt basés sur la procédure de génération peuvent inclure :

- **Atteinte d'un seuil prédéfini :** Cela peut être un seuil tel que le nombre minimal ou maximal d'attributs dans le sous-ensemble sélectionné, ou le nombre maximal d'itérations permises.
- **Absence d'amélioration de la précision :** La sélection s'arrête lorsque le sous-ensemble actuel ne peut plus être amélioré en termes de précision. En d'autres termes, il n'y a plus de possibilité de trouver un sous-ensemble qui soit meilleur.

Certains critères d'arrêt sont basés sur l'ordre des attributs classés selon un score de pertinence. Dans ce cas, les attributs ayant les scores les plus élevés sont sélectionnés pour être utilisés par un classificateur (méthode "filtre").

2.4.4 Procédure de validation

La validation des résultats est essentielle pour garantir que les attributs sélectionnés améliorent les performances du modèle et généralisent bien à de nouvelles données. Elle peut être réalisée de différentes manières, notamment par l'utilisation de techniques de validation croisée où les données sont divisées en ensembles pour évaluer la stabilité des attributs sélectionnés. Une autre approche consiste à diviser les données en ensembles d'entraînement et de test indépendants pour évaluer la capacité de généralisation du modèle. Dans certains cas, lorsque les variables pertinentes sont connues à l'avance, la validation peut être plus directe. En résumé, la validation des résultats assure que les attributs sélectionnés améliorent les performances du modèle et sont robustes à différentes partitions des données, contribuant ainsi à la fiabilité et à la généralisabilité du modèle final.

2.5 Revue des méthodes de sélection d'attributs

Les techniques de sélection d'attributs peuvent être regroupées en deux grandes catégories selon leur approche d'évaluation : les méthodes enveloppantes (**wrapper**), les méthodes filtrantes (**filter**) et les méthodes embarquées (**embedded**).

2.5.1 Méthodes filtrantes

Les méthodes filtrantes sélectionnent des variables avant le processus d'apprentissage en évaluant leur pertinence par rapport à la variable cible. Voici quelques-unes des principales méthodes :

2.5.1.1 Méthode de sélection basée sur le test du chi2

Cette méthode utilise le test du chi-carré pour évaluer la dépendance entre chaque caractéristique et la variable cible. Elle sélectionne un pourcentage des caractéristiques les plus importantes en fonction de leur score de chi2. Cette approche est particulièrement adaptée pour les caractéristiques catégorielles et s'avère efficace pour identifier les attributs les plus pertinents au sein de grands ensembles de données. Elle est implémentée en Python et est largement utilisée dans le domaine de la sélection d'attributs avec le nom de **percentile** [25].

2.5.1.2 Méthode de sélection d'attributs avec chi2 en mode k_best

La méthode GenericUnivariateSelect offre une flexibilité dans la sélection d'attributs en permettant de choisir parmi plusieurs modes de sélection. En utilisant chi2 comme fonction de score et en spécifiant le mode k_bestcette méthode identifie les k meilleures caractéristiques en fonction de leur score de chi2. Ce mode k_best est particulièrement utile dans les cas où l'on souhaite contrôler précisément le nombre d'attributs sélectionnés, ce qui est bénéfique pour réduire la dimensionnalité des données. Le test chi2 est approprié pour évaluer la dépendance entre les caractéristiques catégorielles et la variable cible, en en faisant un choix pertinent pour la sélection d'attributs dans de telles situations. Elle est implémentée en Python et est largement utilisée dans le domaine de la sélection d'attributs avec le nom de **GenericUnivariateSelect** [26].

2.5.1.3 Sélection des meilleures caractéristiques avec ANOVA

La méthode utilise le test d'ANOVA (Analysis of Variance) pour évaluer la relation entre chaque caractéristique et la variable cible. En spécifiant f_classif comme fonction de score, cette méthode sélectionne les k meilleures caractéristiques en fonction de leur score d'ANOVA. Le test d'ANOVA mesure la variation entre les moyennes des différentes classes par rapport à la variation à l'intérieur des classes. Les caractéristiques avec les scores d'ANOVA les plus élevés sont considérées comme les plus pertinentes pour prédire la variable cible. SelectKBest est particulièrement adapté aux caractéristiques continues et peut être utilisé pour sélectionner un nombre spécifique de caractéristiques en fonction de leurs contributions à la variance de la variable cible. Elle est implémentée en Python et est largement utilisée dans le domaine de la sélection d'attributs avec le nom de **SelectKBest** [27].

2.5.1.4 Relief

La méthode Relief calcule une mesure globale de pertinence des caractéristiques en comparant les distances entre des exemples d'apprentissage choisis aléatoirement et leurs plus proches voisins de la même classe et de l'autre classe. Malgré sa simplicité et sa précision sur des données bruitées, elle peut manquer de cohérence dans ses résultats et ne prend pas en compte la corrélation entre les caractéristiques [21, 28].

2.5.1.5 Best First Search

Best First Search est une stratégie de recherche qui explore l'espace de recherche en ajoutant itérativement les caractéristiques les plus pertinentes. Elle peut revenir en arrière à un sous-ensemble précédent plus prometteur si nécessaire, et elle est généralement considérée comme offrant de bons résultats pour la sélection de caractéristiques [21].

2.5.1.6 FOCUS

L'algorithme FOCUS effectue une recherche exhaustive sur l'ensemble initial des caractéristiques en évaluant tous les sous-ensembles possibles. L'inconvénient majeur de cette méthode est sa sensibilité au bruit dans sa méthode d'évaluation, ainsi que le temps de calcul qui devient important lorsque la taille des caractéristiques est grande [21].

2.5.1.7 Branch & Bound

L'algorithme Branch & Bound commence avec toutes les caractéristiques et en enlève une à la fois en évaluant chaque nœud. Une variante de cet algorithme, appelée ABB, utilise une mesure monotone pour garantir une sélection de caractéristiques efficace [13].

2.5.1.8 Las Vegas Filter (LVF)

L'algorithme Las Vegas Filter (LVF) utilise un critère d'incohérence pour éliminer les sous-ensembles de caractéristiques non pertinents. LVF garantit l'obtention du sous-ensemble optimal mais peut être plus lent que certaines approches heuristiques en raison du temps nécessaire à la génération de résultats [29]

2.5.2 Les méthodes enveloppantes (Wrapper)

Les méthodes enveloppantes utilisent l'algorithme d'induction comme une boîte noire : elles effectuent l'apprentissage avec les variables sélectionnées et estiment les performances à partir de l'erreur de généralisation.

2.5.2.1 Sequential Forward Selection (SFS)

La méthode de Sélection Avant Séquentielle (SFS) est une approche heuristique de recherche. Elle commence avec un ensemble vide et ajoute successivement une caractéristique jusqu'à ce que le critère d'arrêt soit satisfait. Cette méthode était utilisée pour réduire la taille des données et améliorer les résultats de classification [20]. Voici l'algorithme de cette méthode :

Algorithm 3: Pseudo-algorithme de Sélection de caractéristiques par SFS

Entrée: Ensemble initial de caractéristiques
Sortie : Ensemble final de caractéristiques sélectionnées

```
Initialiser un ensemble vide de caractéristiques sélectionnées;
while le critère d'arrêt n'est pas satisfait do
    for chaque caractéristique non sélectionnée do
        Ajouter la caractéristique à l'ensemble sélectionné;
        Évaluer les performances du modèle avec l'ensemble actuel de
         caractéristiques;
    end
    Sélectionner la caractéristique qui maximise les performances du modèle;
end
Retourner l'ensemble final de caractéristiques sélectionnées;
```

2.5.2.2 Sequential Backward Selection (SBS)

La méthode de Sélection Arrière Séquentielle (SBS) consiste à commencer avec l'ensemble complet de toutes les caractéristiques, puis à procéder à la suppression successive de ces caractéristiques. Bien que cette technique soit plus performante que la précédente SFS, son principal inconvénient réside dans son temps de calcul plus important [20].

Algorithm 4: Pseudo-algorithme de Sélection d'attributs par SBS

Entrée: Ensemble initial de caractéristiques

Sortie : Ensemble final de caractéristiques sélectionnées

```
Initialiser l'ensemble avec toutes les caractéristiques;
while critère d'arrêt non satisfait do
    foreach caractéristique dans l'ensemble do
        Supprimer la caractéristique de l'ensemble;
        Évaluer les performances du modèle avec l'ensemble restant de
         caractéristiques;
    end
    Sélectionner l'ensemble qui maximise les performances du modèle;
end
Retourner l'ensemble final de caractéristiques sélectionnées;
```

2.5.2.3 L'algorithme des essaims de lucioles (Firefly algorithm)

L'algorithme des essaims de lucioles, inspiré par le comportement lumineux des lucioles, est une technique d'optimisation globale basée sur la métaheuristique. Les lucioles, dans cet algorithme, représentent des solutions potentielles qui se déplacent dans l'espace de recherche en suivant certaines règles d'attraction et de luminosité [30].

Algorithm 5: Pseudo-algorithme pour l'algorithme des lucioles

Entrée: Population initiale de lucioles avec positions et intensités aléatoires

Sortie : Position de la luciole la plus lumineuse (solution optimale)

```
Initialiser la population de lucioles avec des positions et des intensités aléatoires;
while critère d'arrêt non satisfait do
    foreach paire de lucioles do
        Calculer l'intensité de chaque luciole en fonction de sa distance avec les autres;
        Mettre à jour la position de chaque luciole en suivant les règles d'attraction et
         de déplacement;
    end
end
Retourner la position de la luciole la plus lumineuse comme solution optimale;
```

2.5.2.4 Optimisation par essaims de particules (PSO)

L'optimisation par essaim de particules repose sur un ensemble d'individus, appelés particules, initialement répartis de manière aléatoire et homogène dans l'espace de recherche. Chaque particule est une solution potentielle et possède une mémoire contenant sa meilleure solution visitée jusqu'à présent. Elle a la capacité de communiquer avec les particules voisines. Avec ces informations, chaque particule ajuste sa position en suivant une tendance qui combine sa volonté de retourner vers sa meilleure solution locale et son mimétisme par rapport aux solutions de son voisinage. En combinant les optimums locaux et empiriques, l'ensemble des particules converge généralement vers la solution optimale globale du problème traité [31].

Algorithm 6: Pseudo-algorithme pour l'algorithme d'optimisation par essaim de particules

Entrée: Ensemble de particules avec positions et vitesses aléatoires

Sortie : Meilleure solution trouvée parmi toutes les particules (solution optimale)

```
1 Initialiser un ensemble de particules avec des positions et des vitesses aléatoires;
2 while critère d'arrêt non satisfait do
3     foreach particule do
4         Mettre à jour la vitesse et la position de la particule en fonction de sa meilleure
            solution locale et globale;
5     end
6 end
7 Retourner la meilleure solution trouvée comme solution optimale;
```

2.6 Conclusion

La sélection d'attributs est un processus essentiel dans la modélisation des données, visant à choisir les caractéristiques les plus pertinentes pour une tâche d'apprentissage spécifique tout en réduisant la dimensionnalité des données. Ce chapitre a abordé les défis associés à cette pratique, tels que la dimensionnalité croissante des données et la pertinence des attributs. Différentes méthodes, telles que les approches "Filtre", "Wrapper" ont été présentées pour relever ces défis. En comprenant l'importance de choisir judicieusement les attributs et en explorant les diverses méthodes disponibles, les praticiens peuvent améliorer la qualité de leurs modèles et prendre des décisions plus éclairées dans leurs domaines respectifs.

Chapitre 3

Conception, Réalisation et Modélisation

3.1 Introduction

L'apprentissage automatique a révolutionné la manière dont les données médicales sont utilisées dans le domaine des soins de santé, offrant des outils sophistiqués pour la prise de décisions et la prévision basée sur de vastes ensembles de données. Parmi les défis médicaux majeurs, les maladies du foie occupent une place prépondérante en raison de leur impact significatif sur la santé publique mondiale. Dans cette étude, nous avons exploré la prédiction des différents stades de cette maladie (nous avons quatres stades), en utilisant trois techniques d'apprentissage automatique : Le plus proche voisin (KNN), les arbres de décision (AD) et le classifieur bayésien (CB) sur les données médicales de la « Mayo Clinic Primary Biliary Cirrhosis Data ». Ceci est shématisé par la figure 3.1

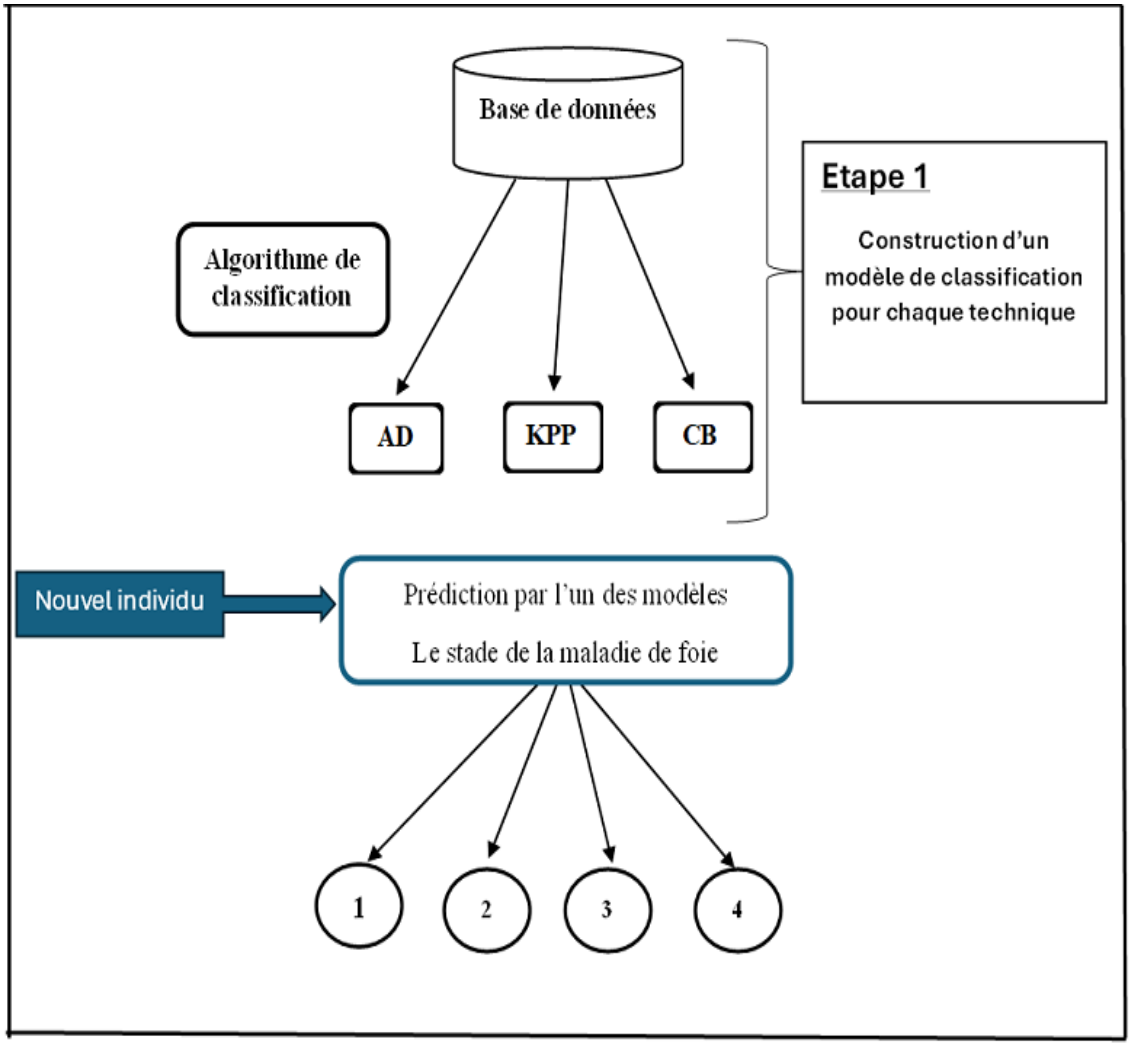


FIGURE 3.1 – Schéma de la première étape de construction des modèles de classification.

Notre objectif principal est l'amélioration de la prédiction. Pour parvenir à des modèles de prédiction plus efficaces, nous avons d'abord procédé à une sélection minutieuse des caractéristiques pertinentes en utilisant la technique de sélection d'attributs. Cette étape, vise à identifier les attributs les plus informatifs dans notre ensemble de données. Nous avons utilisé deux approches principales pour cette sélection : le filtrage et l'enveloppement avec plusieurs algorithmes. Le filtrage implique l'utilisation de mesures statistiques pour évaluer l'importance des caractéristiques indépendamment du modèle d'apprentissage, tandis que l'enveloppement utilise le modèle lui-même pour évaluer la qualité des caractéristiques en fonction de leur contribution à la performance globale du modèle. Ceci est shématisé par la figure 3.2

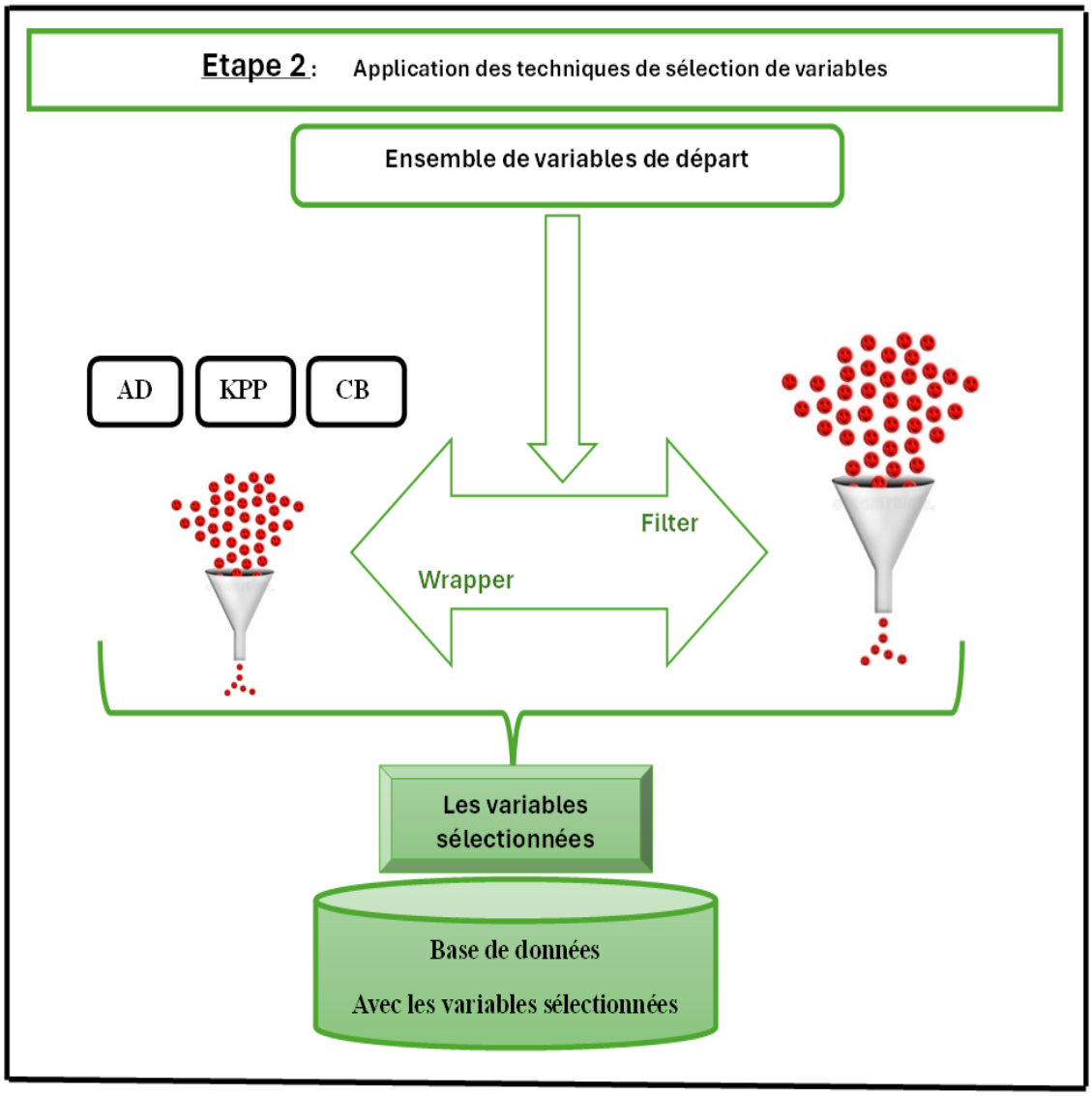


FIGURE 3.2 – Schéma de la deuxième étape d'utilisation des techniques de sélection d'attributs filter et wrapper.

Après avoir sélectionné les caractéristiques les plus pertinentes, nous avons entraîné une nouvelle fois nos modèles KNN, AD et le CB sur les données médicales de la « Mayo Clinic Primary Biliary Cirrhosis Data » avec uniquement les caractéristiques pertinentes. Dans notre démarche visant à améliorer encore la performance de nos modèles, nous avons exploré l'utilisation de la méta-classification. Cette approche consiste à combiner les décisions de plusieurs classifieurs de base pour obtenir une prédiction finale plus fiable. À cette fin, nous avons mis en place un méta-classifieur pour agréger les prédictions de nos modèles KNN, AD et le CB, augmentant ainsi la robustesse et l'accuracy (Exactitude en français) de nos prédictions. Pour cela avons testé plusieurs combinaisons de validations simples et croisées avec deux variantes de méta-classifieurs de niveau deux. En explorant ces techniques de sélection de fonctionna-

lités et de méta-classification, nous visons à améliorer l'accuracy et la généralisabilité de nos modèles de prédiction, ouvrant ainsi la voie à de nouvelles avancées dans la prise en charge de cette problématique de santé publique cruciale. Ceci est shématisé par la figure 3.3

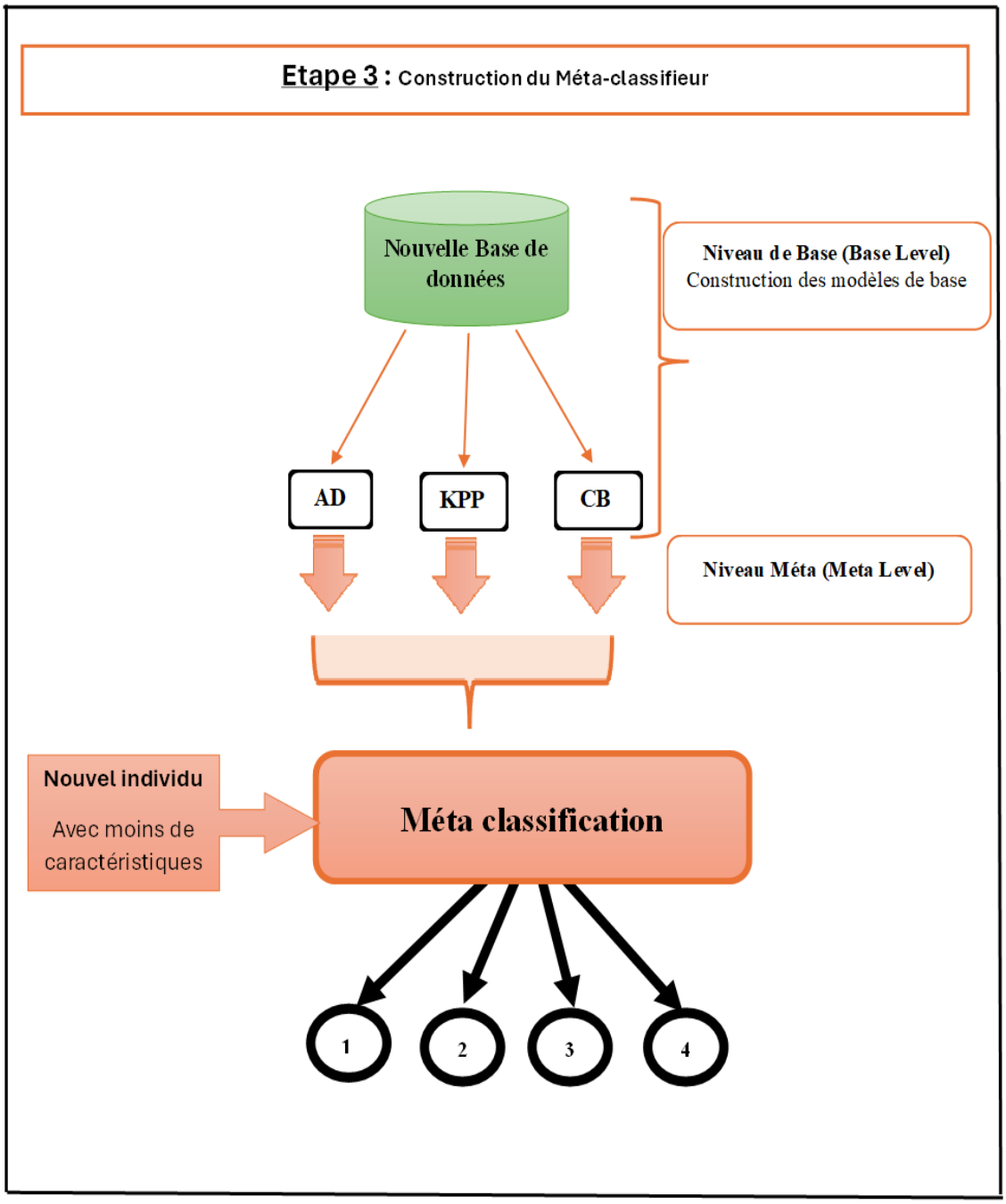


FIGURE 3.3 – Schéma de la troisième étape de construction du Méta-classifieur pour la classification.

La diversité des stades de maladies du foie (1,2 3 ou bien 4), nous a mené à la réflexion sur la possibilité de migration d'une classe à une autre (autrement dit d'un stade à un autre). Nous avons aussi proposé un algorithme de simultaion de migration entre stades de la maladie per-

mettant de calculer de nouvelles valeurs des paramètres pour pouvoir effectuer cette migration. Dans ce chapitre, nous détaillons ces processus. Nous commençons par présenter rapidement la base de données utilisée dans notre étude et nous terminons le chapitre par la présentaion de notre application. Ceci est shématisé par la figure 3.4

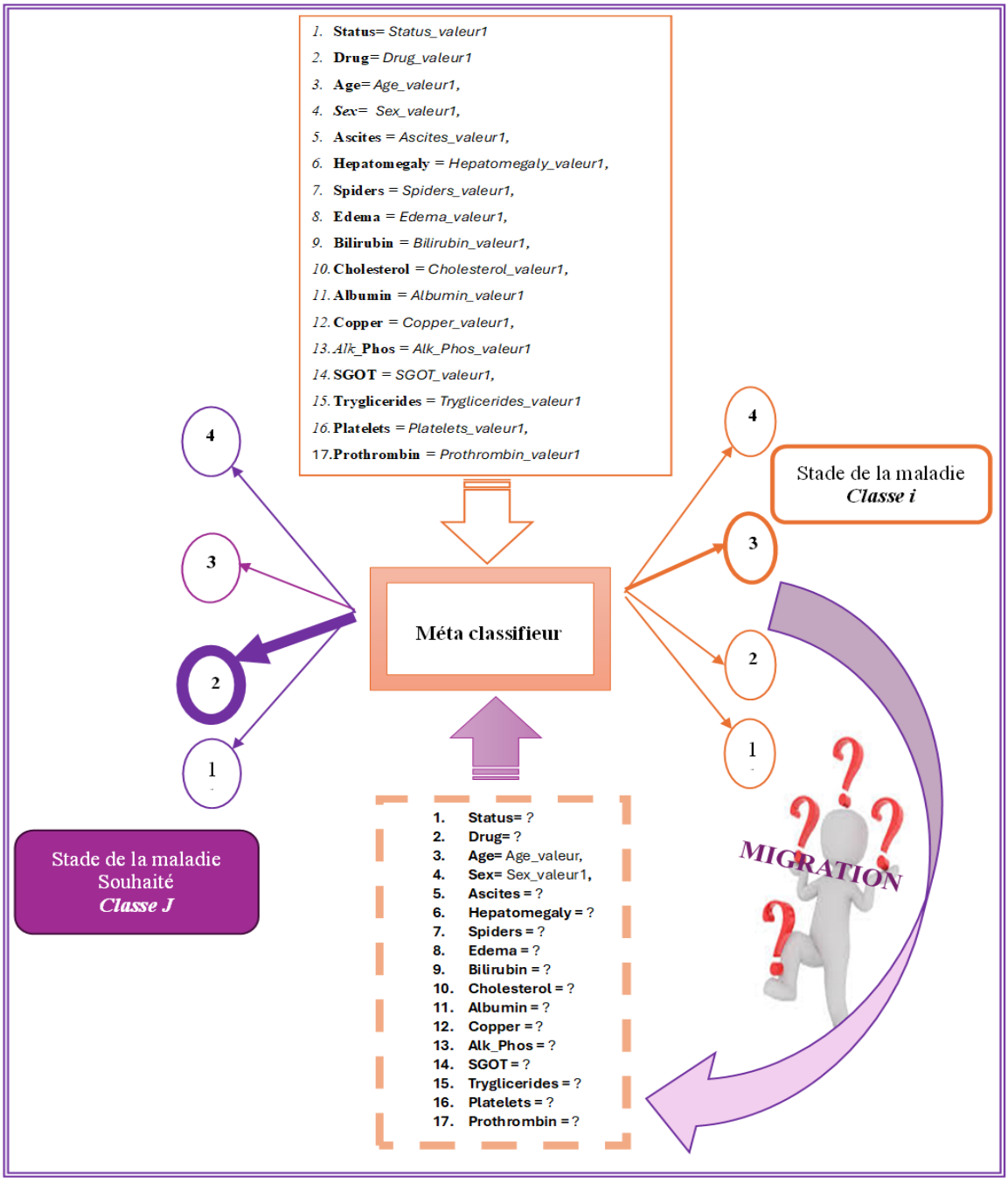
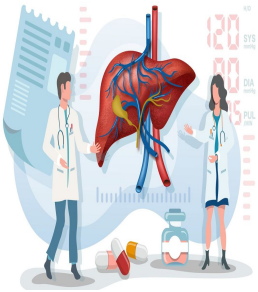


FIGURE 3.4 – Schéma de la quatrième étape de calcul des valeurs des paramètres pour la migration entre classes.

3.2 La base de données médicale de la maladie de foie



Pour évaluer notre système, nous avons utilisé la base de données “Mayo Clinic Primary Biliary Cirrhosis Data”, qui contient des informations sur la cirrhose biliaire primitive (CBP). La base inclut des données cliniques et des résultats de tests biologiques de 418 patients suivis sur dix ans. 312 patients ont participé à un essai clinique sur le médicament D-pénicillamine, tandis que 106 autres ont été suivis pour leur survie. Les données sont utilisées pour construire des modèles d’apprentissage automatique et améliorer la compréhension de la CBP.

Les attributs étudiés incluent :

- **N_Days** : Nombre de jours entre l’inscription et un événement (décès, transplantation, analyse).
- **Statut** : État du patient (censuré, transplantation, décès).
- **Drug** : Type de traitement administré (D-pénicillamine ou placebo).
- **Age** : Âge des patients
- **Sex** : Sexe des patients.
- **Ascites** : Présence ou absence d’ascite.
- **Hepatomegaly** : Présence ou absence d’hépatomégalie.
- **Spiders** : Présence ou absence de naevus araignée.
- **Edema** : Présence ou absence d’œdème.
- **Bilirubin** : Niveaux de bilirubine sanguine.
- **Cholesterol** : Niveaux de cholestérol sanguin.
- **Albumin** : Niveaux d’albumine sanguine.
- **Copper** : Taux de cuivre urinaire sur 24 heures.
- **Alk_Phos** : Niveaux de phosphatase alcaline.
- **SGOT** : Niveaux d’aspartate aminotransférase.
- **Triglycerides** : Niveaux de triglycérides sanguins.
- **Platelets** : Numération plaquettaire.
- **Prothrombin** : Temps de prothrombine.
- **Stage** : Stade histologique de la cirrhose.

Ces données permettent d'évaluer divers aspects de la CBP, y compris les symptômes, les résultats des tests et les réponses au traitement. Elles sont cruciales pour la recherche médicale et le développement de nouvelles stratégies thérapeutiques.

3.3 Techniques d'evaluation des résultats

Dans cette section, nous présenterons les résultats obtenus à l'aide de nos différentes techniques utilisées.

3.3.1 Validation croisée

Dans le domaine de l'apprentissage automatique, diverses métriques sont utilisées pour évaluer la précision prédictive d'un modèle. L'une de ces mesures est la validation croisée, qui permet d'évaluer les performances d'un modèle en simulant son utilisation dans le monde réel. Plutôt que de simplement diviser les données en un seul ensemble d'apprentissage et de test, la validation croisée consiste à diviser les données en plusieurs sous-ensembles appelés "plis" ou "folds". Le processus de validation croisée se déroule de la manière suivante : le modèle est entraîné sur une partie des données appelée ensemble d'apprentissage, puis évalué sur les données restantes, appelées ensemble de validation. Ce processus est répété plusieurs fois, chaque fois en utilisant un pli différent comme ensemble de validation, tandis que les autres plis sont utilisés comme ensemble d'apprentissage. Les performances du modèle sur chaque pli sont ensuite agrégées pour obtenir une estimation globale de sa performance. Elle permet d'évaluer la capacité du modèle à généraliser à de nouvelles données en simulant différentes situations d'apprentissage et de test. Cela permet d'obtenir une mesure plus robuste de la performance du modèle, car elle est évaluée sur plusieurs jeux de données différents. En utilisant la validation croisée, il est possible de détecter les problèmes de surapprentissage (overfitting) ou de sous-apprentissage (underfitting) du modèle. La validation croisée est une technique qui permet d'estimer la performance d'un modèle d'apprentissage automatique en utilisant plusieurs sous-ensembles de données pour l'entraînement et l'évaluation, offrant ainsi une évaluation plus fiable et plus représentative de la capacité du modèle à généraliser à de nouvelles données.

Pour évaluer les performances des trois modèles, nous avons utilisé la validation croisée. Nous avons deux annexes AnnexeA pour les expérimentaion du choix du meilleur paramètre K

pour l'algorithme KNN et AnnexeB pour les expérimentation du choix de la meilleur profondeur de l'arbre pour la classification par AD.

3.3.2 Critères et mesures d'évaluation

Pour mesurer la performance des modèles, il existe des indices ou critères qui permettent de quantifier l'écart entre les prédictions du modèle et les valeurs réelles. Ces critères servent à évaluer la précision, la sensibilité, la spécificité ou d'autres aspects de la performance prédictive du modèle. Dans cette section, nous examinerons ces différents indices et critères afin d'évaluer la performance de nos modèles d'apprentissage automatique dans la prédiction des maladies de foie.

- **Matrice de confusion :** Dans les problématiques de classification, la plupart des indices de performance sont calculés à partir d'une matrice de confusion. Cette matrice affiche le nombre de succès et d'échecs de prédiction pour chaque catégorie de la variable (attribut) à prédire. La matrice de confusion est une table qui montre chaque classe dans les données d'évaluation, ainsi que le nombre ou le pourcentage de prédictions correctes et incorrectes.

  Dans le cas d'une tâche de classification supervisée binaire, où la modalité de la variable à prédire correspond à la classe « positive » et l'autre à la classe « négative », on nomme les coefficients de la matrice de confusion de la manière suivante :

  — VN : Nombre de vrais négatifs (True Negative TN)
  — FN : Nombre de faux négatifs (False Negative FN)
  — FP : Nombre de faux positifs (False Positive FP)
  — VP : Nombre de vrais positifs (True Positive TP)

		Y prédit par le modèle	
		Y=1	Y=0
Y réel(Y')	Y'=1	Nombre de 1 prédits correctement Vrai Positifs (VP) True Positif (TP)	Nombre de 1 prédits en 0 Faux Négatif (FN) False Negatif (FN)
	Y'=0	Nombre de 0 prédits en 1 Faux Positifs (FP) False Positif (FP)	Nombre de 0 prédits correctement Vrai Négatif (VN) True Negatif (TN)

TABLE 3.1 – Matrice de Confusion

- **Accuracy (Exactitude, justesse)** : (Proportion de prédictions correctes).il s'agit d'une description d'erreurs systématiques, d'une mesure du biais statistique ; faible précision provoque une différence entre un résultat et une valeur "vraie".

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \times 100\%$$

- **Précision (Precision) :** Proportion de solutions trouvées qui sont pertinentes. A quel point les prédictions positives sont précises.

$$\text{Précision} = \frac{TP}{TP+FP} \times 100\%$$

- **Rappel (Sensitivity, Recall) :** Proportion des solutions pertinentes qui sont trouvées. Mesure la capacité du système à donner toutes les solutions pertinentes. Couverture des observations vraiment positives.

$$\text{R}appel(\text{sensitivity, recall}) = \frac{TP}{TP+FN} \times 100\%$$

- **F-mesure (F-score) :** La F-mesure correspond à un compromis de la précision et du rappel donnant la performance du modèle. Moyenne harmonique de la précision et du rappel. Mesure la capacité du modèle à donner toutes les solutions pertinentes et à refuser les autres.

$$F1 \text{ score} = \frac{2 \times (Rappel \times Prcision)}{Rappel + Prcision} \times 100\%$$

- **Matrice de confusion pour la prédiction de maladie de foie :** Notre variable de prédiction n'est pas binaire mais prend ses valeurs dans l'ensemble $\{1,2,3,4\}$.

		Classe prédite par le modèle			
		Y=1	Y=2	Y=3	Y=4
Classe réelle	Y'=1	m_{11}	m_{12}	m_{13}	m_{14}
	Y'=2	m_{21}	m_{22}	m_{23}	m_{24}
	Y'=3	m_{31}	m_{32}	m_{33}	m_{34}
	Y'=4	m_{41}	m_{42}	m_{43}	m_{44}

TABLE 3.2 – La matrice de confusion de nos modèles.

m_{ij} : représente le nombre de patients de **classe** i prédits comme **classe** j par le modèle.

m_{cc} : Le nombre de patients de la **classe** c prédits correctement par le modèle (classe i)

(représentant les vrais positifs de la **classe** c).

— **Accuracy** : Correspond à la proportion d'observations bien classées.

$$\text{Accuracy} = \frac{\sum_i m_{ii}}{\sum_{i,j} m_{ij}}$$

— **Taux d'erreur global** : Le taux d'erreur global, correspond à la proportion d'observations mal classées, qui dépend du ratio entre la trace de la matrice de confusion (c'est-à-dire la somme des coefficients diagonaux, donc le nombre de bonnes prédictions), et la somme de tous les coefficients (autrement dit le nombre total de prédictions) :

$$E = 1 - \frac{\sum_i m_{ii}}{\sum_{i,j} m_{ij}}$$

— **Précision par rapport à une classe** : La précision d'un classifieur par rapport à une certaine classe (autrement dit, par rapport à une certaine modalité de la variable à prédire), se mesure comme la proportion d'individus, parmi tous ceux pour lesquels le classifieur a prédit cette classe, qui appartiennent réellement à celle-ci.

$$Prcision_{\text{classe } c}(P_c) = \frac{m_{cc}}{\sum_i m_{ci}}$$

— **Rappel par rapport à une classe** :Le rappel d'un classifieur par rapport à une certaine classe se mesure, quant à lui, comme la proportion d'individus, parmis tous ceux qui appartiennent réellement à cette classe, pour lesquels le classifieur a prédit cette classe c.

$$Rappel_{\text{classe } c}(R_c) = \frac{m_{cc}}{\sum_j m_{cj}}$$

— **F-mesure par rapport à une classe** : On peut résumer les mesures de précision de rappel par rapport à une classe c en un seul indicateur, en calculant la moyenne harmonique :

$$F_{\text{classe } c} = \frac{P_c \times R_c}{P_c + R_c}$$

3.4 Construction de Modèles de Classification

Dans cette section, nous abordons la construction de modèles de classification à l'aide de trois techniques d'apprentissage automatique : le classifieur bayésien (CB), le plus proche voisin (KNN), et l'arbre de décision (AD). Chaque modèle est brièvement récapitulé avec une discussion sur les résultats obtenus en termes d'accuracy à l'aide de la validation croisée. Les résultats actuels montrent que ces modèles ne répondent pas pleinement aux attentes pour un classifieur efficace. Le problème se pose au niveau de l'ensemble des données utilisées pour l'apprentissage (voir section3.4.4).

3.4.1 Classifieur Bayésien (CB)

Le classifieur bayésien CB est basé sur le théorème de Bayes et fait l'hypothèse d'indépendance entre les caractéristiques. Ce modèle est simple à implémenter et efficace pour des ensembles de données avec une structure claire.

- **Classifieur Bayésien (CB) :** Accuracy moyenne de 40%

	precision	**recall**	**f1-score**	**support**
1.0	0.09	0.50	0.15	4
2.0	0.67	0.21	0.32	19
3.0	0.45	0.59	0.51	32
4.0	0.69	0.31	0.43	29
accuracy		0.40		84
weighted avg	0.57	0.40	0.42	84

TABLE 3.3 – Paramètres de précision, Rappel et Accuracy du modèle (CB)

		Classe prédite par le modèle			
		Classe 1	Classe 2	Classe 3	Classe 4
Classe réel	classe 1	2	0	2	0
	classe 2	9	4	6	0
	classe 3	8	1	19	4
	classe 4	4	1	15	9

TABLE 3.4 – Matrice de confusion du modèle (CB)

Discusion. Le modèle classifieur bayésien (CB) a montré des performances limitées, avec une accuracy globale de 40 %. Les scores de précision, de rappel et de F1 pour les classes sous-représentées (1.0 et 2.0) sont particulièrement faibles, suggérant que le modèle a des difficultés

à généraliser correctement. Cela pourrait être dû à la nature des données ou à des déséquilibres dans les classes.

3.4.2 K plus proche voisin (KNN)

Le modèle KNN classifie une observation en se basant sur les k observations les plus proches dans l'espace des caractéristiques. La classe prédite est déterminée par la majorité des k voisins les plus proches. Nous avons effectué une expérimentation pour trouver le meilleur K (voir Annexe1). Nous avons trouvé que K=5 est la meilleure valeur.

- **KNN :** Accuracy moyenne de 32%

	Précision	**Rappel**	**F1-score**	**Support**
1.0	0.09	0.25	0.13	4
2.0	0.21	0.21	0.21	19
3.0	0.34	0.38	0.36	32
4.0	0.47	0.31	0.38	29
accuracy		0.32		84
weighted avg	0.57	0.40	0.42	84

TABLE 3.5 – Paramètres de précision, rappel et accuracy du modèle (KNN)

		Classe prédite par le modèle			
		Classe 1	Classe 2	Classe 3	Classe 4
Classe réel	classe 1	1	1	2	0
	classe 2	1	5	8	5
	classe 3	0	9	16	7
	classe 4	0	5	18	6

TABLE 3.6 – Matrice de confusion de modèle (KNN)

Discusion. Le modèle du plus proche voisin KNN a une accuracy globale de 32%, inférieure à celle du classifieur bayésien (CB). Les scores de précision, de rappel et de F1 sont faibles, indiquant que ce modèle ne parvient pas à capturer efficacement les motifs dans les données.

3.4.3 Arbre de décision (AD)

L'arbre de décision segmente l'espace des caractéristiques en sous-espaces définis par des règles de décision simples. Chaque nœud de l'arbre représente une caractéristique, et chaque

branche représente une règle de décision. Nous avons effectué une expérimentation pour trouver la meilleure profondeur de l'arbre de décision (voir Annexe 2). Nous avons trouvé que la meilleure profondeur est 5. Les résultats initiaux en termes d'accuracy étaient les suivants :

- **Arbre de Décision (AD) :** Acurracy moyenne de 45%

	Précision	**Rappel**	**F1-score**	**Support**
1.0	1.00	0.25	0.40	4
2.0	0.33	0.16	0.21	19
3.0	0.43	0.62	0.51	32
4.0	0.50	0.48	0.49	29
accuracy		0.45		84
weighted avg	0.57	0.38	0.40	84

TABLE 3.7 – Paramètres de précision, rappel et accuracy du modèle (AD)

		Classe prédite par le modèle			
		Classe 1	Classe 2	Classe 3	Classe 4
Classe réel	classe 1	1	1	2	0
	classe 2	0	3	12	4
	classe 3	0	3	20	9
	classe 4	0	2	13	14

TABLE 3.8 – Matrice de confusion du modèle AD

Discusion. Le modèle arbre de décision présente les meilleures performances parmi les trois modèles testés, avec une accuracy globale de 45%. Cependant, les scores de précision et de rappel varient considérablement entre les classes, ce qui suggère des problèmes de généralisation similaires à ceux observés avec les autres modèles.

3.4.4 Problématique des données

Malgré les tentatives d'optimisation des modèles, les résultats obtenus ne sont pas satisfaisants pour un classifieur. Le problème semble résider au niveau de l'ensemble des données. Ce constat est partagé par les discussions sur le site Kaggle, où d'autres utilisateurs ont également observé des performances limitées sur ce jeu de données. Nous avons choisi de travailler avec cette base de données car c'est la seule base multiclasses disponible pour la prédiction de la cirrhose du foie que nous avons pu trouver. Les défis incluent :

- **Déséquilibre des classes :** Certaines classes sont sous-représentées, ce qui rend difficile la formation de modèles capables de bien généraliser.

- **Qualité des données :** La qualité et la diversité des features peuvent limiter la capacité des modèles à capturer les relations sous-jacentes importantes pour la prédiction.

Ces résultats montrent que l'accuracy des modèles n'était pas optimale, ce qui nous a poussés à explorer des méthodes pour améliorer les performances.

3.5 Application des techniques de Sélection d'attributs

Pour améliorer les performances de nos modèles de classification, nous avons appliqué plusieurs techniques de sélection d'attributs. La sélection d'attributs (cartéristiques) vise à identifier les caractéristiques les plus significatives et à éliminer celles qui sont redondantes ou non pertinentes, ce qui peut améliorer l'accuracy et la robustesse des modèles.

3.5.1 Techniques de Sélection d'attributs Filter

Nous avons appliqué les techniques de sélection sur l'ensemble d'apprentissage. Nous avons fait varier les paramètres de chaque technique, ce qui nous a donné une variété de combinaisons d'attributs considérés comme meilleurs pour une approche donnée. Nous avons ensuite généré une base d'apprentissage en ne gardant à chaque fois que les attributs sélectionnés. Nous avons ensuite lancé la procédure d'apprentissage pour les trois modèles et nous avons calculé l'accuracy de chaque modèle. Nous avons un tableau pour chaque méthode par modèle.

3.5.1.1 Méthode de sélection basée sur le test du chi2 (SelectPercentile)

Basée sur l'utilisation des scores Chi2 (f_classif) pour sélectionner un certain pourcentage des meilleures caractéristiques. Dans notre étude, nous avons varié le pourcentage de sélection de 5% à 100% par paliers de 5%.

Pourcentage	Atts sélectionnés	Accuracy Knn	Accuracy AD	Accuracy CB
5	4	0.34	0.43	0.09
10	4 6	0.43	0.45	0.09
15	4 5 6	0.44	0.45	0.09
20	0 4 5 6	0.42	0.48	0.25
25	0 4 5 6	0.42	0.48	0.25
30	0 4 5 6 8	0.39	0.33	0.26
35	0 4 5 6 8 11	0.30	0.37	0.26
40	0 4 5 6 8 11 16	0.27	0.40	0.28
45	0 1 4 5 6 8 11 16	0.31	0.40	0.28
50	0 1 4 5 6 8 11 16	0.31	0.40	0.28
55	0 1 4 5 6 8 10 11 16	0.28	0.51	0.27
60	0 1 4 5 6 8 10 11 15 16	0.30	0.46	0.28
65	0 1 2 4 5 6 8 10 11 15 16	0.30	0.45	0.28
70	0 1 2 4 5 6 8 9 10 11 15 16	0.28	0.45	0.34
75	0 1 2 4 5 6 8 9 10 11 15 16	0.28	0.45	0.34
80	0 1 2 4 5 6 8 9 10 11 13 15 16	0.28	0.48	0.32
85	0 1 2 4 5 6 8 9 10 11 13 14 15 16	0.36	0.48	0.38
90	0 1 2 4 5 6 8 9 10 11 12 13 14 15 16	0.32	0.46	0.37
95	0 1 2 3 4 5 6 8 9 10 11 12 13 14 15 16	0.32	0.45	0.37
100	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16	0.32	0.44	0.37

TABLE 3.9 – Calcul des accuracy des modèles Knn, AD, et CB avec les attributs proposés par SelectPercentile

Discussion des résultats de SelectPercentile

L'utilisation du SelectPercentile pour sélectionner un certain pourcentage des meilleures caractéristiques a été explorée dans notre étude. Nous avons varié le pourcentage de sélection de 5% à 100% par paliers de 5% et observé son impact sur la performance des modèles de classification KNN, AD et le CB. Dans le cas du KNN, nous avons constaté que l'ajout de caractéristiques supplémentaires au modèle n'a pas toujours amélioré sa performance. Par exemple, avec seulement 4 caractéristiques sélectionnées, nous avons obtenu une accuracy de 0.34, qui est passée à 0.45 avec l'ajout de deux autres caractéristiques. Cependant, au-delà de 10 caractéristiques, l'accuracy a diminué pour atteindre 0.27 avec 40 caractéristiques. Pour l'AD, l'ajout de caractéristiques a généralement amélioré la performance du modèle. L'accuracy est passée de 0.43 avec 4 caractéristiques à 0.51 avec 55 caractéristiques, avant de diminuer légèrement à 0.44 avec 100 caractéristiques. En ce qui concerne le CB, l'ajout de caractéristiques a eu un impact limité sur la performance. L'accuracy est restée faible, autour de 0.09 à 0.28, indépendamment du nombre de caractéristiques sélectionnées.

L'utilisation du SelectPercentile a montré des résultats variables en fonction du modèle de classification utilisé. Pour le KNN, un nombre optimal de caractéristiques semble être nécessaire pour maximiser l'accuracy, tandis que pour les AD, l'ajout de caractéristiques peut améliorer la performance jusqu'à un certain seuil avant de diminuer. Le CB, en revanche, semble ne pas bénéficier de l'ajout de caractéristiques supplémentaires.

3.5.1.2 Méthode de sélection d'attributs avec chi2 en mode k_best (GenericUnivariateSelect)

Sélection univariée basée sur les scores Chi2, avec différentes stratégies de sélection comme k_best. Nous avons testé la sélection en variant le nombre de caractéristiques de 1 à 16.

Nbr Atts	Atts sélectionnés	Accuracy KNN	Accuracy AD	Accuracy CB
1	4	0.43	0.43	0.09
2	4 6	0.34	0.45	0.09
3	4 5 6	0.45	0.45	0.09
4	0 4 5 6	0.42	0.48	0.25
5	0 4 5 6 8	0.38	0.33	0.26
6	0 4 5 6 8 11	0.30	0.37	0.26
7	0 4 5 6 8 11 16	0.27	0.42	0.28
8	0 1 4 5 6 8 11 16	0.31	0.42	0.28
9	0 1 4 5 6 8 10 11 16	0.28	0.50	0.27
10	0 1 4 5 6 8 10 11 15 16	0.30	0.46	0.28
11	0 1 2 4 5 6 8 10 11 15 16	0.30	0.45	0.28
12	0 1 2 4 5 6 8 9 10 11 15 16	0.28	0.45	0.34
13	0 1 2 4 5 6 8 9 10 11 13 15 16	0.28	0.48	0.32
14	0 1 2 4 5 6 8 9 10 11 13 14 15 16	0.36	0.48	0.38
15	0 1 2 4 5 6 8 9 10 11 12 13 14 15 16	0.32	0.45	0.37
16	0 1 2 3 4 5 6 8 9 10 11 12 13 14 15 16	0.32	0.45	0.37
17	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16	0.32	0.45	0.37

TABLE 3.10 – Calcul des accuracy des modèles Knn, AD, et CB avec GenericUnivariate

Discussion des résultats de GenericUnivariateSelect

Dans le cas du KNN, nous avons observé une tendance où l'ajout de caractéristiques supplémentaires n'a pas toujours entraîné une amélioration de la performance. Par exemple, avec seulement 4 caractéristiques sélectionnées, l'accuracy était de 0.43, et elle est passée à 0.45 avec l'ajout de deux autres caractéristiques. Cependant, au-delà de 10 caractéristiques, l'accuracy a diminué pour atteindre 0.32 avec 17 caractéristiques. Pour l'AD, l'ajout de caractéris-

tiques a généralement amélioré la performance du modèle. L'accuracy est passée de 0.43 avec 4 caractéristiques à 0.48 avec 4 caractéristiques, avant de diminuer légèrement à 0.45 avec 17 caractéristiques. En ce qui concerne le CB, l'ajout de caractéristiques a eu un impact variable sur la performance. L'accuracy est restée faible, entre 0.09 et 0.38, indépendamment du nombre de caractéristiques sélectionnées.

L'utilisation de GenericUnivariateSelect a montré des résultats variables en fonction du modèle de classification utilisé. Pour le KNN, un nombre optimal de caractéristiques semble être nécessaire pour maximiser l'accuracy, tandis que pour les AD, l'ajout de caractéristiques peut améliorer la performance jusqu'à un certain seuil avant de diminuer. Le CB, en revanche, semble avoir une performance variable qui n'est pas fortement influencée par l'ajout de caractéristiques supplémentaires.

3.5.1.3 Sélection des meilleures caractéristiques avec ANOVA(SelectKBest)

Sélection des k meilleures caractéristiques basées sur des tests univariés (ANOVA ou f_classif). Nous avons exploré différentes valeurs de k pour identifier le nombre optimal de caractéristiques à retenir.

Nbr Atts	Atts sélectionnés	Accuracy Knn	Accuracy AD	Accuracy CB
1	5	0.38	0.44	0.44
2	5 10	0.39	0.46	0.42
3	5 10 16	0.45	0.46	0.44
4	4 5 10 16	0.43	0.48	0.11
5	4 5 6 10 16	0.45	0.5	0.2
6	4 5 6 10 15 16	0.35	0.45	0.32
7	0 4 5 6 10 15 16	0.32	0.45	0.34
8	0 4 5 6 10 11 15 16	0.37	0.45	0.31
9	0 2 4 5 6 10 11 15 16	0.29	0.38	0.33
10	0 2 4 5 6 8 10 11 15 16	0.3	0.43	0.3
11	0 2 4 5 6 8 10 11 13 15 16	0.21	0.46	0.3
12	0 2 4 5 6 8 9 10 11 13 15 16	0.29	0.48	0.32
13	0 2 4 5 6 8 9 10 11 13 14 15 16	0.36	0.48	0.37
14	0 1 2 4 5 6 8 9 10 11 13 14 15 16	0.36	0.48	0.39
15	0 1 2 4 5 6 8 9 10 11 12 13 14 15 16	0.32	0.46	0.37
16	0 1 2 3 4 5 6 8 9 10 11 12 13 14 15 16	0.32	0.45	0.37
17	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16	0.32	0.45	0.37

TABLE 3.11 – Calcul des accuracy des modèles Knn, AD, et CB avec les attributs proposés par K-Best

Discussion des résultats de SelectKBest

L'utilisation de SelectKBest pour la sélection des k meilleures caractéristiques basées sur des tests univariés tels que Chi2 ou f_classif a été explorée dans notre étude. Nous avons varié le nombre de caractéristiques sélectionnées de 1 à 16 et évalué son impact sur la performance des modèles. Pour le KNN, nous avons observé que l'ajout de caractéristiques supplémentaires n'a pas toujours entraîné une amélioration de la performance. Par exemple, avec seulement 5 caractéristiques sélectionnées, l'accuracy était de 0.38, et elle est passée à 0.45 avec l'ajout de deux autres caractéristiques. Cependant, au-delà de 10 caractéristiques, l'accuracy a diminué pour atteindre 0.32 avec 17 caractéristiques. En revanche, pour l'AD, l'ajout de caractéristiques a généralement amélioré la performance du modèle. L'accuracy est passée de 0.44 avec 5 caractéristiques à 0.48 avec 4 caractéristiques, avant de diminuer légèrement à 0.45 avec 17 caractéristiques. Pour le CB, l'ajout de caractéristiques a eu un impact variable sur la performance. L'accuracy est restée modérée, entre 0.11 et 0.39, indépendamment du nombre de caractéristiques sélectionnées.

L'utilisation de SelectKBest a montré des résultats variables en fonction du modèle de classification utilisé. Pour le KNN, un nombre optimal de caractéristiques semble être nécessaire pour maximiser l'accuracy, tandis que pour les Arbres de Décision, l'ajout de caractéristiques peut améliorer la performance jusqu'à un certain seuil avant de diminuer. Le Classifieur Bayésien, en revanche, semble avoir une performance variable qui n'est pas fortement influencée par l'ajout de caractéristiques supplémentaires.

3.5.2 Techniques de Sélection d'attributs Wrapper

3.5.2.1 La Sélection Par Elimination Séquentielle (SBS)

La méthode de Sélection Par Elimination Séquentielle (SBS) a été appliquée au jeu de données pour sélectionner les attributs (caractéristiques) les plus pertinents en supprimant progressivement les attributs les moins importants. Le tableau 3.12 ci-dessous résume les résultats de l'application de la méthode SBS à notre jeu de données avec le modèle AD. Les itérations montrent la performance du modèle après la suppression séquentielle des caractéristiques.

Max Attributs	Attributs Sélectionnés	Accuracy
17	[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]	0.4524
16	[0, 1, 2, 3, 4, 5, 6, 7, 9, 10, 11, 12, 13, 14, 15, 16]	0.4405
15	[0, 1, 3, 4, 5, 6, 7, 8, 9, 10, 12, 13, 14, 15, 16]	0.4524
14	[0, 1, 2, 3, 4, 5, 6, 7, 9, 10, 11, 12, 13, 14, 15, 16]	0.4405
13	[0, 1, 3, 4, 5, 7, 9, 10, 11, 12, 13, 14, 16]	0.4762
12	[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 12, 13, 14, 15, 16]	0.4524
11	[0, 1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 12, 13, 14, 15, 16]	0.4405
10	[0, 1, 3, 4, 5, 6, 9, 11, 14, 16]	0.4405
9	[0, 1, 2, 3, 4, 5, 6, 7, 9, 10, 12, 13, 14, 16]	0.4524
8	[0, 1, 3, 4, 5, 6, 8, 10, 12, 13, 14, 15, 16]	0.4524
7	[0, 1, 3, 4, 5, 7, 8, 9, 10, 12, 13, 14, 15, 16]	0.4524
6	[0, 1, 3, 4, 5, 6, 7, 10, 12, 13, 14, 15, 16]	0.4405
5	[0, 1, 2, 4, 5, 6, 7, 9, 10, 12, 13, 14, 16]	0.4643
4	[0, 1, 3, 4, 5, 6, 8, 9, 11, 12, 13, 14, 15, 16]	0.4405
3	[0, 2, 3, 4, 5, 6, 7, 8, 10, 12, 14, 15, 16]	0.3929
2	[0, 3, 4, 5, 6, 7, 9, 10, 11, 12, 14, 16]	0.4524
1	[1, 4, 5, 7, 9, 10, 12, 13, 14, 16]	0.4643

TABLE 3.12 – Résultat de La Sélection Par Elimination Séquentielle (SBS) combinée avec AD

Discussion des résultats SBS combinée avec AD

En utilisant la méthode de Sélection Par Élimination Séquentielle (SBS), nous avons progressivement réduit le nombre d'attributs (caractéristiques) tout en évaluant l'impact sur la performance du modèle. Même en réduisant le nombre d'attributs jusqu'à un minimum de 3, le modèle maintient une certaine capacité de prédiction, bien que l'accuracy diminue significativement. Cependant, le modèle atteint sa meilleure accuracy lorsqu'il est entraîné avec un sous-ensemble de 13 attributs, avec un score de test de 0.4762. Au fur et à mesure que le nombre d'attributs diminue, des variations mineures de l'accuracy sont observées, mais dans l'ensemble, le modèle conserve une certaine stabilité dans sa performance. La sélection d'un nombre optimal d'attributs est essentielle pour équilibrer la complexité du modèle avec sa capacité prédictive. Dans ce cas, un sous-ensemble de 13 attributs semble offrir le meilleur compromis entre performance et simplicité. Ces résultats confirment l'efficacité de la méthode SBS pour simplifier les modèles tout en maintenant des performances acceptables.

Le tableau 3.13 ci-dessous résume les résultats de l'application de la méthode SBS à notre jeu de données avec le modèle KNN.

Max Attributs	Attributs Sélectionnés	Accuracy
17	[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]	0.3214
16	[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 15, 16]	0.3095
15	[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 15, 16]	0.3095
14	[0, 1, 2, 3, 4, 5, 6, 7, 9, 10, 11, 12, 15, 16]	0.3571
13	[0, 1, 2, 3, 4, 5, 6, 7, 9, 10, 11, 12, 15]	0.3452
12	[0, 1, 2, 3, 4, 5, 6, 7, 9, 11, 12, 15]	0.3452
11	[0, 1, 2, 3, 4, 5, 6, 9, 11, 12, 15]	0.3452
10	[0, 1, 2, 3, 4, 5, 9, 11, 12, 15]	0.3452
9	[0, 1, 2, 3, 4, 9, 11, 12, 15]	0.3452
8	[0, 1, 2, 3, 9, 11, 12, 15]	0.3452
7	[0, 1, 2, 9, 11, 12, 15]	0.3452
6	[0, 2, 9, 11, 12, 15]	0.3452
5	[2, 9, 11, 12, 15]	0.3452
4	[2, 9, 11, 12, 15]	0.3452
3	[2, 9, 11, 12, 15]	0.3452
2	[2, 9, 11, 12, 15]	0.3452
1	[2, 9, 11, 12, 15]	0.3452

TABLE 3.13 – Résultat de La Sélection Par Elimination Séquentielle (SBS) combinée avec KNN

Discussion des résultats SBS combinée avec KNN

En utilisant la méthode de Sélection Par Élimination Séquentielle (SBS), nous avons appliqué un modèle KNN sur notre jeu de données, et les résultats montrent que la performance initiale avec tous les attributs est de 0.3214 en termes d'accuracy. En réduisant progressivement le nombre d'attributs, nous observons que l'accuracy reste relativement stable autour de 0.3452 à partir de 14 attributs et en dessous, malgré des réductions substantielles du nombre d'attributs jusqu'à un minimum de 1 attribut. Cette stabilité de l'accuracy indique que certains attributs, notamment [2, 9, 11, 12, 15], contiennent l'essentiel de l'information nécessaire pour le modèle, permettant ainsi de maintenir la performance prédictive même avec un sous-ensemble d'attributs très réduit. La méthode SBS a donc prouvé son efficacité en identifiant les attributs les plus informatifs, simplifiant ainsi le modèle tout en conservant une performance acceptable.

Le tableau 3.14 ci-dessous résume les résultats de l'application de la méthode SBS à notre jeu de données avec le modèle CB.

Max Attributs	Attributs Sélectionnés	Accuracy
17	-	0.4524
16	[0, 1, 2, 3, 4, 5, 6, 7, 9, 10, 11, 12, 13, 14, 15, 16]	0.4405
15	[0, 1, 3, 4, 5, 6, 7, 8, 9, 10, 12, 13, 14, 15, 16]	0.4524
14	[0, 1, 2, 3, 4, 5, 6, 7, 9, 10, 11, 12, 13, 14, 15, 16]	0.4405
13	[0, 1, 3, 4, 5, 7, 9, 10, 11, 12, 13, 14, 16]	0.4762
12	[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 12, 13, 14, 15, 16]	0.4524
11	[0, 1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 12, 13, 14, 15, 16]	0.4405
10	[0, 1, 3, 4, 5, 6, 9, 11, 14, 16]	0.4405
9	[0, 1, 2, 3, 4, 5, 6, 7, 9, 10, 12, 13, 14, 16]	0.4524
8	[0, 1, 3, 4, 5, 6, 8, 10, 12, 13, 14, 15, 16]	0.4524
7	[0, 1, 3, 4, 5, 7, 8, 9, 10, 12, 13, 14, 15, 16]	0.4524
6	[0, 1, 3, 4, 5, 6, 7, 10, 12, 13, 14, 15, 16]	0.4405
5	[0, 1, 2, 4, 5, 6, 7, 9, 10, 12, 13, 14, 16]	0.4643
4	[0, 1, 3, 4, 5, 6, 8, 9, 11, 12, 13, 14, 15, 16]	0.4405
3	[0, 2, 3, 4, 5, 6, 7, 8, 10, 12, 14, 15, 16]	0.3929
2	[0, 3, 4, 5, 6, 7, 9, 10, 11, 12, 14, 16]	0.4524
1	[1, 4, 5, 7, 9, 10, 12, 13, 14, 16]	0.4643

TABLE 3.14 – Résultats de la Sélection Par Élimination Séquentielle (SBS) combinée avec CB

Discussion des résultats SBS combinée avec CB

L'analyse des résultats de la méthode de Sélection Par Élimination Séquentielle (SBS) appliquée au classifieur bayésien montre une tendance remarquable de stabilité de l'accuracy à mesure que le nombre d'attributs est réduit. Malgré des variations mineures, l'accuracy reste robuste autour de 0.44 à 0.47, même avec seulement un attribut sélectionné. Les attributs choisis par SBS, tels que [0, 1, 3, 4, 5, 6, 7, 9, 10, 12, 13, 14, 16] pour 13 attributs, réapparaissent fréquemment dans les sélections optimales, soulignant leur importance dans la classification des données. Le modèle atteint son meilleur score d'accuracy de 0.4762 avec 13 attributs, démontrant ainsi un compromis efficace entre performance et simplicité. Ces résultats valident l'efficacité de SBS pour simplifier le modèle tout en préservant sa capacité prédictive, ce qui est crucial pour optimiser les modèles d'apprentissage automatique en évitant le surapprentissage et en améliorant la généralisation.

Discussion des résultats de l'application de la technique SBS

L'application de la méthode de Sélection Par Élimination Séquentielle (SBS) sur les mêmes données, mais avec différents modèles tels que les arbres de décision, les k plus proches voisins (KNN) et le classifieur bayésien (CB), révèle une capacité robuste à optimiser la sélection des attributs tout en maintenant des performances prédictives acceptables. Pour les arbres de décision, le modèle atteint une précision maximale de 0.4762 avec 13 attributs sélectionnés, soulignant l'importance critique de trouver un équilibre entre la complexité du modèle et ses performances. De manière similaire, pour KNN et CB, la SBS démontre une stabilité remarquable de l'accuracy, maintenant des scores autour de 0.3452 pour KNN et entre 0.44 et 0.47 pour CB, même lorsque le nombre d'attributs est réduit. Ces résultats confirment que la méthode SBS est efficace pour simplifier les modèles tout en préservant leur capacité à généraliser et à éviter le surapprentissage, ce qui est essentiel en apprentissage automatique pour assurer des prédictions fiables et interprétables.

3.5.2.2 La Sélection Par Ajout Séquentiel (SFS)

La méthode (SFS) a été appliquée au jeu de données pour sélectionner les attributs en ajoutant progressivement les attributs les plus informatifs. Le tableau 3.15 ci-dessous résume les résultats de l'application de la méthode SFS à notre jeu de données avec le modèle AD.

Max Features	Attributs Sélectionnés	Accuracy du Test
1	[16]	0,44047619
2	[16, 12]	0,428571429
3	[16, 12, 4]	0,452380952
4	[16, 12, 4, 7]	0,416666667
5	[16, 12, 4, 7, 3]	0,416666667
6	[16, 12, 4, 7, 3]	0,416666667
7	[16, 12, 4, 7]	0,452380952
8	[16, 12, 4, 7]	0,416666667
9	[16, 12, 7, 4]	0,416666667
10	[16, 12, 4, 7]	0,416666667
11	[16, 12, 7, 4, 0, 6]	0,404761905
12	[16, 12, 4]	0,452380952
13	[16, 12, 7, 4]	0,416666667
14	[16, 12, 4, 7]	0,416666667
15	[16, 12, 4, 7]	0,416666667
16	[16, 12, 4, 7, 3]	0,416666667
17	[16, 12, 4, 7]	0,416666667

TABLE 3.15 – Résultat de La Sélection Par Par Ajout Séquentiel (SFS) combinée avec AD

Discussion des résultats de l'application de SFS combinée avec AD

Lorsque le paramètre "Max Features" est fixé à 1, seule la caractéristique 16 est sélectionnée, entraînant un score de 0.4671 et une accuracy de test de 0.4405. En augmentant "Max Features" à 2, la caractéristique 12 est ajoutée, mais l'accuracy diminue légèrement à 0.4286. L'inclusion de caractéristiques supplémentaires jusqu'à 4 améliore l'accuracy du test à 0.4524. Cependant, au-delà de 4 caractéristiques sélectionnées, l'accuracy du test diminue à nouveau, ce qui suggère un possible surapprentissage du modèle. En examinant ces résultats, il semble que le modèle obtient les meilleures performances lorsque 4 caractéristiques sont sélectionnées. Ajouter plus de caractéristiques n'améliore pas nécessairement l'accuracy du modèle et peut même entraîner une baisse de performance due au surapprentissage. Il est donc important de trouver un équilibre entre le nombre de caractéristiques sélectionnées et la performance du modèle pour éviter le surapprentissage.

Le tableau 3.16 ci-dessous résume les résultats de l'application de la méthode SFS à notre jeu de données avec le modèle CB.

Max Attributs	Attributs Sélectionnés	Accuracy
1	16	0.428571429
2	15,16	0.416666667
3	15,16	0.416666667
4	15,16	0.416666667
5	15,16	0.416666667
6	15,16	0.416666667
7	15,16	0.416666667
8	15,16	0.416666667
9	15,16	0.416666667
10	15,16	0.416666667
11	15,16	0.416666667
12	15,16	0.416666667
13	15,16	0.416666667
14	15,16	0.416666667
15	15,16	0.416666667
16	15,16	0.416666667
17	15,16	0.416666667

TABLE 3.16 – Résultats de l'application de la méthode SFS combinée avec CB

Discussion des résultats de l'application de SFS combinée avec CB

Les résultats de l'application de la méthode de Sélection Par Ajout Séquentiel (SFS) combinée avec le classifieur bayésien présentent une tendance surprenante, avec une sélection constante des attributs 15 et 16 et une accuracy de test identique à 0,4167 pour toutes les configurations, quelle que soit la taille du sous-ensemble d'attributs sélectionnés. Cette constance dans les résultats peut soulever des questions quant à l'efficacité de la méthode de sélection des attributs dans ce contexte particulier. Il est important de noter que cette constance peut être influencée par plusieurs facteurs, notamment le déséquilibre des classes dans la base de données multiclasse qu nous utilisons. Dans un contexte de déséquilibre, le modèle peut être biaisé vers les classes majoritaires, ce qui peut affecter la sélection des attributs. Dans ce cas, il est possible que les classes correspondant aux attributs 15 et 16 soient prédominantes dans la base de données, ce qui pourrait expliquer leur sélection constante.

Discussion des résultats de l'application de SFS

La méthode de Sélection Par Ajout Séquentiel (SFS) a été appliquée à nos données avec des résultats variés selon le classifieur utilisé. Avec le classifieur bayésien (CB), SFS a montré une sélection constante des attributs 15 et 16, atteignant une précision de test stable à 0.4167 pour toutes les configurations, indépendamment du nombre d'attributs sélectionnés. Cette constance pourrait indiquer que les attributs 15 et 16 sont prédominants dans notre ensemble de données multiclasse, ce qui peut influencer la méthode de sélection. Lorsque SFS a été combiné avec d'autres modèles comme l'arbre de décision (AD), nous avons observé une amélioration progressive de la précision du test en ajoutant des attributs informatifs. Par exemple, avec AD, l'accuracy a atteint 0.4524 lorsque quatre attributs ont été sélectionnés. Cependant, ajouter plus d'attributs n'a pas amélioré les performances et a même conduit à une diminution, suggérant un risque de surapprentissage du modèle. Ces résultats soulignent l'importance de choisir soigneusement le modèle de machine learning en fonction des caractéristiques spécifiques des données. Bien que SFS puisse simplifier et améliorer les performances prédictives avec certains modèles comme AD, il peut ne pas être aussi efficace avec d'autres comme KNN, comme indiqué par des résultats constants et non améliorés malgré la sélection d'attributs. En conclusion, l'application de SFS nécessite une évaluation approfondie de la pertinence des attributs sélectionnés par rapport au modèle de classification utilisé, afin d'optimiser à la fois la précision et la généralisation du modèle sans tomber dans le surapprentissage.

3.5.2.3 La sélection de caractéristiques avec la technique SelectFromModel

C'est une technique de sélection de fonctionnalités intégrée fournie par scikit-learn. Elle permet de sélectionner automatiquement les fonctionnalités les plus importantes à partir d'un modèle d'apprentissage automatique donné. Cette sélection se fait en fonction des poids attribués à chaque fonctionnalité par le modèle. Les fonctionnalités dont les poids dépassent un seuil spécifié sont conservées, tandis que les autres sont éliminées. Cette méthode est souvent utilisée dans le cadre de la réduction de la dimensionnalité et de la sélection de caractéristiques.

Nombre d'attributs	Arbre attributs selectionnés	Accuracy Arbres_selection
1	16	0.44
2	15 16	0.46
3	10 15 16	0.44
4	10 13 15 16	0.43
5	10 11 13 15 16	0.46
6	8 10 11 13 15 16	0.46
7	2 8 10 11 13 15 16	0.48
8	2 8 10 11 12 13 15 16	0.48
9	2 8 9 10 11 13 15 16	0.48
10	2 8 9 10 11 13 15 16	0.48
11	2 8 10 11 12 13 15 16	0.49
12	2 8 9 10 11 13 15 16	0.49
13	2 8 10 11 12 13 15 16	0.48
14	2 8 9 10 11 13 15 16	0.49
15	2 8 9 10 11 13 15 16	0.48
16	2 8 9 10 11 13 15 16	0.48
17	2 8 10 11 12 13 15 16	0.49

TABLE 3.17 – Résultat de La sélection de caractéristiques avec la technique SelectFromModel

Discussion des résultats

En observant les résultats, nous pouvons constater que la performance du modèle varie en fonction du nombre de caractéristiques sélectionnées. L'accuracy du modèle sur l'ensemble de test après la sélection des caractéristiques varie entre environ 0.44 et 0.49.

Nous avons remarqué que lorsque le nombre de caractéristiques sélectionnées augmente, l'accuracy du modèle sur l'ensemble de test a tendance à augmenter dans une certaine mesure, atteignant un maximum à un certain nombre de caractéristiques, puis diminue ou reste relativement stable. Cela peut être dû à un compromis entre la capacité du modèle à généraliser et la complexité du modèle.

3.5.3 Optimisation des Combinaisons de Caractéristiques

Après cette étape, nous avons examiné toutes les combinaisons de caractéristiques qui ont donné une bonne accuracy. Notre objectif était d'identifier les ensembles de caractéristiques les plus performants pour chaque modèle.

Modèle	Accuracy	Indices des colonnes sélectionnées
KNN	0.464285714	[5, 10, 16]
Decision Tree	0.464285714	[5, 10, 16]
Naive Bayes	0.44047619	[5, 10, 16]
KNN	0.44047619	[5, 6, 10, 16]
Decision Tree	0.488095238	[5, 6, 10, 16]
Naive Bayes	0.416666667	[5, 6, 10, 16]
KNN	0,36904762	[5, 6, 10, 16, 7, 12]
Decision Tree	0,44047619	[5, 6, 10, 16, 7, 12]
Naive Bayes	0,46428571	[5, 6, 10, 16, 7, 12]
KNN	0,42857143	[5, 6, 10, 16, 7]
Decision Tree	0,45238095	[5, 6, 10, 16, 7]
Naive Bayes	0,44047619	[5, 6, 10, 16, 7]
KNN	0,36904762	[5, 6, 10, 16, 12]
Decision Tree	0,5	[5, 6, 10, 16, 12]
Naive Bayes	0,41666667	[5, 6, 10, 16, 12]
KNN	0.44047619	[4, 5, 6]
Decision Tree	0.452380952	[4, 5, 6]
Naive Bayes	0.095238095	[4, 5, 6]
KNN	0.416666667	[0, 4, 5, 6]
Decision Tree	0.476190476	[0, 4, 5, 6]
Naive Bayes	0.25	[0, 4, 5, 6]
KNN	0.392857143	[0, 4, 5, 6, 8]
Decision Tree	0.333333333	[0, 4, 5, 6, 8]
Naive Bayes	0.261904762	[0, 4, 5, 6, 8]
KNN	0.428571429	[4, 5, 10, 16]
Decision Tree	0.476190476	[4, 5, 10, 16]
Naive Bayes	0.107142857	[4, 5, 10, 16]
KNN	0.452380952	[4, 5, 6, 10, 16]
Decision Tree	0.5	[4, 5, 6, 10, 16]
Naive Bayes	0.202380952	[4, 5, 6, 10, 16]
KNN	0.428571429	[4, 5, 6, 8, 10, 16]
Decision Tree	0.523809524	[4, 5, 6, 8, 10, 16]
Naive Bayes	0.226190476	[4, 5, 6, 8, 10, 16]
KNN	0.404761905	[0, 4, 5, 6, 8, 10, 16]
Decision Tree	0.452380952	[0, 4, 5, 6, 8, 10, 16]

Naive Bayes	0.30952381	[0, 4, 5, 6, 8, 10, 16]
KNN	0.464285714	[0, 4, 5, 6, 10, 16]
Decision Tree	0.5	[0, 4, 5, 6, 10, 16]
Naive Bayes	0.285714286	[0, 4, 5, 6, 10, 16]
KNN	0.464285714	[0, 4, 5, 6, 8, 16]
Decision Tree	0.476190476	[0, 4, 5, 6, 8, 16]
Naive Bayes	0.297619048	[0, 4, 5, 6, 8, 16]
KNN	0.452380952	[0, 4, 5, 6, 16]
Decision Tree	0.488095238	[0, 4, 5, 6, 16]
Naive Bayes	0.238095238	[0, 4, 5, 6, 16]
KNN	0.321428571	[0, 4, 5, 6, 10, 15, 16]
Decision Tree	0.452380952	[0, 4, 5, 6, 10, 15, 16]
Naive Bayes	0.345238095	[0, 4, 5, 6, 10, 15, 16]
KNN	0.285714286	[0, 1, 4, 5, 6, 8, 10, 11, 16]
Decision Tree	0.488095238	[0, 1, 4, 5, 6, 8, 10, 11, 16]
Naive Bayes	0.273809524	[0, 1, 4, 5, 6, 8, 10, 11, 16]

TABLE 3.18 – Optimisation des Combinaisons de Caractéristiques

Nous avons évalué plusieurs combinaisons de caractéristiques pour améliorer les performances des modèles de classification. Parmi celles testées, les meilleures performances ont été obtenues avec les colonnes : 5 "Ascites" , 6 "Hepatomegaly", 10 "Cholesterol" et 16 "Platelets".

3.5.4 Résultats Finaux d'Amélioration des Performances

Après application de cette combinaison optimale, nous avons reconstruit et évalué les modèles avec validation croisée, montrant une nette amélioration des performances :

3.5.4.1 Classifieur Bayésien (CB)

- **Accuracy initiale :** 40.48%
- **Accuracy améliorée :** 44%

Cette amélioration à 44% suggère que la sélection de caractéristiques a permis au CB de mieux capturer les différences entre les classes.

3.5.4.2 Le plus proche voisin (KNN)

- **Accuracy initiale :** 32.14%
- **Accuracy améliorée :** 42%

Le KNN a bénéficié d'une amélioration à 42%, réduisant le "curse of dimensionality" grâce à une sélection réduite de caractéristiques.

3.5.4.3 Arbre de décision

- **Accuracy initiale :** 45.24%
- **Accuracy améliorée :** 49%

L'arbre de décision a atteint 49%, illustrant une meilleure partition de l'espace des données avec des caractéristiques pertinentes.

Ces résultats soulignent l'importance cruciale de la sélection des caractéristiques dans la modélisation, réduisant le bruit et améliorant la généralisation des modèles.

3.6 Application de la Méta-classification

Une approche prometteuse est l'utilisation de méta-classificateurs, qui combinent les prédictions de plusieurs modèles de base pour produire une prédiction finale. Pour notre implémentation de méta-classifieurs dans le cadre de notre projet, nous avons exploré deux approches de stacking : `StackingClassifier` et `StackingCVClassifier`.

3.6.1 StackingClassifier

`StackingClassifier` est une classe de la bibliothèque scikit-learn permettant de combiner les prédictions de plusieurs modèles de base pour entraîner un méta-classifieur. L'un des avantages majeurs de `StackingClassifier` est sa simplicité d'utilisation et son intégration complète dans l'écosystème scikit-learn. Cette classe gère automatiquement la validation croisée interne, rendant le processus transparent pour l'utilisateur [6].

3.6.2 StackingCVClassifier

`StackingCVClassifier`, de la bibliothèque mlxtend, offre une gestion plus fine des folds de la validation croisée. Cela permet aux utilisateurs de mieux contrôler la façon dont les modèles de base sont évalués et combinés [32].

Nous avons utilisé ces deux techniques avec différents méta-classificateurs : `DecisionTreeClassifier` et `LogisticRegression`. Voici les résultats obtenus pour chaque combinaison :

Méta-classificateur	**Accuracy**
StackingCVClassifier + DecisionTreeClassifier	0.44
StackingCVClassifier + LogisticRegression	0.48

TABLE 3.19 – Résultats avec StackingCVClassifier

Méta-classificateur	**Accuracy**
StackingClassifier + LogisticRegression	0.48
StackingClassifier + DecisionTreeClassifier	0.5

TABLE 3.20 – Résultats avec StackingClassifier

En conclusion, l'utilisation de la technique de stacking pour combiner plusieurs modèles de classification peut améliorer les performances de prédiction par rapport à l'utilisation de modèles individuels. La combinaison avec `DecisionTreeClassifier` comme méta-classificateur a montré les meilleurs résultats avec une accuracy de 0.5. Cela suggère que la capacité de `DecisionTreeClassifier` à capturer des relations complexes entre les caractéristiques des données a contribué à améliorer la capacité de prédiction du modèle stacking. Toutefois, les performances peuvent varier en fonction des caractéristiques spécifiques de l'ensemble de données et des paramètres des modèles utilisés.

3.7 Simulation de la Migration entre classes

En matière de gestion des maladies chroniques, l'intervention précoce est cruciale. Il est essentiel d'intervenir le plus tôt possible et de manière soutenue afin de prévenir l'aggravation de la maladie et de réduire sa gravité. Une fois la maladie du foie détectée, la préoccupation principale chez les patients est de prendre le contrôle de cette maladie et le désir d'améliorer leur état de santé afin d'éviter la moindre détérioration possible. L'objectif principal demeure le ralentissement de l'évolution vers une forme invalidante de la maladie du foie.

En général, les individus se préoccupent également des changements qu'ils doivent apporter pour se maintenir dans une certaine classe de maladie ou éviter de passer à une autre classe. Nous avons donc envisagé l'idée de la migration entre les classes. Cette migration entre les classes peut être perçue comme une possibilité d'amélioration de l'état de santé d'un patient ou,

malheureusement, comme une détérioration de son état de santé avec l'apparition de complications. La maladie du foie peut être classée en cinq catégories, ce qui nous a poussés à réfléchir à la possibilité d'une transition d'une classe à une autre. Ceci se traduit par des ajustements de certaines valeurs de paramètres spécifiques. Nous avons proposé un algorithme permettant de calculer de nouvelles valeurs de paramètres pour faciliter la migration d'une classe à une autre.

3.7.1 Principe de la simulation de migration

Dans un premier temps notre système permet la prédiction du stade de sa maladie du patient donné P, en fonction de ses informations (valeurs de ces attributs). Supposons que la classe prédite est : classe_i. Ceci peut être schématisé par le schéma explicatif figure 3.5. La simulation de migration entre la classe revient à trouver de nouvelles valeurs de certains paramètres ou bien de tous les paramètres. Ces nouvelles valeurs une fois réinjectées dans le système de prédiction permettent de prédire la classe souhaitée par le patient classe_j voir figure 3.6.

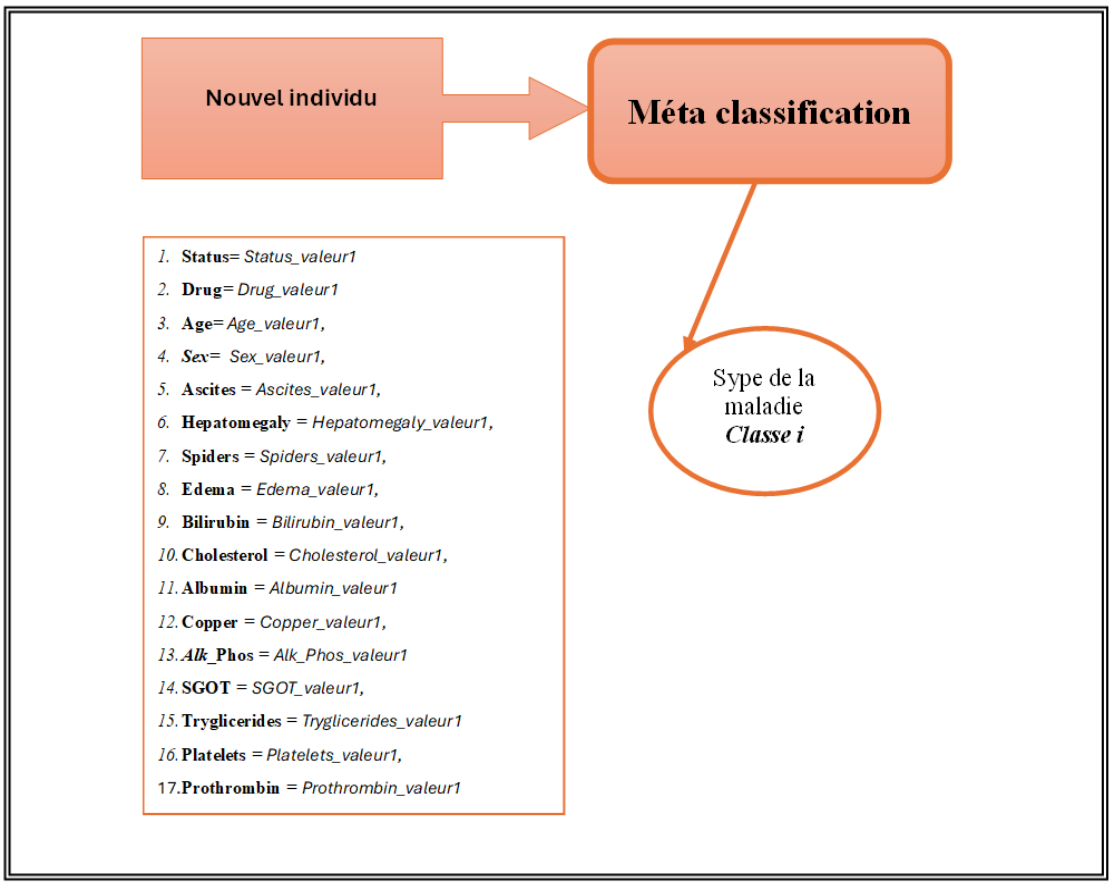


FIGURE 3.5 – Schéma explicatif de la prédiction.

P(Status=Status_valeur1, **Drug**=Drug_valeur1, **Age**=Age_valeur1, **Sex**=Sex_valeur1, **Ascites**=Ascites_valeur1, **Hepatomegaly**=Hepatomegaly_valeur1, **Spiders**=Spiders_valeur1, **Edema**=Edema_valeur1, **Bilirubin**=Bilirubin_valeur1, **Cholesterol**=Cholesterol_valeur1, **Albumin**=Albumin_valeur1, **Copper**=Copper_valeur1, **Alk_Phos**=Alk_Phos_valeur1, **SGOT**=SGOT_valeur1, **Tryglicerides**=Tryglicerides_valeur1, **Platelets**=Platelets_valeur1, **Prothrombin**=Prothrombin_valeur1)

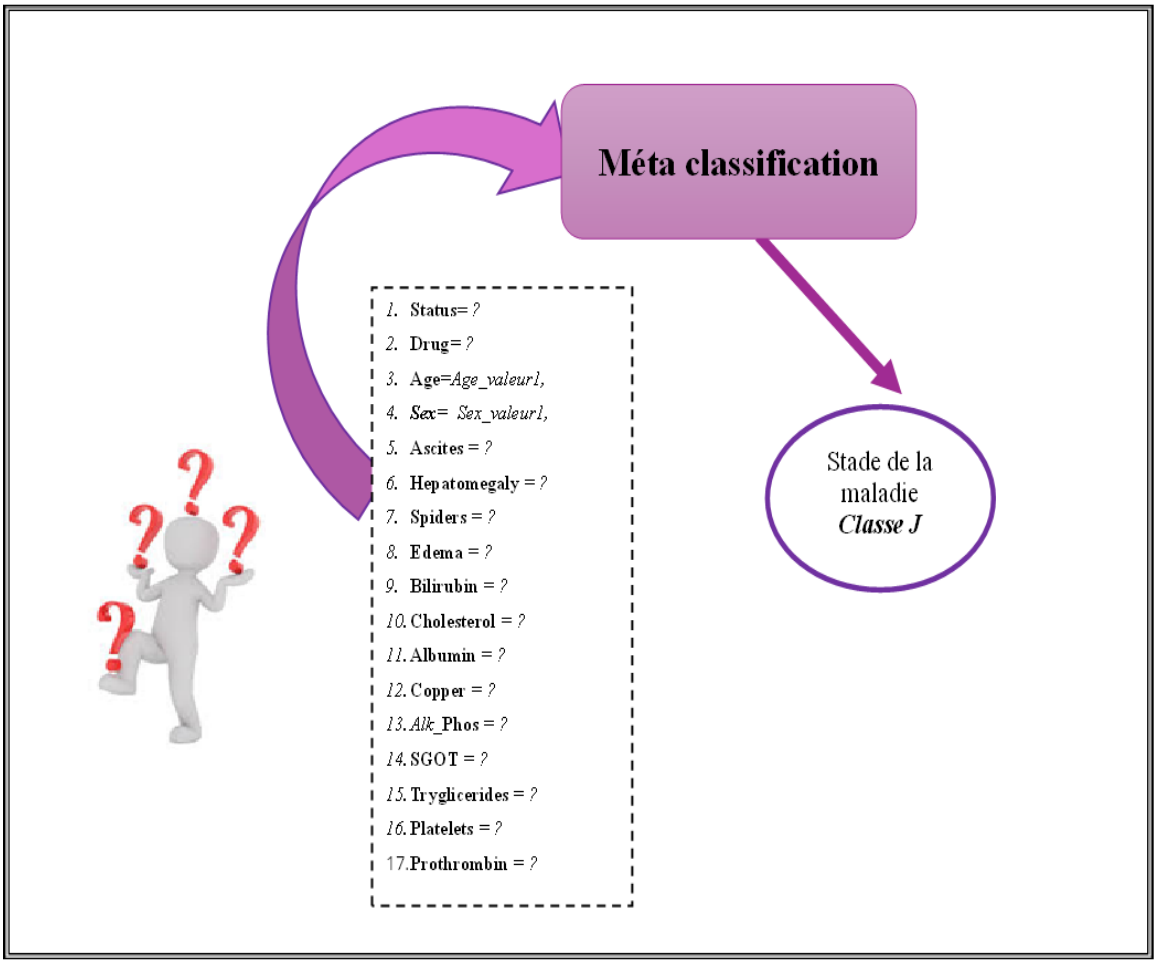


FIGURE 3.6 – Schéma explicatif de la migration entre classe.

Cet algorithme basé sur le principe du plus proche voisin qui permet de calculer de les valeurs des attributs qui permettent cette simulation de migration. Fait une recherche parmi les plus proches voisins pour trouver le voisin le plus proche qui appartient à la classe souhaitée de la migration, en prenant en compte les paramètres que le patient souhaite maintenir leurs valeurs ou pas. Les voisins sont l'ensemble d'apprentissage augmenté d'une colonne. Cette colonne représente simplement le résultat de la prédiction de cet ensemble complet, fourni par notre meilleur modèle obtenu lors de l'étape de méta-classification. Le processus consiste

à identifier, parmi les plus proches voisins, ceux qui répondent à deux types de contraintes : la classe souhaitée (contrainte_stricte) et la liste des paramètres inchangés (contraintes de valeurs). Dans ce cas, seuls les voisins ayant les mêmes valeurs que le patient pour ces paramètres spécifiques sont sélectionnés. Notre algorithme permet de déterminer les nouvelles valeurs des paramètres en prenant en considération les contraintes de classe et les valeurs des paramètres inchangés, en se basant sur le principe des plus proches voisins satisfaisant les critères imposés.

Partie 1: Méta-classification

- ***Entrée*** : Ensemble de paramètres du patient P
- ***Sortie*** :
  - La classe du patient P (***Classe-i***)
  - Tous les voisins du patient P ***ordonné*** selon leur distance par rapport au patient P (***Voisins***).

Partie 2 Méta-classification de tous les voisins

- ***Entrée*** : ***Voisins***
- ***Sortie*** : ***Voisins-prédits*** : Tous les voisins utilisés pour la prédiction du patient P ***ordonné*** en plus de la colonne ***Classe_prédite*** par le méta-classifeur.

Partie 3

- ***Entrée :***
  - La classe souhaitée (***Classe-j)***
  - ***Voisins-prédits***
  - ***Liste des paramètres inchangés L***
- ***Etape1 :***
  - *Sélection des voisins dont la classe prédite par le méta-classifieur est la classe_j*
  - ***Voisins-prédits*** = *select * from* ***Voisins-prédits*** *where classe_predite=classe_j*
- ***Etape 2:***
  - *Pour chaque paramètre* ***li*** *dans la liste* ***L*** *faire une sélection dans les* ***Voisins-prédits*** de ceux ayant la même valeur pour le paramètre ***li*** du patient ***P***
  - ***Voisins-prédits*** = *select * from* ***Voisins-prédits*** *where li=****li_val_patient_P***
- ***Sortie :***
  - ***Les voisins satisfaisant les contraintes***
  - ***Les valeurs du premier voisins satisfaisant l'enseble des contraintes***

FIGURE 3.7 – Méta-algorithme de la simulation de la migration entre classes.

3.7.2 Exemple de Migration

Soit le nouvel patient P ayant les caractéristiques présentées dans le tableau suivant, le patient est dans le stade 4 de la maladie de foie.

Status	Drug	Age	Sex	Ascites	Hepatom egaly	Spiders	Edema	Bilirubin	Choles terol	Albumin	Copper	Alk _Phos	SGOT	Trygli cerides	Platelets	Prothrombin
0	1	45	1	0	0	0	0	0,7	298	4,1	40	661	106,95	66	324	11,3

TABLE 3.21 – Les données d'un patient qui souhaite savoir les valeurs de ces paramètres pour une migration de stade de la maladie

- **Migration vers le stade de la maladie 1 :**
  - **Attribut inchangés(Staus, Age, Drug, Sex, Edema)**

    Résultat : Nous n'avons pas trouvé de résultats.

- **Migration vers le stade de la maladie 2 :**
  - **Attribut inchangés (Staus, Age, Drug, Sex, Edema)**

    Résultat : Nous avons trouvé deux combinaisons.

Status	Drug	Age	Sex	Ascites	Hepatomegaly	Spiders	Edema	Bilirubin	Cholesterol	Albumin	Copper	Alk_Phos	SGOT	Tryglicerides	Platelets	Prothrombin	Classe prédite
0	1	45	1	0	0	0	0	0.7	298	3.57	40	661	106.95	66	324	11.3	2
0	1	45	1	0	0	0	0	1	393	3.57	50	1307	74	103	295	10.5	2

  - **Attribut inchangés (Sex, Status)**

    Résultat : Nous avons trouvé 51 combinaisons.

Status	**Drug**	**Age**	**Sex**	**Ascites**	**Hepatomegaly**	**Spiders**	**Edema**	**Bilirubin**	**Cholesterol**	**Albumin**	**Copper**	**Alk_Phos**	**SGOT**	**Tryglicerides**	**Platelets**	**Prothrombin**	**Classe_Predite**
0	1	45	1	0	0	0	0	0,7	298	4,1	40	661	106,95	66	324	11,3	2
0	0	56	1	0	0	0	0	0,5	309,5	3,85	63	663	79,05	108	311	9,7	2
0	0	53	1	0	1	0	0	0,9	308	3,69	67	696	51,15	101	344	9,8	2
0	0	56	1	0	0	0	0	0,6	309,5	4,64	20	666	54,25	108	265	10,6	2
0	0	38	1	0	0	0	0	0,7	335	3,95	43	657	52	104	268	10,6	2
0	1	63	1	0	0	0	0	0,6	212	4,03	10	648	71,3	77	316	17,1	2
0	0	46	1	0	0	0	0	0,8	253	3,48	65	688	57	80	252	10	2
0	0	57	1	0	0	0	0	0,5	227	3,61	40	676	83	120	249	9,9	2
0	0	53	1	0	0	0	0	0,5	309,5	4,52	31	784	74,4	108	361	10,1	2

- **Migration vers le stade de la maladie 3 :**
  - **Attribut inchangés (Age, Sex, Status)**

    Résultat : Nous avons trouvé 2 combinations.

Status	Drug	Age	Sex	Ascites	Hepatomegaly	Spiders	Edema	Bilirubin	Cholesterol	Albumin	Copper	Alk_Phos	SGOT	Tryglicerides	Platelets	Prothrombin	Classe prédite
0	0	45	1	0	1	1	0	1.4	248	3.58	63	554	75.95	106	79	10.3	3
0	1	45	1	0	0	0	0	3.6	374	3.5	143	1428	188	44	151	10.1	3

— **Attribut inchangés (Sex, albumin)**

Résultat : Nous avons trouvé une combinaison.

Status	Drug	Age	Sex	Ascites	Hepatomegaly	Spiders	Edema	Bilirubin	Cholesterol	Albumin	Copper	Alk_Phos	SGOT	Tryglicerides	Platelets	Prothrombin	Classe prédite
-1	0	48	1	0	1	0	0	2.1	309.5	4.1	73	1259	114.7	108	200	9	3

3.7.3 Test de la simulation de la migration entre les stades de la maladie

Nous avons décidé d'appliquer notre algorithme de migration sur l'ensemble des données d'apprentissage et calculer le pourcentage de passage entre toutes les classes.

Nous avons décidé de ne maintenir que les deux paramètres **age** et **sex** :

		Migration vers le stade (classe)			
		Classe 1	Classe 2	Classe 3	Classe 4
Classe prédite	classe 1	-	89%	88%	100%
	classe 2	20%	-	88%	88%
	classe 3	19%	81%	-	91%
	classe 4	18%	68%	81%	-

TABLE 3.22 – Pourcentage des possibilités de migration (age, sex)

Nous avons décidé de ne maintenir les paramètres **age**, **sex**, **status** et **drug** :

		Migration vers la classe			
		Classe 1	Classe 2	Classe 3	Classe 4
Classe prédite	classe 1	-	75%	62%	25%
	classe 2	8%	-	45%	33%
	classe 3	7%	33%	-	44%
	classe 4	2%	19%	39%	-

TABLE 3.23 – Pourcentage des possibilités de migration (age, sex, status, drug)

Nous avons décidé de ne maintenir les paramètres **sex**, **spider** et **albumin** :

		Migration vers la classe			
		Classe 1	Classe 2	Classe 3	Classe 4
Classe prédite	classe 1	-	38%	38%	38%
	classe 2	58%	-	45%	25%
	classe 3	3%	37%	-	33%
	classe 4	4%	18%	23%	-

TABLE 3.24 – Pourcentage des possibilités de migration (albumin, sex, spider)

Il est crucial de noter que l'avis des professionnels de la santé aurait été précieux pour mieux expliquer le concept de migration entre classes dans cette étape. Certains paramètres ne peuvent pas être modifiés, mais dans notre application, nous permettons ces changements. L'application dans un domaine moins sensible, tel que le domaine commercial, permet une meilleure appréciation. La raison de choisir cette base de données est l'idée d'avoir plusieurs classes, et l'absence de bases de données avec plusieurs classes dans les références disponibles a motivé ce choix.

3.8 Outils et langage utilisés

PyCharm : Environnement de développement intégré pour Python, comprenant un débogueur graphique, la gestion des tests unitaires, l'intégration de logiciel de gestion de versions, et supportant Django.

LaTeX : Système de préparation de documents basé sur TeX, utilisé pour les documents scientifiques et techniques grâce à sa gestion des formules mathématiques et des références bibliographiques.

Python : Langage de programmation puissant et facile à apprendre, avec des structures de données de haut niveau et une programmation orientée objet. Python est idéal pour l'écriture de scripts et le développement rapide d'applications.

- **Scikit-Learn :** Cette bibliothèque Python complète offre une gamme d'algorithmes pour l'apprentissage supervisé et non supervisé, facilitant ainsi le développement de modèles prédictifs dans divers domaines.
- **Matplotlib et Seaborn :** Nous avons utilisé Matplotlib et Seaborn pour créer des graphiques et des visualisations de nos données, ce qui nous a permis d'explorer et de comprendre les tendances et les modèles présents dans les données.
- **NumPy et Pandas :** Extensions de Python permettant la manipulation de matrices multidimensionnelles et des fonctions mathématiques associées. Pandas, quant à lui, est une bibliothèque Python pour la manipulation et l'analyse des données, offrant des structures de données et des opérations sur les tableaux numériques, sous licence libre.
- **Openpyxl :** Nous avons utilisé Openpyxl pour manipuler des fichiers Excel, notamment pour enregistrer les résultats de nos expériences et les visualisations générées à partir des données.
- **Scikit-Learn et MLxtend :** Nous avons utilisé Scikit-Learn pour accéder à une variété d'algorithmes d'apprentissage automatique, y compris les arbres de décision, les voisins les plus proches et le classificateur naïf de Bayes. MLxtend a été utilisé pour implémen-

ter des techniques d'apprentissage ensembliste, telles que le stacking, qui combine plusieurs modèles de base pour améliorer les performances de prédiction.

- **Joblib :** Nous avons utilisé Joblib pour sauvegarder et charger nos modèles d'apprentissage automatique, ce qui nous a permis de les réutiliser ultérieurement sans avoir à les recalculer.
- **Autres fonctionnalités de Scikit-Learn :** Nous avons utilisé les fonctionnalités de Scikit-Learn telles que le calcul des métriques de classification (précision, rappel, F1-score), la création de courbes ROC et de courbes de précision-rappel, ainsi que la normalisation des données avec MinMaxScaler.

3.9 Présentation de l'application

La Figure 3.8 présente le menu principal de l'application.

La Figure 3.8 présente le menu principal de l'application. Il est composé de quatre sous-menus :

- "Nouveau modèle" : Donne à l'utilisateur la possibilité de prédire le stade de sa maladie du foie et la simulation de la migration entre les classes après l'étape de remplissage des paramètres en entrée en utilisant les meilleurs caractéristiques.
- "Sélection d'attributs" : Affiche les résultats de la sélection.
- "Ancien Modèle" même travail que le nouveau modèle mais avec l'ensemble complet des caractéristiques.
- "À Propos" : Offre des détails sur l'application elle-même.

Le sous-menu Prédiction de la maladie avec l'un des trois modèles avec les meilleurs caractéristiques : le plus proche voisin (KNN), l'Arbre de Décision (AD) ou le classifieur bayésien (CB) (voir Figure 3.9). Le formulaire donne la main au patient pour remplir ses informations et de prédire à quelle classe est-il associé (Stade de sa maladie 1.2.3.4). Selon les 3 méthodes le résultat s'affiche sur l'écran.

FIGURE 3.8 – Menu Principal.

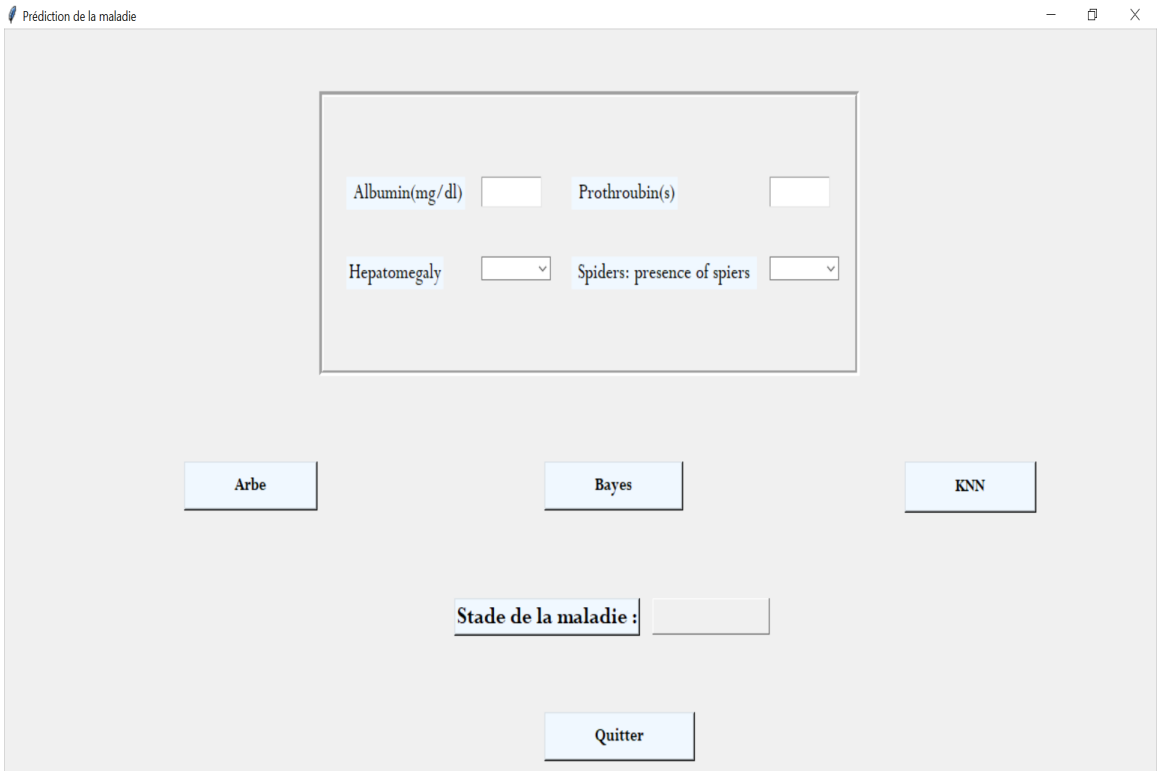
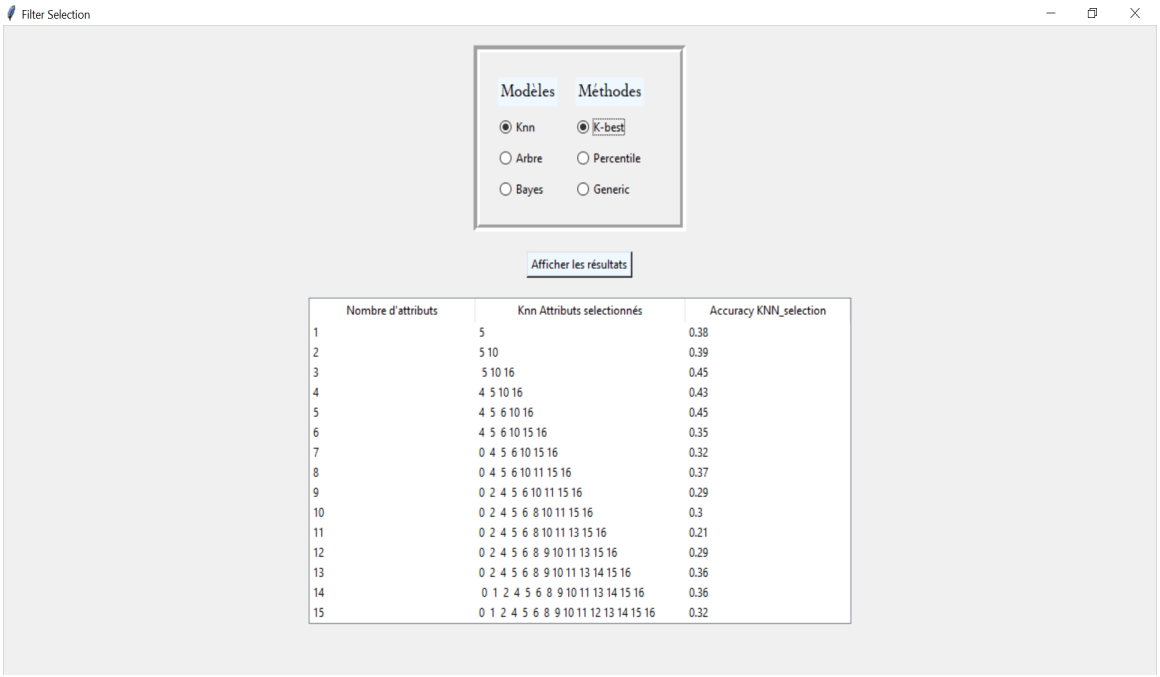


FIGURE 3.9 – Formulaire de prédiction du stade de la maladie de foie.

La figure 3.10 présente le sous menu "Selection d'attributs", il contient l'ensemble des résulats des tests de sélection d'attributs pour l'ensembles des téchniques que nous avons testé. Nous avons choisi le résultat de l'application de la méthode k-best avec le modèle KNN.



Nombre d'attributs	Knn Attributs selectionnés	Accuracy KNN_selection
1	5	0.38
2	5 10	0.39
3	5 10 16	0.45
4	4 5 10 16	0.43
5	4 5 6 10 16	0.45
6	4 5 6 10 15 16	0.35
7	0 4 5 6 10 15 16	0.32
8	0 4 5 6 10 11 15 16	0.37
9	0 2 4 5 6 10 11 15 16	0.29
10	0 2 4 5 6 8 10 11 15 16	0.3
11	0 2 4 5 6 8 10 11 13 15 16	0.21
12	0 2 4 5 6 8 9 10 11 13 15 16	0.29
13	0 2 4 5 6 8 9 10 11 13 14 15 16	0.36
14	0 1 2 4 5 6 8 9 10 11 13 14 15 16	0.36
15	0 1 2 4 5 6 8 9 10 11 12 13 14 15 16	0.32

FIGURE 3.10 – Résultat de la sélection d'attributs.

Le formulaire suivant concerne la simulation de la migration entre les différents stades de la maladie. Tout d'abord, le patient remplit ses informations. Ensuite, une prédiction du stade de sa maladie est effectuée à l'aide du méta-classifieur. Il doit ensuite choisir le stade qu'il souhaite simuler, et une opération de recherche est effectuée pour lui proposer les valeurs des attributs les plus proches de ses caractéristiques, lui permettant ainsi de passer au stade souhaité.

3.10 Conclusion

Dans notre étude, nous avons exploré une approche complète pour la prédiction des maladies du foie en combinant plusieurs techniques d'apprentissage automatique. Notre démarche a débuté par la sélection minutieuse des caractéristiques pertinentes à l'aide de méthodes de filtrage, suivie par une optimisation des combinaisons de paramètres pour chaque modèle. Par la suite, nous avons intégré une composante cruciale à notre approche : la méta-classification.

La méta-classification consiste à combiner les prédictions des différents modèles de prédiction pour obtenir une prédiction finale plus fiable et robuste. En utilisant une méthode de stacking, les prédictions des modèles individuels ont été agrégées pour renforcer l'accuracy globale des prédictions. Cette étape a été essentielle pour améliorer la performance prédictive de nos modèles.

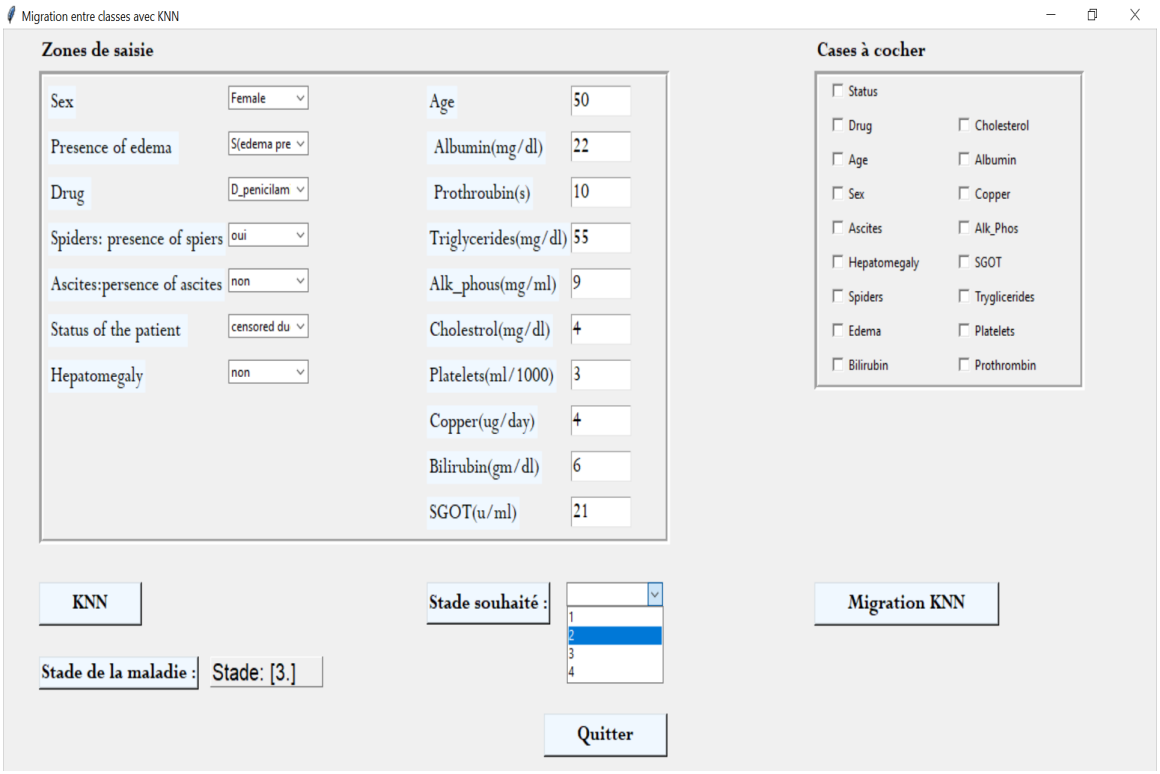


FIGURE 3.11 – Résultat de la migration.

Par la suite, nous avons exploré l'idée novatrice de migration entre les stades de la maladie, impliquant le calcul des valeurs des paramètres pour faciliter cette transition. Cette approche intégrée nous a permis de développer des modèles de prédiction des maladies du foie précis et fiables. Ces modèles offrent des perspectives prometteuses pour améliorer la prise en charge clinique des patients et contribuer à la lutte contre les maladies du foie à l'échelle mondiale. Cette approche intégrée illustre l'importance de la combinaison de différentes techniques d'apprentissage automatique pour obtenir des résultats optimaux dans des domaines médicaux complexes.

Conclusion Générale et perspectives

Conclusion

Les maladies chroniques, telles que les maladies du foie, le diabète et le cancer, représentent un défi majeur pour les systèmes de santé à travers le monde. Leur impact considérable sur la qualité de vie des individus et sur les ressources médicales nécessite une approche proactive et novatrice pour leur prévention et leur gestion. La complexité des interactions entre les différents facteurs de risque rend souvent difficile la prédiction manuelle de ces maladies, soulignant ainsi le besoin de solutions innovantes. Dans ce contexte, l'apprentissage automatique émerge comme une solution prometteuse. En exploitant l'intelligence artificielle et l'analyse avancée des données, l'apprentissage automatique offre la possibilité de modéliser et de prédire le développement de ces maladies chroniques. En analysant de vastes ensembles de données médicales, ces techniques permettent d'identifier des schémas subtils et des corrélations cachées, ouvrant ainsi la voie à une détection précoce et à une intervention ciblée. Par ailleurs, les outils d'aide à la décision basés sur l'apprentissage automatique offrent une nouvelle perspective sur la manière dont les professionnels de la santé peuvent aborder ces maladies. En intégrant ces technologies dans les pratiques cliniques, les médecins sont mieux équipés pour prendre des décisions éclairées et personnalisées, tout en optimisant l'allocation des ressources et en améliorant les résultats pour les patients. En conclusion, ce projet de modélisation de la maladie du foie a été une exploration approfondie des différentes facettes de la prédiction de ces affections chroniques. À travers une série d'étapes méthodiques, nous avons abordé les défis complexes liés à la prédiction, à la sélection des caractéristiques, à l'optimisation des modèles de prédiction combinée à l'utilisation de la méta-classification, à la simulation de la migration entre les stades de la maladie, ainsi qu'à la création d'une application fonctionnelle pour tous ces aspects. Les résultats obtenus et l'application développée fournissent une base solide pour

améliorer la compréhension et les pratiques de gestion de ces maladies graves.

- **Prédiction et intervention** L'analyse des données a confirmé l'importance cruciale de la détection précoce des maladies du foie pour une intervention efficace. En utilisant des techniques avancées de modélisation, nous avons appliqué et adapté des modèles prédictifs existants, leur permettant d'identifier avec précision les patients présentant un risque de progression de la maladie.
- **Sélection des caractéristiques** La sélection des caractéristiques a joué un rôle essentiel dans l'amélioration des performances des modèles de prédiction. En identifiant les variables les plus pertinentes, nous avons pu réduire la dimensionnalité des données tout en maintenant ou en améliorant la précision des prédictions.
- **Optimisation des modèles** Nous avons utilisé diverses techniques d'optimisation pour affiner nos modèles de prédiction et améliorer leur performance. Cela comprenait l'ajustement des hyperparamètres, la validation croisée et l'utilisation d'ensembles de méthodes pour améliorer la robustesse et la généralisation de nos modèles.
- **Méta-classification** La méta-classification, qui consiste à combiner les prédictions de différents modèles de prédiction pour obtenir une prédiction finale plus fiable et robuste, a été une composante cruciale de notre approche. En utilisant une méthode de stacking, les prédictions des modèles individuels ont été agrégées pour renforcer la précision globale des prédictions.
- **Simulation de la migration entre les stades** Nous avons développé une méthodologie pour simuler la migration entre les stades de la maladie, permettant ainsi d'explorer les transitions possibles et les ajustements nécessaires pour maintenir ou modifier le stade de la maladie. Cette approche a ouvert la voie à une meilleure compréhension des dynamiques de progression de la maladie.
- **Développement d'une application fonctionnelle** En plus des modèles de prédiction, nous avons également conçu et développé une application fonctionnelle pour tous les aspects du processus de gestion des maladies du foie. Cette application offre aux praticiens de la santé et aux patients un outil pratique pour suivre et gérer la progression de la maladie.

Perspectives futures

- **Exploration de nouvelles techniques d'apprentissage automatique :** Pour continuer à améliorer les performances des modèles de prédiction, il serait intéressant d'explorer de nouvelles techniques d'apprentissage automatique telles que l'apprentissage en ligne, l'apprentissage par transfert, les réseaux de neurones profonds, ou encore l'utilisation de méthodes d'ensemble. Ces approches pourraient permettre de développer des modèles plus robustes et plus précis pour la prédiction et la classification.
- **Collaboration avec des experts médicaux :** Une collaboration étroite avec des experts médicaux pourrait enrichir la recherche en fournissant des informations cliniques précieuses et en validant les modèles développés dans des contextes réels. Cela garantirait que les outils et les techniques développés sont adaptés aux besoins des praticiens de la santé et des patients.
- **Déploiement de l'application :** En développant une application basée sur nos modèles de prédiction, nous pourrions avoir un impact direct sur la prise en charge des patients atteints de maladies du foie. Cette application pourrait être utilisée par les médecins pour évaluer le risque de progression de la maladie et recommander des interventions personnalisées aux patients.
- **Élargissement vers d'autres domaines d'application :** Les techniques d'apprentissage automatique développées dans le cadre de ce projet ne se limitent pas à la médecine. Elles pourraient être étendues à d'autres domaines tels que l'économie, l'environnement, le marketing, la logistique, la sécurité et la défense, l'éducation, ainsi que l'industrie. Cette expansion vers d'autres domaines offrirait de nouvelles opportunités pour résoudre des problèmes complexes et contribuer à l'innovation dans ces secteurs.

En conclusion, Ce projet a été une exploration enrichissante du domaine de l'apprentissage automatique appliqué aux maladies chroniques. Nous avons eu l'opportunité d'explorer plusieurs techniques de modélisation et de prédiction, et l'idée de simuler la migration entre les stades de la maladie s'est révélée innovante et intéressante. Les perspectives futures ouvrent de nombreuses possibilités pour continuer à améliorer la gestion et la prévention des maladies du foie.

Références

[1] J. Han, M. Kamber, and J. Pei, *Data Mining : Concepts and Techniques*. Elsevier, 2011.

[2] A. Djeffal, "Cours fouille de données avancée," *Faculté des Sciences Exactes et des Sciences de la Nature et de la Vie : Université Mohamed Khider-Biskra*, pp. 6–8, 2014.

[3] L. Batache and S. Drici, "Etude et recherche bibliographique sur les méthodes de classification," Master's thesis, Université Mouloud Mammeri, 2019.

[4] S. Boutouhami, "Cours fouille et extraction de données," *Département d'Informatique, Université de BBA*, 2024.

[5] U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, "Advances in knowledge discovery and data mining." American Association for Artificial Intelligence, 1996.

[6] Scikit-learn, "Stackingclassifier," accessed : 2024-05-31. [Online]. Available : https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.StackingClassifier.html

[7] F. Chamroukhi, "Classification supervisée : Les k-plus proches voisins," Master's thesis, Université du Sud Toulon–Var, 2013.

[8] M. Taffar, "Initialisation à l'apprentissage automatique," 2014, support de cours pour étudiants en Master en Intelligence Artificielle, Université de Jijel.

[9] W. J. Frawley, G. Piatetsky-Shapiro, and C. J. Matheus, "Knowledge discovery in databases : An overview," *AI Magazine*, vol. 13, no. 3, pp. 57–57, 1992.

[10] R. Yade, "Classification bayésienne, apprentissage et réseaux de neurones : application en science des données," Ph.D. dissertation, Université du Québec à Trois-Rivières, 2022.

[11] J. Vaidya, M. Kantarcioglu, and C. Clifton, "Privacy-preserving naive bayes classification," *The VLDB Journal*, vol. 17, no. 4, pp. 879–898, 2008.

[12] L. Rokach, "Ensemble-based classifiers," *Artificial Intelligence Review*, vol. 33, no. 1-2, pp. 1–39, 2010.

[13] H. Liu and L. Yu, "Toward integrating feature selection algorithms for classification and clustering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 4, pp. 491–502, 2005.

[14] C. Meynet, "Sélection de variables pour la classification non supervisée en grande dimension," Ph.D. dissertation, Paris 11, 2012.

[15] J. Vaidya, M. Kantarcioglu, and C. Clifton, "Privacy-preserving naive bayes classification," *The VLDB Journal*, vol. 17, no. 4, pp. 879–898, 2008.

[16] J. H. Gennari, P. Langley, and D. Fisher, "Models of incremental concept formation," *Artificial Intelligence*, vol. 40, no. 1-3, pp. 11–61, 1989.

[17] R. Kohavi and G. H. John, "Wrappers for feature subset selection," *Artificial Intelligence*, vol. 97, no. 1-2, pp. 273–324, 1997.

[18] A. Alaoui, "Application des techniques des métaheuristiques pour l'optimisation de la tâche de la classification de la fouille de données," Master's thesis, USTO, 2012.

[19] D. Koller and M. Sahami, "Toward optimal feature selection," in *ICML*, vol. 96, no. 28, 1996, p. 292.

[20] H. Chouaib, "Sélection de caractéristiques : méthodes et applications," *Paris Descartes University*, 2011.

[21] K. Kira and L. A. Rendell, "The feature selection problem : Traditional methods and a new algorithm," in *Proceedings of the tenth national conference on Artificial Intelligence*, 1992, pp. 129–134.

[22] B. Chetiou and N. Halle, "Nouvelles variantes des méta-heuristiques compactes," Master's thesis, Abdelhafid Boussouf University Centre-Mila, 2019.

[23] M. Karima, “Détection d’anomalies sur les données biologiques par svm,” Master’s thesis, Université Mouloud Mammeri de Tizi-Ouzou, 2012.

[24] S. Benhammada, “Etude comparative de méthodes de sélection de caractéristiques en apprentissage automatique. proposition d’une variante,” Ph.D. dissertation, Université Mentouri de Constantine Faculté des sciences de l’ingénieur, 2007.

[25] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, “Scikit-learn : Machine learning in python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[26] Scikit-learn, “Feature selection using selectfrommodel.” [Online]. Available : https://scikit-learn.org/stable/modules/feature_selection.html

[27] ——, “Feature selection using selectkbest.” [Online]. Available : https://scikit-learn.org/stable/modules/feature_selection.html

[28] I. Kononenko, “Estimating attributes : Analysis and extensions of relief,” in *European Conference on Machine Learning*. Springer, 1994, pp. 171–182.

[29] L. Yu and H. Liu, “Feature selection for high-dimensional data : A fast correlation-based filter solution,” in *Proceedings of the Twentieth International Conference on Machine Learning (ICML-03)*, 2003.

[30] X.-S. Yang, “Firefly algorithms for multimodal optimization,” in *International Symposium on Stochastic Algorithms*. Springer, 2009, pp. 169–178.

[31] J. Kennedy and R. C. Eberhart, “Particle swarm optimization,” in *Proceedings of ICNN’95 - International Conference on Neural Networks*, vol. 4. IEEE, 1995, pp. 1942–1948.

[32] S. Raschka, *StackingCVClassifier*, accessed : 2024-05-31. [Online]. Available : http://rasbt.github.io/mlxtend/user_guide/classifier/StackingCVClassifier/

Annexe A

La recherche du meilleur K pour le modèle KNN

k = 1

For Fold 1 the accuracy is **29.761904761904763**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	1.00	1.00	1.00	17
2.0	1.00	1.00	1.00	73
3.0	1.00	1.00	1.00	129
4.0	1.00	1.00	1.00	115
accuracy			1.00	334
macro avg	1.00	1.00	1.00	334
weighted avg	1.00	1.00	1.00	334

Paramètres de precision rappel et accuracy du model KNN pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.16	0.21	0.18	19
3.0	0.35	0.43	0.35	32
4.0	0.43	0.34	0.38	29
accuracy			0.30	84
macro avg	0.24	0.22	0.23	84
weighted avg	0.32	0.30	0.31	84

For Fold 2 the accuracy is **34.523809523809526**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	1.00	1.00	1.00	17
2.0	1.00	1.00	1.00	73
3.0	1.00	1.00	1.00	129
4.0	1.00	1.00	1.00	115
accuracy			1.00	334
macro avg	1.00	1.00	1.00	334
weighted avg	1.00	1.00	1.00	334

Paramètres de precision rappel et accuracy du model KNN pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.27	0.21	0.24	19
3.0	0.37	0.34	0.35	32
4.0	0.45	0.48	0.47	29
accuracy			0.35	84
macro avg	0.27	0.26	0.26	84
weighted avg	0.36	0.35	0.35	84

For Fold 3 the accuracy is **39.285714285714285**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	1.00	1.00	1.00	16
2.0	1.00	1.00	1.00	74
3.0	1.00	1.00	1.00	129
4.0	1.00	1.00	1.00	115
accuracy			1.00	334
macro avg	1.00	1.00	1.00	334
weighted avg	1.00	1.00	1.00	334

Paramètres de precision rappel et accuracy du model KNN pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	5
2.0	0.31	0.44	0.36	18

3.0	0.39	0.44	0.41	32
4.0	0.65	0.38	0.48	29
accuracy			0.39	84
macro avg	0.34	0.32	0.31	84
weighted avg	0.44	0.39	0.40	84

For Fold 4 the accuracy is **37.34939759036144**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	1.00	1.00	1.00	17
2.0	1.00	1.00	1.00	74
3.0	1.00	1.00	1.00	129
4.0	1.00	1.00	1.00	115
accuracy			1.00	335
macro avg	1.00	1.00	1.00	335
weighted avg	1.00	1.00	1.00	335

Paramètres de precision rappel et accuracy du model KNN pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.20	0.25	0.22	4
2.0	0.33	0.39	0.36	18
3.0	0.45	0.41	0.43	32
4.0	0.36	0.34	0.35	29
accuracy			0.37	83
macro avg	0.33	0.35	0.34	83
weighted avg	0.38	0.37	0.38	83

For Fold 5 the accuracy is **31.32530120481928**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	1.00	1.00	1.00	17
2.0	1.00	1.00	1.00	74
3.0	1.00	1.00	1.00	128
4.0	1.00	1.00	1.00	116
accuracy			1.00	335
macro avg	1.00	1.00	1.00	335

weighted avg	1.00	1.00	1.00	335

Paramètres de precision rappel et accuracy du model KNN pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.14	0.17	0.15	18
3.0	0.48	0.39	0.43	33
4.0	0.37	0.36	0.36	28

accuracy			0.31	83
macro avg	0.25	0.23	0.24	83
weighted avg	0.35	0.31	0.33	83

k = 2

For Fold 1 the accuracy is **25.0**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.41	1.00	0.59	17
2.0	0.51	0.86	0.64	73
3.0	0.69	0.65	0.67	129
4.0	1.00	0.43	0.60	115

accuracy			0.64	334
macro avg	0.66	0.74	0.62	334
weighted avg	0.75	0.64	0.64	334

Paramètres de precision rappel et accuracy du model KNN pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.11	0.25	0.15	4
2.0	0.18	0.32	0.23	19
3.0	0.30	0.28	0.29	32
4.0	0.45	0.17	0.25	29

accuracy			0.25	84
macro avg	0.26	0.25	0.23	84
weighted avg	0.32	0.25	0.26	84

For Fold 2 the accuracy is **33.33333333333333**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	recall	f1-score	support
1.0	0.47	1.00	0.64	17
2.0	0.49	0.90	0.64	73
3.0	0.67	0.64	0.66	129
4.0	1.00	0.36	0.53	115
accuracy			0.62	334
macro avg	0.66	0.73	0.62	334
weighted avg	0.74	0.62	0.61	334

Paramètres de précision, rappel et accuracy du modèle KNN pour test

	precision	recall	f1-score	support
1.0	0.00	0.00	0.00	4
2.0	0.25	0.32	0.28	19
3.0	0.42	0.50	0.46	32
4.0	0.55	0.21	0.30	29
accuracy			0.33	84
macro avg	0.30	0.26	0.26	84
weighted avg	0.41	0.33	0.34	84

For Fold 3 the accuracy is **32.142857142857146**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	recall	f1-score	support
1.0	0.44	1.00	0.62	16
2.0	0.50	0.88	0.64	74
3.0	0.71	0.69	0.70	129
4.0	1.00	0.38	0.55	115
accuracy			0.64	334
macro avg	0.67	0.74	0.63	334
weighted avg	0.75	0.64	0.63	334

Paramètres de précision, rappel et accuracy du modèle KNN pour test

	precision	recall	f1-score	support
1.0	0.00	0.00	0.00	5
2.0	0.24	0.50	0.33	18
3.0	0.38	0.34	0.36	32

4.0	0.78	0.24	0.37	29
accuracy		0.32	84	
macro avg	0.35	0.27	0.26	84
weighted avg	0.47	0.32	0.33	84

For Fold 4 the accuracy is **34.93975903614458**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.44	1.00	0.61	17
2.0	0.52	0.86	0.65	74
3.0	0.67	0.67	0.67	129
4.0	1.00	0.37	0.54	115
accuracy		0.63	335	
macro avg	0.65	0.73	0.62	335
weighted avg	0.74	0.63	0.62	335

Paramètres de précision, rappel et accuracy du modèle KNN pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.12	0.25	0.17	4
2.0	0.32	0.56	0.41	18
3.0	0.38	0.41	0.39	32
4.0	0.50	0.17	0.26	29
accuracy		0.35	83	
macro avg	0.33	0.35	0.31	83
weighted avg	0.40	0.35	0.34	83

For Fold 5 the accuracy is **26.506024096385545**

Paramètres de précision, rappel et accuracy du modèle KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.44	1.00	0.61	17
2.0	0.53	0.93	0.67	74
3.0	0.66	0.60	0.63	128
4.0	1.00	0.41	0.59	116

accuracy			0.63	335
macro avg	0.66	0.74	0.62	335
weighted avg	0.74	0.63	0.62	335

Paramètres de précision, rappel et accuracy du modèle KNN pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.15	0.28	0.20	18
3.0	0.46	0.39	0.43	33
4.0	0.40	0.14	0.21	28

accuracy			0.27	83
macro avg	0.25	0.20	0.21	83
weighted avg	0.35	0.27	0.28	83

k = 3

For Fold 1 the accuracy is **28.57142857142857**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.29	0.71	0.41	17
2.0	0.52	0.64	0.57	73
3.0	0.71	0.61	0.66	129
4.0	0.82	0.63	0.72	115

accuracy			0.63	334
macro avg	0.58	0.65	0.59	334
weighted avg	0.68	0.63	0.65	334

Paramètres de précision, rappel et accuracy du modèle KNN pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.12	0.16	0.14	19
3.0	0.41	0.34	0.37	32
4.0	0.43	0.34	0.38	29

accuracy			0.29	84
macro avg	0.24	0.21	0.22	84
weighted avg	0.33	0.29	0.31	84

For Fold 2 the accuracy is **30.952380952380953**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.35	0.71	0.47	17
2.0	0.43	0.56	0.49	73
3.0	0.70	0.63	0.66	129
4.0	0.78	0.60	0.68	115
accuracy			0.61	334
macro avg	0.56	0.62	0.57	334
weighted avg	0.65	0.61	0.62	334

Paramètres de précision, rappel et accuracy du modèle KNN pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.08	0.25	0.12	4
2.0	0.23	0.26	0.24	19
3.0	0.41	0.34	0.37	32
4.0	0.41	0.31	0.35	29
accuracy			0.31	84
macro avg	0.28	0.29	0.27	84
weighted avg	0.35	0.31	0.32	84

For Fold 3 the accuracy is **28.57142857142857**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.35	0.88	0.50	16
2.0	0.50	0.62	0.55	74
3.0	0.73	0.63	0.68	129
4.0	0.79	0.63	0.70	115
accuracy			0.64	334
macro avg	0.59	0.69	0.61	334
weighted avg	0.68	0.64	0.65	334

Paramètres de précision, rappel et accuracy du modèle KNN pour test

	precision	recall	f1-score	support
1.0	0.00	0.00	0.00	5
2.0	0.17	0.28	0.21	18
3.0	0.33	0.31	0.32	32
4.0	0.56	0.31	0.40	29
accuracy			0.29	84
macro avg	0.27	0.23	0.23	84
weighted avg	0.36	0.29	0.31	84

For Fold 4 the accuracy is **28.915662650602407**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	recall	f1-score	support
1.0	0.31	0.65	0.42	17
2.0	0.50	0.65	0.56	74
3.0	0.71	0.62	0.66	129
4.0	0.79	0.63	0.70	115
accuracy			0.63	335
macro avg	0.58	0.64	0.59	335
weighted avg	0.67	0.63	0.64	335

Paramètres de précision, rappel et accuracy du modèle KNN pour test

	precision	recall	f1-score	support
1.0	0.00	0.00	0.00	4
2.0	0.24	0.33	0.28	18
3.0	0.33	0.31	0.32	32
4.0	0.38	0.28	0.32	29
accuracy			0.29	83
macro avg	0.24	0.23	0.23	83
weighted avg	0.31	0.29	0.30	83

For Fold 5 the accuracy is **21.686746987951807**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	recall	f1-score	support
1.0	0.40	0.82	0.54	17

2.0	0.49	0.64	0.55	74
3.0	0.73	0.66	0.69	128
4.0	0.82	0.61	0.70	116
accuracy			0.65	335
macro avg	0.61	0.68	0.62	335
weighted avg	0.69	0.65	0.66	335

Paramètres de précision, rappel et accuracy du modèle KNN pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.04	0.06	0.04	18
3.0	0.36	0.24	0.29	33
4.0	0.35	0.32	0.33	28
accuracy			0.22	83
macro avg	0.19	0.15	0.17	83
weighted avg	0.27	0.22	0.24	83

k = 4

For Fold 1 the accuracy is **33.33333333333333**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.21	0.29	0.24	17
2.0	0.49	0.51	0.50	73
3.0	0.60	0.65	0.62	129
4.0	0.69	0.56	0.62	115
accuracy			0.57	334
macro avg	0.49	0.50	0.49	334
weighted avg	0.58	0.57	0.57	334

Paramètres de precision rappel et accuracy du model KNN pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.22	0.21	0.22	19
3.0	0.38	0.44	0.41	32

4.0	0.40	0.34	0.37	29
accuracy			0.33	84
macro avg	0.25	0.25	0.25	84
weighted avg	0.33	0.33	0.33	84

For Fold 2 the accuracy is **30.952380952380953**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.33	0.35	0.34	17
2.0	0.51	0.52	0.51	73
3.0	0.61	0.73	0.66	129
4.0	0.76	0.57	0.65	115
accuracy			0.61	334
macro avg	0.55	0.54	0.54	334
weighted avg	0.62	0.61	0.61	334

Paramètres de précision, rappel et accuracy du modèle KNN pour le test :

	precision	**recall**	**f1-score**	**support**
1.0	0.09	0.25	0.13	4
2.0	0.21	0.21	0.21	19
3.0	0.34	0.38	0.36	32
4.0	0.47	0.31	0.38	29
accuracy			0.31	84
macro avg	0.28	0.29	0.27	84
weighted avg	0.35	0.31	0.32	84

For Fold 3 the accuracy is **38.095238095238095**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.42	0.50	0.46	16
2.0	0.55	0.55	0.55	74
3.0	0.59	0.70	0.64	129
4.0	0.65	0.50	0.56	115
accuracy			0.59	334

macro avg	0.55	0.56	0.55	334
weighted avg	0.59	0.59	0.58	334

Paramètres de précision, rappel et accuracy du modèle KNN pour le test :

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	5
2.0	0.26	0.33	0.29	18
3.0	0.44	0.53	0.48	32
4.0	0.53	0.31	0.39	29

accuracy			0.38	84
macro avg	0.31	0.29	0.29	84
weighted avg	0.40	0.38	0.38	84

For Fold 4 the accuracy is **36.144578313253014**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.24	0.41	0.30	17
2.0	0.51	0.47	0.49	74
3.0	0.63	0.67	0.65	129
4.0	0.67	0.57	0.62	115

accuracy			0.58	335
macro avg	0.51	0.53	0.52	335
weighted avg	0.60	0.58	0.59	335

Paramètres de précision, rappel et accuracy du modèle KNN pour le test :

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.43	0.33	0.38	18
3.0	0.36	0.47	0.41	32
4.0	0.36	0.31	0.33	29

accuracy			0.36	83
macro avg	0.29	0.28	0.28	83
weighted avg	0.36	0.36	0.35	83

For Fold 5 the accuracy is **34.93975903614458**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.24	0.29	0.26	17
2.0	0.56	0.59	0.58	74
3.0	0.58	0.66	0.62	128
4.0	0.73	0.55	0.63	116
accuracy			0.59	335
macro avg	0.53	0.53	0.52	335
weighted avg	0.61	0.59	0.59	335

Paramètres de précision, rappel et accuracy du modèle KNN pour le test :

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.09	0.06	0.07	18
3.0	0.42	0.52	0.47	33
4.0	0.46	0.39	0.42	28
accuracy			0.35	83
macro avg	0.24	0.24	0.24	83
weighted avg	0.34	0.35	0.34	83

k = 5

For Fold 1 the accuracy is **36.904761904761905**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.18	0.12	0.14	17
2.0	0.44	0.53	0.48	73
3.0	0.58	0.64	0.61	129
4.0	0.69	0.54	0.60	115
accuracy			0.56	334
macro avg	0.47	0.46	0.46	334
weighted avg	0.56	0.56	0.56	334

Paramètres de précision, rappel et accuracy du modèle KNN pour le test :

	precision	recall	f1-score	support
1.0	0.00	0.00	0.00	4
2.0	0.26	0.26	0.26	19
3.0	0.42	0.50	0.46	32
4.0	0.45	0.34	0.39	29
accuracy			0.37	84
macro avg	0.28	0.28	0.28	84
weighted avg	0.38	0.37	0.37	84

For Fold 2 the accuracy is **34.523809523809526**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	recall	f1-score	support
1.0	0.25	0.18	0.21	17
2.0	0.47	0.52	0.50	73
3.0	0.53	0.66	0.59	129
4.0	0.67	0.47	0.55	115
accuracy			0.54	334
macro avg	0.48	0.46	0.46	334
weighted avg	0.55	0.54	0.54	334

Paramètres de precision rappel et accuracy du modèle KNN pour le test :

	precision	recall	f1-score	support
1.0	0.00	0.00	0.00	4
2.0	0.28	0.37	0.32	19
3.0	0.37	0.44	0.40	32
4.0	0.47	0.28	0.35	29
accuracy			0.35	84
macro avg	0.28	0.27	0.27	84
weighted avg	0.37	0.35	0.34	84

For Fold 3 the accuracy is **33.33333333333333**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	recall	f1-score	support

1.0	0.36	0.25	0.30	16
2.0	0.53	0.59	0.56	74
3.0	0.58	0.69	0.63	129
4.0	0.63	0.47	0.54	115
accuracy			0.57	334
macro avg	0.52	0.50	0.51	334
weighted avg	0.57	0.57	0.57	334

Paramètres de precision rappel et accuracy du modèle KNN pour le test :

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	5
2.0	0.17	0.28	0.21	18
3.0	0.41	0.41	0.41	32
4.0	0.56	0.34	0.43	29
accuracy			0.33	84
macro avg	0.28	0.26	0.26	84
weighted avg	0.38	0.33	0.35	84

For Fold 4 the accuracy is **27.710843373493976**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.29	0.24	0.26	17
2.0	0.50	0.55	0.53	74
3.0	0.56	0.66	0.60	129
4.0	0.67	0.50	0.58	115
accuracy			0.56	335
macro avg	0.50	0.49	0.49	335
weighted avg	0.57	0.56	0.56	335

Paramètres de precision rappel et accuracy du model KNN pour le test :

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.20	0.28	0.23	18

3.0	0.26	0.25	0.25	32
4.0	0.43	0.34	0.38	29
accuracy			0.28	83
macro avg	0.22	0.22	0.22	83
weighted avg	0.29	0.28	0.28	83

For Fold 5 the accuracy is **26.506024096385545**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.32	0.35	0.33	17
2.0	0.48	0.49	0.48	74
3.0	0.56	0.70	0.62	128
4.0	0.76	0.54	0.63	116
accuracy			0.58	335
macro avg	0.53	0.52	0.52	335
weighted avg	0.60	0.58	0.58	335

Paramètres de precision rappel et accuracy du modèle KNN pour le test :

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.08	0.06	0.07	18
3.0	0.34	0.36	0.35	13
4.0	0.31	0.32	0.32	28
accuracy			0.27	83
macro avg	0.18	0.19	0.18	83
weighted avg	0.26	0.27	0.26	83

k = 6

For Fold 1 the accuracy is **39.285714285714285**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.15	0.12	0.13	17
2.0	0.46	0.51	0.48	73

3.0	0.56	0.62	0.59	129
4.0	0.66	0.56	0.60	115
accuracy			0.55	334
macro avg	0.46	0.45	0.45	334
weighted avg	0.55	0.55	0.55	334

Paramètres de précision rappel et accuracy du modèle KNN pour le test :

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.23	0.26	0.2	19
3.0	0.46	0.50	0.48	32
4.0	0.46	0.41	0.44	29
accuracy			0.39	84
macro avg	0.29	0.29	0.29	84
weighted avg	0.38	0.39	0.39	84

For Fold 2 the accuracy is **38.095238095238095**

Paramètres de précision rappel et accuracy du modèle KNN pour l'apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.36	0.29	0.32	17
2.0	0.46	0.48	0.47	73
3.0	0.55	0.65	0.59	129
4.0	0.62	0.49	0.55	115
accuracy			0.54	334
macro avg	0.50	0.48	0.48	334
weighted avg	0.54	0.54	0.54	334

Paramètres de précision rappel et accuracy du modèle KNN pour le test :

	precision	**recall**	**f1-score**	**support**
1.0	0.14	0.25	0.18	4
2.0	0.22	0.21	0.22	19
3.0	0.44	0.50	0.47	32
4.0	0.48	0.38	0.42	29

accuracy			0.38	84
macro avg	0.32	0.33	0.32	84
weighted avg	0.39	0.38	0.38	84

For Fold 3 the accuracy is **38.095238095238095**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.19	0.19	0.19	16
2.0	0.51	0.51	0.51	74
3.0	0.60	0.71	0.65	129
4.0	0.61	0.49	0.54	115

accuracy			0.56	334
macro avg	0.48	0.47	0.47	334
weighted avg	0.56	0.56	0.56	334

Paramètres de precision rappel et accuracy du modèle KNN pour le test :

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	5
2.0	0.26	0.39	0.31	18
3.0	0.44	0.50	0.47	32
4.0	0.53	0.31	0.39	29

accuracy			0.38	84
macro avg	0.31	0.30	0.29	84
weighted avg	0.41	0.38	0.38	84

For Fold 4 the accuracy is **30.120481927710845**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.29	0.24	0.26	17
2.0	0.43	0.53	0.47	74
3.0	0.56	0.62	0.59	129
4.0	0.62	0.48	0.54	115

accuracy			0.53	335
macro avg	0.48	0.47	0.47	335
weighted avg	0.54	0.53	0.53	335

Paramètres de precision rappel et accuracy du modèle KNN pour le test :

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.14	0.11	0.12	18
3.0	0.32	0.41	0.36	32
4.0	0.42	0.34	0.38	29

accuracy			0.30	83
macro avg	0.22	0.22	0.21	83
weighted avg	0.30	0.30	0.30	83

For Fold 5 the accuracy is **31.32530120481928**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.19	0.18	0.18	17
2.0	0.44	0.53	0.48	74
3.0	0.59	0.64	0.62	128
4.0	0.74	0.59	0.66	116

accuracy			0.58	335
macro avg	0.49	0.48	0.49	335
weighted avg	0.59	0.58	0.58	335

Paramètres de precision rappel et accuracy du modèle KNN pour le test :

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.08	0.06	0.06	18
3.0	0.36	0.39	0.38	33
4.0	0.43	0.43	0.43	28

accuracy			0.31	83
macro avg	0.22	0.22	0.22	83

weighted avg	0.30	0.31	0.31	83

k = 7

For Fold 1 the accuracy is **41.66666666666667**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.36	0.24	0.29	17
2.0	0.52	0.48	0.50	73
3.0	0.57	0.67	0.62	128
4.0	0.66	0.59	0.62	115
accuracy			0.58	334
macro avg	0.53	0.50	0.51	334
weighted avg	0.58	0.58	0.58	334

Paramètres de precision rappel et accuracy du modèle KNN pour le test :

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.28	0.26	0.27	19
3.0	0.47	0.53	0.50	33
4.0	0.46	0.45	0.46	29
accuracy			0.42	84
macro avg	0.30	0.31	0.31	84
weighted avg	0.40	0.42	0.41	84

For Fold 2 the accuracy is **36.904761904761905**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.22	0.12	0.15	17
2.0	0.43	0.34	0.38	73
3.0	0.53	0.70	0.60	128
4.0	0.63	0.54	0.58	115
accuracy			0.54	334
macro avg	0.45	0.42	0.43	334

weighted avg	0.53	0.54	0.52	334

Paramètres de precision rappel et accuracy du modèle KNN pour test :

	precision	**recall**	**f1-score**	**support**
1.0	0.12	0.25	0.17	4
2.0	0.27	0.16	0.20	19
3.0	0.43	0.56	0.49	32
4.0	0.39	0.31	0.35	29
accuracy			0.37	84
macro avg	0.30	0.32	0.30	84
weighted avg	0.37	0.37	0.36	84

For Fold 3 the accuracy is **42.857142857142854**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.21	0.25	0.23	16
2.0	0.54	0.42	0.47	74
3.0	0.56	0.70	0.62	128
4.0	0.60	0.50	0.55	115
accuracy			0.55	334
macro avg	0.48	0.47	0.47	334
weighted avg	0.55	0.55	0.54	334

Paramètres de precision rappel et accuracy du modèle KNN pour test :

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	5
2.0	0.27	0.33	0.30	18
3.0	0.45	0.59	0.51	32
4.0	0.61	0.38	0.47	29
accuracy			0.43	84
macro avg	0.33	0.33	0.32	84
weighted avg	0.44	0.43	0.42	84

For Fold 4 the accuracy is **42.168674698795186**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	recall	f1-score	support
1.0	0.22	0.24	0.23	17
2.0	0.47	0.38	0.42	74
3.0	0.57	0.68	0.62	128
4.0	0.62	0.56	0.59	115
accuracy			0.55	335
macro avg	0.47	0.46	0.46	335
weighted avg	0.55	0.55	0.54	335

Paramètres de precision rappel et accuracy du modèle KNN pour test :

	precision	recall	f1-score	support
1.0	0.00	0.00	0.00	4
2.0	0.45	0.28	0.34	18
3.0	0.43	0.50	0.46	32
4.0	0.44	0.48	0.46	29
accuracy			0.42	83
macro avg	0.33	0.32	0.32	83
weighted avg	0.42	0.42	0.41	83

For Fold 5 the accuracy is **31.32530120481928**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	recall	f1-score	support
1.0	0.27	0.24	0.25	17
2.0	0.45	0.45	0.45	74
3.0	0.58	0.69	0.63	128
4.0	0.72	0.59	0.65	116
accuracy			0.58	335
macro avg	0.51	0.49	0.50	335
weighted avg	0.58	0.58	0.58	335

Paramètres de precision rappel et accuracy du modèle KNN pour test :

	precision	recall	f1-score	support
1.0	0.00	0.00	0.00	4
2.0	0.15	0.11	0.13	18
3.0	0.36	0.39	0.38	33
4.0	0.39	0.39	0.39	28
accuracy			0.31	83
macro avg	0.23	0.22	0.22	83
weighted avg	0.31	0.31	0.31	83

k = 8

For Fold 1 the accuracy is **40.476190476190474**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	recall	f1-score	support
1.0	0.67	0.12	0.20	17
2.0	0.50	0.45	0.47	73
3.0	0.55	0.67	0.60	129
4.0	0.62	0.59	0.61	115
accuracy			0.57	334
macro avg	0.59	0.46	0.47	334
weighted avg	0.57	0.57	0.56	334

Paramètres de précision, rappel et accuracy du modèle KNN pour test :

	precision	recall	f1-score	support
1.0	0.00	0.00	0.00	4
2.0	0.24	0.21	0.22	19
3.0	0.47	0.47	0.47	32
4.0	0.44	0.52	0.48	29
accuracy			0.40	84
macro avg	0.29	0.30	0.29	84
weighted avg	0.38	0.40	0.39	84

For Fold 2 the accuracy is **38.095238095238095**

Paramètres de précision, rappel et accuracy du modèle KNN pour apprentissage

	precision	recall	f1-score	support

1.0	0.33	0.18	0.23	17
2.0	0.51	0.40	0.45	73
3.0	0.54	0.70	0.61	129
4.0	0.62	0.55	0.58	115
accuracy			0.55	334
macro avg	0.50	0.45	0.47	334
weighted avg	0.55	0.55	0.54	334

Paramètres de précision, rappel et accuracy du modèle KNN pour test :

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.31	0.21	0.25	19
3.0	0.40	0.59	0.48	32
4.0	0.50	0.31	0.38	29
accuracy			0.38	84
macro avg	0.30	0.28	0.28	84
weighted avg	0.40	0.38	0.37	84

For Fold 3 the accuracy is **40.476190476190474**

Paramètres de précision, rappel et accuracy du modèle KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.23	0.19	0.21	16
2.0	0.45	0.35	0.39	74
3.0	0.55	0.76	0.64	129
4.0	0.67	0.50	0.57	115
accuracy			0.55	334
macro avg	0.48	0.45	0.45	334
weighted avg	0.55	0.55	0.54	334

Paramètres de précision, rappel et accuracy du modèle KNN pour test :

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	5
2.0	0.24	0.33	0.28	18
3.0	0.45	0.62	0.53	32
4.0	0.57	0.28	0.37	29

accuracy			0.40	84
macro avg	0.32	0.31	0.29	84
weighted avg	0.42	0.40	0.39	84

For Fold 4 the accuracy is **43.373493975903614**

Paramètres de précision, rappel et accuracy du modèle KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.29	0.12	0.17	17
2.0	0.48	0.39	0.43	74
3.0	0.54	0.71	0.62	129
4.0	0.60	0.51	0.55	115

accuracy			0.54	335
macro avg	0.48	0.43	0.44	335
weighted avg	0.54	0.54	0.53	335

Paramètres de précision, rappel et accuracy du modèle KNN pour test :

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.20	0.11	0.14	18
3.0	0.44	0.59	0.51	32
4.0	0.52	0.52	0.52	29

accuracy			0.43	83
macro avg	0.29	0.31	0.29	83
weighted avg	0.39	0.43	0.41	83

For Fold 5 the accuracy is **31.32530120481928**

Paramètres de précision, rappel et accuracy du modèle KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.22	0.12	0.15	17
2.0	0.45	0.39	0.42	74
3.0	0.56	0.70	0.62	128
4.0	0.66	0.58	0.62	116

accuracy			0.56	335
macro avg	0.47	0.45	0.45	335
weighted avg	0.55	0.56	0.55	335

Paramètres de précision, rappel et accuracy du modèle KNN pour test :

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.14	0.11	0.12	18
3.0	0.35	0.42	0.38	33
4.0	0.38	0.36	0.37	28
accuracy			0.31	83
macro avg	0.22	0.22	0.22	83
weighted avg	0.30	0.31	0.30	83

k = 9

For Fold 1 the accuracy is **44.047619047619044**

Paramètres de precision rappel et accuracy du model KNN pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.33	0.06	0.10	17
2.0	0.45	0.45	0.45	73
3.0	0.53	0.64	0.58	129
4.0	0.58	0.51	0.55	115
accuracy			0.53	334
macro avg	0.47	0.42	0.42	334
weighted avg	0.52	0.53	0.52	334

Paramètres de précision, rappel et accuracy du modèle KNN pour test :

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.31	0.26	0.29	19
3.0	0.51	0.59	0.55	32
4.0	0.46	0.45	0.46	29
accuracy			0.44	84

macro avg	0.32	0.33	0.32	84
weighted avg	0.43	0.44	0.43	84

For Fold 2 the accuracy is **38.095238095238095**

Paramètres de précision, rappel et accuracy du modèle KNN pour apprentissage :

	precision	**recall**	**f1-score**	**support**
1.0	0.38	0.18	0.24	17
2.0	0.47	0.37	0.41	73
3.0	0.56	0.71	0.63	129
4.0	0.62	0.56	0.59	115
accuracy			0.56	334
macro avg	0.50	0.45	0.47	334
weighted avg	0.55	0.56	0.55	334

Paramètres de précision, rappel et accuracy du modèle KNN pour test :

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.31	0.26	0.29	19
3.0	0.41	0.59	0.49	32
4.0	0.44	0.28	0.34	29
accuracy			0.44	84
macro avg	0.29	0.28	0.28	84
weighted avg	0.38	0.38	0.37	84

For Fold 3 the accuracy is **38.095238095238095**

Paramètres de précision, rappel et accuracy du modèle KNN pour apprentissage :

	precision	**recall**	**f1-score**	**support**
1.0	0.22	0.12	0.16	16
2.0	0.44	0.35	0.39	74
3.0	0.55	0.71	0.62	129
4.0	0.61	0.52	0.56	115
accuracy			0.54	334
macro avg	0.46	0.43	0.43	334

weighted avg	0.53	0.54	0.53	334

Paramètres de précision, rappel et accuracy du modèle KNN pour test :

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	5
2.0	0.21	0.22	0.22	18
3.0	0.43	0.59	0.50	32
4.0	0.45	0.31	0.37	29
accuracy			0.38	84
macro avg	0.27	0.28	0.27	84
weighted avg	0.36	0.38	0.36	84

For Fold 4 the accuracy is **43.373493975903614**

Paramètres de précision, rappel et accuracy du modèle KNN pour apprentissage :

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	17
2.0	0.51	0.47	0.49	74
3.0	0.57	0.70	0.63	129
4.0	0.62	0.58	0.60	115
accuracy			0.57	335
macro avg	0.42	0.44	0.43	335
weighted avg	0.54	0.57	0.56	335

Paramètres de précision, rappel et accuracy du modèle KNN pour test :

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.30	0.17	0.21	18
3.0	0.42	0.53	0.47	32
4.0	0.48	0.55	0.52	29
accuracy			0.43	83
macro avg	0.30	0.31	0.30	83
weighted avg	0.40	0.43	0.41	83

For Fold 5 the accuracy is **36.144578313253014**

Paramètres de précision, rappel et accuracy du modèle KNN pour apprentissage :

	precision	**recall**	**f1-score**	**support**
1.0	0.38	0.18	0.24	17
2.0	0.49	0.38	0.43	74
3.0	0.54	0.72	0.62	128
4.0	0.66	0.58	0.62	116
accuracy			0.57	335
macro avg	0.52	0.46	0.48	335
weighted avg	0.57	0.57	0.56	335

Paramètres de précision, rappel et accuracy du modèle KNN pour test :

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.18	0.17	0.17	18
3.0	0.42	0.48	0.45	33
4.0	0.41	0.39	0.40	28
accuracy			0.36	83
macro avg	0.25	0.26	0.26	83
weighted avg	0.34	0.36	0.35	83

Annexe B

La recherche de la meilleur profendeur du modèle AD

profondeur = 2

For Fold 1 the accuracy is **46.42857142857143**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.00	1.00	1.00	17
2.0	0.83	0.07	0.13	73
3.0	0.47	0.81	0.60	129
4.0	0.67	0.62	0.64	115
accuracy			0.54	334
macro avg	0.49	0.37	0.34	334
weighted avg	0.60	0.54	0.48	334

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.00	0.00	0.00	19
3.0	0.43	0.75	0.55	32
4.0	0.60	0.52	0.56	29
accuracy			0.46	84
macro avg	0.26	0.32	0.28	84
weighted avg	0.37	0.46	0.40	84

For Fold 2 the accuracy is **47.61904761904761**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	recall	f1-score	support
1.0	0.41	0.41	0.41	17
2.0	0.00	0.00	0.00	73
3.0	0.48	0.87	0.62	129
4.0	0.72	0.53	0.61	115
accuracy			0.54	334
macro avg	0.40	0.45	0.41	334
weighted avg	0.45	0.54	0.47	334

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	recall	f1-score	support
1.0	0.20	0.25	0.22	4
2.0	0.00	0.00	0.00	19
3.0	0.46	0.81	0.58	32
4.0	0.59	0.45	0.51	29
accuracy			0.48	84
macro avg	0.31	0.38	0.33	84
weighted avg	0.39	0.48	0.41	84

For Fold 3 the accuracy is **51.19047619047619**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	recall	f1-score	support
1.0	0.00	0.00	0.00	16
2.0	0.00	0.00	0.00	74
3.0	0.49	0.65	0.56	129
4.0	0.57	0.80	0.66	115
accuracy			0.53	334
macro avg	0.26	0.36	0.31	334
weighted avg	0.38	0.53	0.44	334

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	recall	f1-score	support
1.0	0.00	0.00	0.00	5
2.0	0.00	0.00	0.00	18
3.0	0.46	0.72	0.56	32
4.0	0.59	0.69	0.63	29

accuracy			0.51	84
macro avg	0.26	0.35	0.30	84
weighted avg	0.38	0.51	0.43	84

For Fold 4 the accuracy is **49.39759036144578**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	17
2.0	0.00	0.00	0.00	74
3.0	0.49	0.71	0.58	129
4.0	0.58	0.76	0.66	115

accuracy			0.53	335
macro avg	0.27	0.37	0.31	335
weighted avg	0.39	0.53	0.45	335

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.00	0.00	0.00	18
3.0	0.44	0.50	0.47	32
4.0	0.53	0.86	0.66	29

accuracy			0.49	83
macro avg	0.24	0.34	0.28	83
weighted avg	0.36	0.49	0.41	83

For Fold 5 the accuracy is **43.373493975903614**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	17
2.0	0.50	0.30	0.37	74
3.0	0.49	0.71	0.58	128
4.0	0.68	0.62	0.65	116

accuracy			0.55	335
macro avg	0.42	0.41	0.40	335
weighted avg	0.53	0.55	0.53	335

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	recall	f1-score	support
1.0	0.00	0.00	0.00	4
2.0	0.14	0.06	0.08	18
3.0	0.43	0.67	0.52	33
4.0	0.52	0.46	0.49	28
accuracy			0.43	83
macro avg	0.27	0.30	0.27	83
weighted avg	0.38	0.43	0.39	83

profondeur = 3

For Fold 1 the accuracy is **47.61904761904761**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	recall	f1-score	support
1.0	1.00	0.18	0.30	17
2.0	1.00	0.07	0.13	73
3.0	0.48	0.81	0.60	129
4.0	0.67	0.63	0.65	115
accuracy			0.55	334
macro avg	0.79	0.42	0.42	334
weighted avg	0.69	0.55	0.50	334

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	recall	f1-score	support
1.0	0.00	0.00	0.00	4
2.0	0.00	0.00	0.00	19
3.0	0.43	0.75	0.55	32
4.0	0.59	0.55	0.57	29
accuracy			0.48	84
macro avg	0.26	0.33	0.28	84
weighted avg	0.37	0.48	0.41	84

For Fold 2 the accuracy is **48.80952380952381**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	recall	f1-score	support
1.0	0.83	0.29	0.43	17
2.0	0.49	0.36	0.41	73
3.0	0.52	0.74	0.62	129
4.0	0.73	0.58	0.65	115

accuracy			0.58	334
macro avg	0.64	0.49	0.53	334
weighted avg	0.60	0.58	0.57	334

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.50	0.25	0.33	4
2.0	0.18	0.11	0.13	19
3.0	0.52	0.78	0.62	32
4.0	0.57	0.45	0.50	29

accuracy			0.49	84
macro avg	0.44	0.40	0.40	84
weighted avg	0.46	0.49	0.46	84

For Fold 3 the accuracy is **48.80952380952381**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	1.00	0.19	0.32	16
2.0	0.00	0.00	0.00	74
3.0	0.49	0.77	0.60	129
4.0	0.63	0.71	0.67	115

accuracy			0.55	334
macro avg	0.53	0.42	0.40	334
weighted avg	0.45	0.55	0.48	334

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	**recall**	**f1-score**	**support**
1.0	1.00	0.20	0.33	5
2.0	0.00	0.00	0.00	18
3.0	0.45	0.81	0.58	32
4.0	0.56	0.48	0.52	29

accuracy			0.49	84
macro avg	0.50	0.37	0.36	84
weighted avg	0.42	0.49	0.42	84

For Fold 4 the accuracy is **48.19277108433735**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	recall	f1-score	support
1.0	1.00	0.24	0.38	17
2.0	0.80	0.05	0.10	74
3.0	0.50	0.78	0.61	129
4.0	0.65	0.70	0.67	115
accuracy			0.56	335
macro avg	0.74	0.44	0.44	335
weighted avg	0.64	0.56	0.51	335

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	recall	f1-score	support
1.0	0.00	0.00	0.00	4
2.0	0.25	0.06	0.09	18
3.0	0.43	0.62	0.51	32
4.0	0.59	0.66	0.62	29
accuracy			0.48	83
macro avg	0.32	0.33	0.31	83
weighted avg	0.43	0.48	0.43	83

For Fold 5 the accuracy is **44.57831325301205**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	recall	f1-score	support
1.0	1.00	0.18	0.30	17
2.0	0.51	0.47	0.49	74
3.0	0.55	0.73	0.62	128
4.0	0.74	0.60	0.67	116
accuracy			0.60	335
macro avg	0.70	0.49	0.52	335
weighted avg	0.63	0.60	0.59	335

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	recall	f1-score	support
1.0	1.00	0.25	0.40	4
2.0	0.19	0.17	0.18	18
3.0	0.49	0.67	0.56	33
4.0	0.52	0.39	0.45	28
accuracy			0.45	83
macro avg	0.55	0.37	0.40	83

weighted avg	0.46	0.45	0.43	83

pause k

profondeur = 4

For Fold 1 the accuracy is **51.19047619047619**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	1.00	0.18	0.30	17
2.0	0.71	0.16	0.27	73
3.0	0.54	0.68	0.60	129
4.0	0.60	0.78	0.68	115
accuracy			0.58	334
macro avg	0.71	0.45	0.46	334
weighted avg	0.62	0.58	0.54	334

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.57	0.21	0.31	19
3.0	0.50	0.59	0.54	32
4.0	0.51	0.69	0.59	29
accuracy			0.51	84
macro avg	0.40	0.37	0.36	84
weighted avg	0.50	0.51	0.48	84

For Fold 2 the accuracy is **45.23809523809524**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.77	0.59	0.67	17
2.0	0.59	0.41	0.48	73
3.0	0.59	0.68	0.63	129
4.0	0.72	0.76	0.74	115
accuracy			0.64	334
macro avg	0.67	0.61	0.63	334
weighted avg	0.64	0.64	0.64	334

Paramètres de precision rappel et accuracy du model Arbre pour test

precision	**recall**	**f1-score**	**support**

1.0	0.25	0.25	0.25	4
2.0	0.21	0.16	0.18	19
3.0	0.51	0.66	0.58	32
4.0	0.52	0.45	0.48	29
accuracy			0.45	84
macro avg	0.37	0.38	0.37	84
weighted avg	0.44	0.45	0.44	84

For Fold 3 the accuracy is **48.80952380952381**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	1.00	0.19	0.32	16
2.0	0.43	0.58	0.49	74
3.0	0.60	0.63	0.62	129
4.0	0.75	0.63	0.68	115
accuracy			0.60	334
macro avg	0.70	0.51	0.53	334
weighted avg	0.63	0.60	0.60	334

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	**recall**	**f1-score**	**support**
1.0	1.00	0.20	0.33	5
2.0	0.29	0.33	0.31	18
3.0	0.47	0.69	0.56	32
4.0	0.80	0.41	0.55	29
accuracy			0.49	84
macro avg	0.64	0.41	0.44	84
weighted avg	0.58	0.49	0.49	84

For Fold 4 the accuracy is **45.78313253012048**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	1.00	0.24	0.38	17
2.0	0.46	0.68	0.55	74
3.0	0.61	0.65	0.63	129
4.0	0.78	0.58	0.67	115
accuracy			0.61	335
macro avg	0.71	0.54	0.56	335

weighted avg	0.66	0.61	0.61	335

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.38	0.50	0.43	18
3.0	0.41	0.44	0.42	32
4.0	0.60	0.52	0.56	29

accuracy			0.46	83
macro avg	0.35	0.36	0.35	83
weighted avg	0.45	0.46	0.45	83

For Fold 5 the accuracy is **42.168674698795186**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	1.00	0.18	0.30	17
2.0	0.65	0.38	0.48	74
3.0	0.62	0.70	0.66	128
4.0	0.64	0.78	0.70	116

accuracy			0.63	335
macro avg	0.73	0.51	0.53	335
weighted avg	0.65	0.63	0.62	335

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	**recall**	**f1-score**	**support**
1.0	1.00	0.25	0.40	4
2.0	0.38	0.17	0.23	18
3.0	0.43	0.48	0.46	33
4.0	0.41	0.54	0.46	28

accuracy			0.42	83
macro avg	0.55	0.36	0.39	83
weighted avg	0.44	0.42	0.41	83

pause k

profondeur = 5

For Fold 1 the accuracy is **47.61904761904761**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	**recall**	**f1-score**	**support**

1.0	1.00	0.18	0.30	17
2.0	0.74	0.23	0.35	73
3.0	0.56	0.84	0.67	129
4.0	0.75	0.74	0.75	115
accuracy			0.64	334
macro avg	0.76	0.50	0.52	334
weighted avg	0.69	0.64	0.61	334

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.30	0.16	0.21	19
3.0	0.46	0.72	0.56	32
4.0	0.58	0.48	0.53	29
accuracy			0.48	84
macro avg	0.34	0.34	0.32	84
weighted avg	0.44	0.48	0.44	84

For Fold 2 the accuracy is **47.61904761904761**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	1.00	0.59	0.74	17
2.0	0.68	0.49	0.57	73
3.0	0.64	0.81	0.71	129
4.0	0.82	0.76	0.79	115
accuracy			0.71	334
macro avg	0.78	0.66	0.70	334
weighted avg	0.73	0.71	0.71	334

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.50	0.25	0.33	4
2.0	0.38	0.26	0.31	19
3.0	0.49	0.66	0.56	32
4.0	0.50	0.45	0.47	29
accuracy			0.48	84
macro avg	0.47	0.40	0.42	84
weighted avg	0.47	0.48	0.46	84

For Fold 3 the accuracy is **51.19047619047619**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	recall	f1-score	support
1.0	1.00	0.19	0.32	16
2.0	0.61	0.51	0.56	74
3.0	0.57	0.71	0.63	129
4.0	0.77	0.72	0.74	115
accuracy			0.65	334
macro avg	0.74	0.53	0.56	334
weighted avg	0.67	0.65	0.64	334

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	recall	f1-score	support
1.0	1.00	0.20	0.33	5
2.0	0.33	0.17	0.22	18
3.0	0.45	0.75	0.56	32
4.0	0.71	0.52	0.60	29
accuracy			0.51	84
macro avg	0.63	0.41	0.43	84
weighted avg	0.55	0.51	0.49	84

For Fold 4 the accuracy is **43.3734939759036l4**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	recall	f1-score	support
1.0	0.83	0.29	0.43	17
2.0	0.76	0.46	0.57	74
3.0	0.57	0.90	0.70	129
4.0	0.87	0.62	0.72	115
accuracy			0.67	335
macro avg	0.76	0.57	0.61	335
weighted avg	0.73	0.67	0.67	335

	precision	recall	f1-score	support
1.0	0.00	0.00	0.00	4
2.0	0.39	0.39	0.39	18
3.0	0.38	0.44	0.41	32
4.0	0.54	0.52	0.53	29
accuracy			0.43	83
macro avg	0.33	0.34	0.33	83

weighted avg	0.42	0.43	0.42	83

For Fold 5 the accuracy is **48.19277108433735**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	1.00	0.18	0.30	17
2.0	0.76	0.47	0.58	74
3.0	0.63	0.89	0.74	128
4.0	0.79	0.72	0.75	116

accuracy			0.70	335
macro avg	0.80	0.56	0.59	335
weighted avg	0.73	0.70	0.69	335

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	**recall**	**f1-score**	**support**
1.0	1.00	0.25	0.40	4
2.0	0.36	0.28	0.31	18
3.0	0.49	0.64	0.55	33
4.0	0.52	0.46	0.49	28

accuracy			0.48	83
macro avg	0.59	0.41	0.44	83
weighted avg	0.50	0.48	0.47	83

pause k

profondeur = 6

For Fold 1 the accuracy is **39.285714285714285**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.80	0.24	0.36	17
2.0	0.92	0.30	0.45	73
3.0	0.61	0.89	0.72	129
4.0	0.76	0.77	0.76	115

accuracy			0.69	334
macro avg	0.77	0.55	0.58	334
weighted avg	0.74	0.69	0.66	334

Paramètres de precision rappel et accuracy du model Arbre pour test

precision	**recall**	**f1-score**	**support**

1.0	0.00	0.00	0.00	4
2.0	0.27	0.16	0.20	19
3.0	0.40	0.59	0.48	32
4.0	0.44	0.38	0.41	29
accuracy			0.39	84
macro avg	0.28	0.28	0.27	84
weighted avg	0.37	0.39	0.37	84

For Fold 2 the accuracy is **46.42857142857143**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	1.00	0.65	0.79	17
2.0	0.83	0.52	0.64	73
3.0	0.70	0.87	0.77	129
4.0	0.84	0.84	0.84	115
accuracy			0.77	334
macro avg	0.84	0.72	0.76	334
weighted avg	0.79	0.77	0.77	334

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.33	0.25	0.29	4
2.0	0.27	0.21	0.24	19
3.0	0.47	0.62	0.53	32
4.0	0.61	0.48	0.54	29
accuracy			0.46	84
macro avg	0.42	0.39	0.40	84
weighted avg	0.46	0.46	0.46	84

For Fold 3 the accuracy is **48.80952380952381**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	1.00	0.19	0.32	16
2.0	0.93	0.38	0.54	74
3.0	0.65	0.88	0.75	129
4.0	0.76	0.83	0.80	115
accuracy			0.72	334
macro avg	0.84	0.57	0.60	334

weighted avg	0.77	0.72	0.70	334

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	**recall**	**f1-score**	**support**
1.0	1.00	0.20	0.33	5
2.0	0.20	0.06	0.09	18
3.0	0.44	0.75	0.56	32
4.0	0.62	0.52	0.57	29

accuracy			0.49	84
macro avg	0.57	0.38	0.39	84
weighted avg	0.49	0.49	0.45	84

For Fold 4 the accuracy is **49.39759036144578**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	1.00	0.41	0.58	17
2.0	0.75	0.62	0.68	74
3.0	0.67	0.84	0.75	129
4.0	0.83	0.75	0.79	115

accuracy			0.74	335
macro avg	0.81	0.66	0.70	335
weighted avg	0.76	0.74	0.74	335

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.36	0.28	0.31	18
3.0	0.46	0.50	0.48	32
4.0	0.65	0.69	0.67	29

accuracy			0.49	83
macro avg	0.36	0.37	0.36	83
weighted avg	0.48	0.49	0.48	83

For Fold 5 the accuracy is **49.39759036144578**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.71	0.29	0.42	17
2.0	0.78	0.64	0.70	74

3.0	0.70	0.88	0.78	128
4.0	0.82	0.76	0.79	116
accuracy			0.76	335
macro avg	0.76	0.64	0.67	335
weighted avg	0.76	0.76	0.76	335

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.50	0.25	0.33	4
2.0	0.42	0.28	0.33	18
3.0	0.50	0.64	0.56	33
4.0	0.52	0.50	0.51	28
accuracy			0.49	84
macro avg	0.48	0.42	0.43	83
weighted avg	0.49	0.49	0.49	83

pause k

profondeur = 7

For Fold 1 the accuracy is **34.523809523809526**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	1.00	0.35	0.52	17
2.0	0.79	0.51	0.62	73
3.0	0.69	0.86	0.76	129
4.0	0.82	0.84	0.83	115
accuracy			0.75	334
macro avg	0.82	0.64	0.68	334
weighted avg	0.77	0.75	0.74	334

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.27	0.21	0.24	19
3.0	0.38	0.47	0.42	32
4.0	0.36	0.34	0.35	29
accuracy			0.35	84
macro avg	0.25	0.26	0.25	84
weighted avg	0.33	0.35	0.34	84

For Fold 2 the accuracy is **44.047619047619044**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	1.00	0.65	0.79	17
2.0	0.80	0.71	0.75	73
3.0	0.82	0.88	0.85	129
4.0	0.87	0.90	0.89	115
accuracy			0.84	334
macro avg	0.87	0.78	0.84	334
weighted avg	0.84	0.84	0.84	334

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.20	0.25	0.22	4
2.0	0.25	0.26	0.26	19
3.0	0.47	0.53	0.50	32
4.0	0.61	0.48	0.54	29
accuracy			0.44	84
macro avg	0.38	0.38	0.38	84
weighted avg	0.46	0.44	0.44	84

For Fold 3 the accuracy is **47.61904761904761**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.69	0.56	0.62	16
2.0	0.88	0.50	0.64	74
3.0	0.77	0.94	0.85	129
4.0	0.84	0.90	0.87	115
accuracy			0.81	334
macro avg	0.80	0.72	0.74	334
weighted avg	0.82	0.81	0.80	334

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.25	0.20	0.22	5
2.0	0.40	0.11	0.17	18
3.0	0.44	0.69	0.54	32
4.0	0.60	0.52	0.56	29

accuracy			0.48	84
macro avg	0.42	0.38	0.37	84
weighted avg	0.48	0.48	0.45	83

For Fold 4 the accuracy is **44.57831325301205**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.89	0.47	0.62	17
2.0	0.83	0.73	0.78	74
3.0	0.73	0.91	0.81	129
4.0	0.89	0.78	0.83	115

accuracy			0.80	335
macro avg	0.84	0.72	0.76	335
weighted avg	0.82	0.80	0.80	335

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.26	0.28	0.27	18
3.0	0.44	0.44	0.44	32
4.0	0.58	0.62	0.60	29

accuracy			0.45	83
macro avg	0.32	0.33	0.33	83
weighted avg	0.43	0.45	0.44	83

For Fold 5 the accuracy is **42.168674698795186**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.78	0.41	0.54	17
2.0	0.79	0.80	0.79	74
3.0	0.79	0.83	0.81	128
4.0	0.82	0.83	0.82	116

accuracy			0.80	335
macro avg	0.79	0.72	0.74	335
weighted avg	0.80	0.80	0.80	335

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	recall	f1-score	support
1.0	1.00	0.25	0.40	4
2.0	0.31	0.28	0.29	18
3.0	0.48	0.42	0.45	33
4.0	0.41	0.54	0.46	28
accuracy			0.42	83
macro avg	0.55	0.37	0.40	83
weighted avg	0.44	0.42	0.42	83

pause k

profondeur = 8

For Fold 1 the accuracy is **34.523809523809526**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	recall	f1-score	support
1.0	0.79	0.65	0.71	17
2.0	0.76	0.71	0.74	73
3.0	0.79	0.82	0.81	129
4.0	0.87	0.90	0.88	115
accuracy			0.81	334
macro avg	0.80	0.77	0.78	334
weighted avg	0.81	0.81	0.81	334

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	recall	f1-score	support
1.0	0.00	0.00	0.00	4
2.0	0.32	0.37	0.34	19
3.0	0.34	0.31	0.33	32
4.0	0.41	0.41	0.41	29
accuracy			0.35	84
macro avg	0.27	0.27	0.27	84
weighted avg	0.35	0.35	0.34	83

For Fold 2 the accuracy is **45.23809523809524**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	recall	f1-score	support
1.0	0.92	0.71	0.80	17
2.0	0.78	0.89	0.83	73
3.0	0.91	0.87	0.89	129
4.0	0.93	0.93	0.93	115

accuracy			0.89	334
macro avg	0.89	0.85	0.86	334
weighted avg	0.89	0.89	0.89	334

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.33	0.25	0.29	4
2.0	0.29	0.32	0.30	19
3.0	0.47	0.53	0.50	32
4.0	0.58	0.48	0.53	29

accuracy			0.45	84
macro avg	0.42	0.39	0.40	84
weighted avg	0.46	0.45	0.45	84

For Fold 3 the accuracy is **45.23809523809524**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.85	0.69	0.76	16
2.0	0.79	0.78	0.79	74
3.0	0.87	0.91	0.89	129
4.0	0.94	0.91	0.93	115

accuracy			0.87	334
macro avg	0.86	0.82	0.84	334
weighted avg	0.87	0.87	0.87	334

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.50	0.20	0.29	5
2.0	0.14	0.11	0.12	18
3.0	0.44	0.62	0.52	32
4.0	0.65	0.52	0.58	29

accuracy			0.45	84
macro avg	0.43	0.36	0.38	84
weighted avg	0.45	0.45	0.44	84

For Fold 4 the accuracy is **45.78313253012048**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	recall	f1-score	support
1.0	1.00	0.53	0.69	17
2.0	0.95	0.76	0.84	74
3.0	0.79	0.95	0.86	129
4.0	0.91	0.88	0.89	115
accuracy			0.86	335
macro avg	0.91	0.78	0.82	335
weighted avg	0.88	0.86	0.86	335

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	recall	f1-score	support
1.0	0.00	0.00	0.00	4
2.0	0.29	0.28	0.29	18
3.0	0.48	0.47	0.48	32
4.0	0.56	0.62	0.59	29
accuracy			0.46	83
macro avg	0.34	0.34	0.34	83
weighted avg	0.45	0.46	0.45	83

For Fold 5 the accuracy is **44.57831325301205**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	recall	f1-score	support
1.0	0.79	0.65	0.71	17
2.0	0.87	0.84	0.86	74
3.0	0.77	0.95	0.85	128
4.0	0.98	0.77	0.86	116
accuracy			0.85	335
macro avg	0.85	0.80	0.82	335
weighted avg	0.86	0.85	0.85	335

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	recall	f1-score	support
1.0	0.25	0.25	0.25	4
2.0	0.38	0.28	0.32	18
3.0	0.48	0.61	0.53	33
4.0	0.46	0.39	0.42	28
accuracy			0.45	83
macro avg	0.39	0.38	0.38	83

weighted avg	0.44	0.45	0.44	83

pause k

profondeur = 9

For Fold 1 the accuracy is **34.523809523809526**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	1.00	0.65	0.79	17
2.0	0.96	0.74	0.84	73
3.0	0.80	0.93	0.86	129
4.0	0.91	0.93	0.92	115
accuracy			0.87	334
macro avg	0.92	0.81	0.85	334
weighted avg	0.89	0.87	0.87	334

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.35	0.32	0.33	19
3.0	0.36	0.50	0.42	32
4.0	0.32	0.24	0.27	29
accuracy			0.35	84
macro avg	0.26	0.26	0.26	84
weighted avg	0.33	0.35	0.33	84

For Fold 2 the accuracy is **35.714285714285715**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.94	0.88	0.91	17
2.0	0.97	0.88	0.92	73
3.0	0.91	0.96	0.93	129
4.0	0.94	0.94	0.94	115
accuracy			0.93	334
macro avg	0.94	0.91	0.93	334
weighted avg	0.93	0.93	0.93	334

Paramètres de precision rappel et accuracy du model Arbre pour test

precision	**recall**	**f1-score**	**support**

1.0	0.20	0.25	0.22	4
2.0	0.20	0.26	0.23	19
3.0	0.39	0.44	0.41	32
4.0	0.56	0.34	0.43	29
accuracy			0.36	84
macro avg	0.34	0.32	0.32	84
weighted avg	0.39	0.36	0.37	84

For Fold 3 the accuracy is **47.61904761904761**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.88	0.88	0.88	16
2.0	0.95	0.82	0.88	16
3.0	0.89	0.97	0.93	129
4.0	0.96	0.95	0.96	115
accuracy			0.93	334
macro avg	0.92	0.90	0.91	334
weighted avg	0.93	0.93	0.92	334

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.50	0.20	0.29	5
2.0	0.33	0.33	0.33	18
3.0	0.45	0.59	0.51	32
4.0	0.64	0.48	0.55	29
accuracy			0.48	84
macro avg	0.48	0.40	0.42	84
weighted avg	0.49	0.48	0.47	84

For Fold 4 the accuracy is **44.57831325301205**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	1.00	0.71	0.83	17
2.0	0.98	0.78	0.87	74
3.0	0.81	0.98	0.88	129
4.0	0.94	0.89	0.91	115
accuracy			0.89	335

macro avg	0.93	0.84	0.87	335
weighted avg	0.90	0.89	0.89	335

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.00	0.00	0.00	4
2.0	0.31	0.28	0.29	18
3.0	0.44	0.47	0.45	32
4.0	0.53	0.59	0.56	29

accuracy			0.45	83
macro avg	0.32	0.33	0.33	83
weighted avg	0.42	0.45	0.43	83

For Fold 5 the accuracy is **37.34939759036144**

Paramètres de precision rappel et accuracy du model Arbre pour apprentissage

	precision	**recall**	**f1-score**	**support**
1.0	0.93	0.76	0.84	17
2.0	0.95	0.85	0.90	74
3.0	0.84	0.96	0.90	128
4.0	0.94	0.89	0.92	116

accuracy			0.90	335
macro avg	0.92	0.87	0.89	335
weighted avg	0.91	0.90	0.90	335

Paramètres de precision rappel et accuracy du model Arbre pour test

	precision	**recall**	**f1-score**	**support**
1.0	0.33	0.25	0.29	4
2.0	0.27	0.22	0.24	18
3.0	0.42	0.39	0.41	33
4.0	0.38	0.46	0.42	28

accuracy			0.37	83
macro avg	0.35	0.33	0.34	83
weighted avg	0.37	0.37	0.73	83