République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Mouloud MAMMERI de Tizi-Ouzou

Faculté de Génie Electrique et Informatique

Département Informatique

MEMOIRE DE FIN D'ETUDES

En vue de l'obtention d'un diplôme de Master en Informatique

Spécialité : Ingénierie des Systèmes d'Informations

THEME :

Application des techniques de datamining pour la classification automatique des données

Proposé et dirigé par :

M^me.^FELLAG.S

Réalisé par:

ARARBI Assia

AOUAKLI Farida

Promotion : 2020 / 2021

Remerciements

*Nos louanges vont vers **ALLAH** qui nous a guidés et aidés pour L'accomplissement de ce travail.*

Nous tenons particulièrement à remercier notre promotrice :

***M^me^ FELLAG** pour son soutien, ses conseils et son aide précieuse durant toute la période du travail.*

Nos vifs remerciements vont également aux membres du jury pour L'intérêt qu'ils ont porté à notre travail en acceptant de l'examiner et de l'enrichir par leurs propositions.

Nous remercions tous les enseignants qui ont contribué à notre formation.

Nous remercions aussi nos familles à qui on doit ce que nous somme aujourd'hui grâce à leur amour, patience et leurs innombrables sacrifices

Nous voudrions aussi exprimer notre gratitude envers tous nos amis, collègues et toutes les personnes qui nous ont accordé leur soutien, tant par leur gentillesse

que par leur dévouement.

MERCI

Dédicace :

بسم الله الرحمن الرحيم

Merci Allah de m'avoir donné la capacité d'écrire et de réfléchir, la force d'y croire, la patience d'aller jusqu'au bout du rêve. Je tiens à dédier ce modeste travail :

A mes chers parents, aucune dédicace ne saurait exprimer mon respect, mon amour éternel et ma considération pour les sacrifices que vous avez consenti pour mon instruction et mon bien être. Que ce modeste travail soit l'exaucement de vos vœux tant formulés, le fruit de vos innombrables sacrifices, bien que je ne vous en acquitterai jamais assez.

A mes chères sœurs Samira, Djimi et à mon chère frère Amine, ceux qui ont partagé avec moi tous les moments d'émotion lors de la réalisation de ce travail. Ils m'ont chaleureusement supporté et encouragé tout au long de mon parcours.

A mes très chères cousines Amina, Chabha , à mes chers cousins Ghani , Vegas, Djoudi ,Smail et à ma chère tente MESSAID Djedjiga qui ont été toujours un pilier pour moi.

A tous les membres de la famille « ARARBI» et « GACEM», que dieu les protège et leurs offre la chance et le bonheur.

A mes chers neveux Said ,Idir et à mes nièces Ritaj et Rinad, source de joie et de bonheur .

A mes chères amies Houda ,Narimane ,Nesrine, lydia, Mellissa,Meriem, rosa, feriel, Tina, sihem, Dyhia ,lamia, sarah, Lynda ,Nedjma, Lynda ,qui m'ont toujours encouragé et à qui je souhaite plus de succès .

A tous mes amis de la promotion ISI.

A la personne qui a partagé tous le travail, ma chère binôme Aouakli Farida pour son soutien moral et sa patience tout au long de ce travail, je te souhaite une vie pleine de bonheur et de succès et que Dieu, le tout puissant, te protège et te garde.

Dédicace :

بسم الله الرحمن الرحيم

Je dédie ce travail,

À mes parents,

À celle qui m'a toujours poussé à aller de l'avant, mon exemple de courage, à la mémoire de ma chère mère,

À mon père, qui a toujours cru en moi, m'a soutenu inconditionnellement,

Je ne vous remercierais jamais assez.

À mes sœurs Rachida, Fatma et Karima, mon frère Ali, mes piliers dans cette vie,

À mes nièces adorées,

L'amour que je porte pour vous dépasse les mots,

A la personne avec qui j'ai partagé ce travail, ma chère binôme Ararbi Assia, je te souhaite bonheur et succès et que Dieu, le tout puissant, te protège et te garde.
A mes amies, qui ont toujours été à me cotées.

Farida .

Table des matières :

Chapitre 2 : Les techniques de la classification supervisée :

Chapitre 3 : Description de l'application :

Chapitre 4 : Tests et résultats :

Table des figures :

Chapitre1 :

Chapitre2 :

Chapitre3:

Chapitre4:

Table des tableaux :

Chapitre2 :

Chapitre3:

Chapitre4:

Résumé :

Le Data Mining est une technologie dont le but est la valorisation de l'information et l'extraction de connaissances d'un grand nombre de données. Cette technologie est devenue un outil important pour améliorer les revenus des entreprises qui rencontrent des problèmes dus à la quantité énorme de données, nous avons consacrer notre mémoire à l'application des algorithmes de classification supervisée de data mining à savoir KNN, ID3, C4.5, CART et les réseaux de neurones simples et convolutifs.

En premier temps nous avons implémenté les algorithmes que nous avons testés sur des jeux de données réels, ensuite évalué sur les métriques suivantes : matrice de confusion, rappel, précision, justesse, f1_score et le coefficient de Jaccard.

Les résultats d'évaluation des algorithmes ont confirmé qu'on ne peut pas juger la performance d'un algorithme, cela dépend de caractéristiques du jeu de données sur lequel on va le tester.

Mots clés : Data mining, algorithmes de classification supervisée.

Abstract:

Data Mining is a technology whose goal is the valuation of information and knowledge extraction from a large amount of data. This technology become an important tool to improve the income of companies that encounter problems due to the huge amount of data, we have devoted our memory to the application of supervised classification algorithms of data mining namely: KNN, ID3, C4.5, CART and simple and convolutional neural networks.

First, we implemented the algorithms that we tested on real data sets, then evaluated by the following metrics: confusion matrix, recall, precision, accuracy, f1_score and the Jaccard coefficient.

The algorithm evaluation results confirmed that one cannot judge the performance of an algorithm, it depends on the characteristics of the dataset on which it is going to be tested.

Keywords:

Data mining, supervised classification algorithms.

Introduction générale :

Le pétrole a longtemps dominé les débats 20ème siècle, mais les données l'ont surpassé en se qualifiant d'« or noir du 21ème siècle », En effet, l'exploitation des données présente des richesses capitales qui accompagne le développement des sociétés.

L'**IOT**[1], Le **Big Data** [2]et l'avènement de la **5G**[3], ces technologies émergentes qui prennent des formes de plus en plus variées et touchent désormais des domaines très divers comme la domotique, les transports, l'e-santé, le multimédia ou encore les loisirs, et font en sorte que le volume des informations collectées accroit de manière exponentielle.

Les défis sont nombreux pour un usage rentable de l'information. Quelles sont les données utiles, comment les reconnaître et les séparer de celles qui n'ont pas valeurs ? Jusqu'à quel point peuvent-elles servir, aussi bien les administrations et organisations publiques que les entreprises privées ?

Ces données peuvent être exploitées pour faire la prédiction et l'aide à la prise de décision, grâce à la classification automatique supervisée qui est une des tâches principales du data mining, qui est au cœur du processus d'extraction des connaissances à partir des données, appelé ECD ou encore KDD (knowldge data) en anglais.

Le data mining apporte des solutions pour l'extraction de connaissances à partir de données brutes, il regroupe un ensemble d'algorithmes et techniques destinés à l'exploration et l'analyse des grandes bases de données en vue de détecter des règles, des associations, des tendances inconnues non fixés à priori.

Nous nous intéressons dans notre travail aux techniques de classification supervisée du data mining, notre travail consiste à implémenter un ensemble d'algorithmes de classification, ensuite, de les évaluer par la suite sur des collections de données en utilisant des métriques d'évaluation et faire une étude comparative.

La démarche générale s'articule autour de quatre étapes partagées en quatre chapitres :

---

[1] **IOT** : (internet of things) caractérise des objets physiques connectés ayant leur propre identité numérique et capables de communiquer les uns avec les autres. Ce réseau crée en quelque sorte une passerelle entre le monde physique et le monde virtuel.

[2] **Big Data** : des ensembles de très gros volumes de données personnelles générées par les connexions internet et par les connexions avec et entre les objets connectés.

[3] **5G** : un débit internet dix fois plus rapide et plus performant que la 4G.

Nous avons introduit dans le premier chapitre le processus ECD, le data mining qui est au cœur de ce processus, ainsi que ces tâches et ses différentes techniques.

Nous avons détaillé dans le second chapitre les différentes techniques de la classification automatique supervisée.

Le troisième chapitre a été consacré à l'implémentation des algorithmes de classification supervisée, KNN, ID3, C4.5, CART et les réseaux de neurones.

Dans le quatrième et dernier chapitre, nous avons présenté les jeux de données sur lesquels ont été testés les algorithmes, les métriques qui nous ont servi pour faire l'évaluation et enfin les résultats et l'évaluation des algorithmes implémentés.

Chapitre1:

Le processus de KDD et le Data Mining

Introduction :

La technologie de Data mining représente un mélange d'idées et d'outils provenant des Statistiques, l'Intelligence artificielle et l'Informatique, permet aux entreprises et aux organisations de différents domaines de traiter des quantités gigantesques de données, afin d'extraire des connaissances, des informations non connues précédemment et potentiellement utiles pour la prédiction et l'aide à la prise de décision.

La révolution numérique et celle des outils de collecte automatique des données conduisent à d'énormes masses de données stockées dans des **Data Warehouses**[4], submergés par les données, le data mining est donc venu comme une solution pour l'analyse et le traitement de ces mines d'informations.

Le data mining est l'un des maillons de la chaine de traitement pour l'extraction de connaissances à partir de données (ECD). On pourrait dire que l'ECD est un véhicule dont le data mining est le moteur **[1].**

Le data mining regroupe un ensemble de techniques réparties en trois catégories : La classification supervisée, le clustering et les règles d'association.

Nous allons dans ce chapitre présenter le processus ECD ainsi que l'ensemble des concepts de base de data mining ensuite ses tâches et enfin ses différentes techniques.

[4]**Data Warehouse** : C'est une structure (semblable à une base de données) comportant un tas d'informations épurées, organisées, historiées et provenant de plusieurs sources de données, servant aux analyses et à l'aide à la décision. Data warehouse permet une vision centralisée et universelle de toutes les informations de l'entreprise.

1. Processus d'extraction de connaissances à partir de données (ECD) :

C'est un processus itératif et interactif d'analyse d'un grand ensemble de données brutes afin d'en extraire des connaissances exploitables par un utilisateur analyste qui y joue un rôle central. Il se déroule suivant une suite d'opérations. Comme le montre la figure 1.

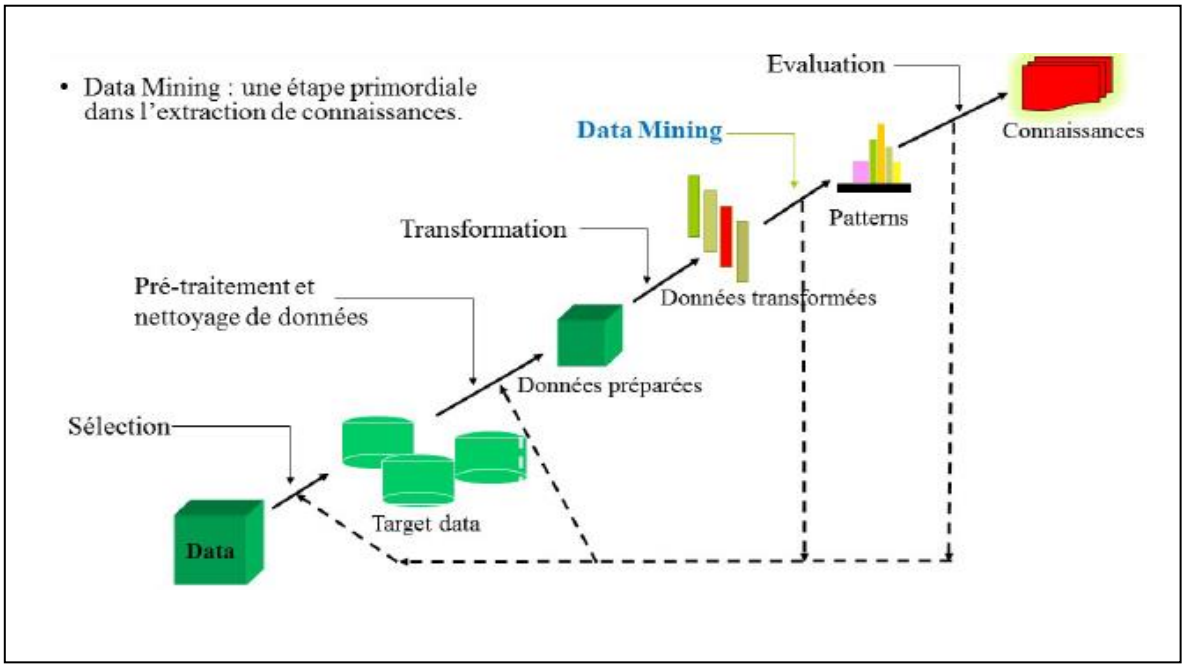

Figure 1: Processus d'extraction de connaissances à partir de données [2]

1.1. Les étapes de processus ECD :

Le processus ECD peut être vu comme un processus en sept étapes :

1. Positionnement du problème.
2. Création d'un ensemble de données (sélection).
3. Nettoyage et prétraitement de données (peut prendre 60% de l'effort).
4. Transformation.
5. Data mining.
6. Evaluation des résultats.
7. Intégration de connaissances.

1.1.1. Positionnement du problème :

Il s'agit de comprendre le contexte de la recherche pour donner une signification logique aux variables de sa problématique. Dans cette étape, toutes les variables susceptibles d'expliquer le phénomène étudié sont identifiées. Une première approche consiste à reformuler le problème pour qu'il puisse être traité par les techniques et les outils de modélisation.

Une autre approche, des plus efficaces, consiste à découper le problème complexe en sous problèmes plus simples et à traiter, indépendamment, chacun de ces sous-problèmes.

1.1.2. Création d'un ensemble de données (sélection) :

Sélectionnez un ensemble de données cible ou un sous-ensemble d'échantillons de données sur lesquels la découverte doit être effectuée. On procède d'abord par une collecte de données. Ces dernières se trouvant généralement dans des bases de données ou dans des Data Warehouses. Une sélection optimale des données nécessite souvent l'aide d'experts du domaine pour déterminer les facteurs, ou variables, les plus aptes à décrire la problématique. Dans le cas où les experts ne sont pas disponibles, une recherche des variables les plus déterminantes effectuée par des techniques d'analyse statistiques.

1.1.3. Nettoyage et prétraitement de données (60% de l'effort) :

De multiples sources de données, souvent hétérogènes, seront combinées dans une seule structure lors de l'étape d'intégration, dans cette étape seront nettoyées et prétraitées les données en décidant des stratégies pour gérer les champs manquants et modifier les données conformément aux exigences. Nous nous intéressons à l'examen de la qualité des données collectées. Ils y seront traités des problèmes courants tels que les doublons, les erreurs de saisie, l'intégrité de données et le problème des valeurs manquantes.

1.1.4. Transformation :

Des ajustements consistant à remanier les valeurs de certains attributs, afin de faciliter les calculs, seront effectués dans l'étape de transformation. Cette dernière inclut notamment des opérations de regroupement, de normalisation et de discrétisation. Il peut arriver parfois que les bases de données contiennent à ce niveau un certain nombre de données incomplètes et/ou bruitées. Ces données erronées, manquantes ou inconsistantes doivent être retravaillées si cela n'a pas été fait précédemment. Cette étape permet de sélectionner et transformer des données de manière à les rendre exploitables par un outil de data mining. Cette étape du processus d'ECD permet d'affiner les données. Parmi ces techniques :

- ✓ **Lissage de données :**

Réduire le bruit dans les données**,** utilisation de techniques de régression.

- ✓ **Normalisation des données** :

La normalisation des données restreint les valeurs numériques à une plage spécifiée. Pour ne pas privilégier les attributs ayant les plus grands domaines de variation (salaire/âge). Les méthodes de normalisation les plus courantes sont les suivantes :

- **Normalisation min-max**: Adapter linéairement les données à une plage comprise entre une valeur minimale et une valeur maximale.

- **Normalisation par Z-score:** Mettre les données à l'échelle en fonction moyenne et de l'écart type.

- ✓ **Agrégation :**

C'est une opération permettant une analyse multidimensionnelle sur les BD volumineuses afin de mettre en évidence une analyse particulière des données, par exemple :
Calculer les niveaux de ventes réalisées de tel produit par mois plutôt que par jour.

- ✓ **Choix de représentations possibles :**

Représentation verticale ou représentation horizontale ou éclatée (plus adaptée à la fouille de données).

1.1.5. Data mining :

Data mining est au cœur du processus d'ECD. Il s'agit à ce niveau de trouver des pépites de connaissances à partir des données. Tout le travail consiste à appliquer des méthodes intelligentes dans le but d'extraire cette connaissance que nous allons détailler dans la suite de ce chapitre.

1.1.6. Evaluation des résultats :

Cette phase, dite d'évaluation ou de validation, a pour objectif de mesurer l'intérêt des modèles extraits, ces derniers doivent être compréhensibles. Deux approches sont communément utilisées dans la validation : une fondée sur des mesures statistiques (méthodes de base de statistique descriptive), et une deuxième par expertise (le diagnostic médical, par exemple) ou l'expert du domaine se charge de la validation du modèle trouvé ou généré.

1.1.7. Intégration de connaissance :

A l'issue des opérations du processus ECD on aboutira à la connaissance qui va se convertir en décision puis en action.

2. Le data Mining :

Après avoir expliqué et définit le processus d'extraction de connaissance à partir de données ECD et ces étapes, nous allons définir et présenter les différentes tâches et techniques du data mining.

2.1. Définition :

Le data mining est l'art d'extraire des connaissances utiles à partir d'une masse de données ou entrepôts de données , Ce processus comprend différents types de services tels que l'exploration de texte, l'exploration de sites Web, l'exploration audio et vidéo, l'exploration de données picturales et l'exploration de réseaux sociaux. Elle se propose d'utiliser un ensemble d'algorithme issus de disciplines scientifiques diverses telles que les statistiques et l'intelligence artificielle pour analyser des bases de données informatiques (souvent grandes) de façon automatique ou semi-automatique, en vue de détecter dans ces données des règles, des associations, des tendances inconnues ou cachées.

En bref, Le data mining est un ensemble de méthodes et de techniques d'analyse et d'extraction de connaissances à partir de données en vue d'aide à la prise de décision.

2.2. Les Domaines d'application du Data Mining :

Le Data mining est une approche d'analyse de données, adaptées et utilisées dans un large nombre de domaine d'activités tels que :

- **Le domaine de la santé :**

L'exploration de données dans le domaine de la santé présente de nombreux intérêts : Les données récoltées à long terme sur de larges populations, permettent d'identifier des facteurs de risque pour certaines maladies comme le cancer et le diabète et mettre en place des programmes à destination des populations à risque. Elle permet en outre le développement de systèmes d'aide au diagnostic médicale.

- **L'analyse du panier de consommation :**

Grâce au data mining, vous pourrez enfin savoir, de façon automatique, quels sont les articles que vos clients ont tendance à acheter ensemble. Vous optimiserez vos processus d'approvisionnement et de stockage. C'est donc l'idéal pour faire des prévisions de vente plus réalistes.

- **Dans l'éducation :**

L'exploration de données générées par les environnements éducatifs aide à reconnaître le comportement d'apprentissage futur de l'élève et prendre des décisions précises pour prédire les résultats de l'étudiant. Avec ces résultats, l'institution peut se concentrer sur ce qu'il faut enseigner et comment enseigner.

- **Dans les CRM :**

La gestion de la relation client (**CRM** en anglais) [5]consiste à obtenir et à conserver des clients, améliorer leur fidélité et mettre en œuvre des stratégies orientées client pour obtenir une relation décente avec lui, pour se faire une organisation commerciale doit collecter des données et les analyser avec les différentes technologies d'exploration de données.

- **La détection de fraude :**

Des milliards de dollars sont perdus à cause de fraudes. Les méthodes traditionnelles de détection de fraude sont un peu longues et sophistiquées. L'exploration de données fournit des modèles significatifs et transforme les données en informations. Un système de détection de fraude idéal devrait protéger les données de tous les utilisateurs. Les méthodes supervisées, consistent en une collection d'échantillons d'enregistrements qui sont classés comme frauduleux ou non frauduleux. Un modèle est construit à partir de ces données, et la technique est faite pour identifier si le document est frauduleux ou non.

- **Web mining :**

Découvrir les préférences des utilisateurs, améliorer l'organisation du site Web, l'analyse de tous les types d'informations sur les logs, adaptation de l'interface utilisateur/service.

Aucun domaine d'application n'est a priori exclue car dès que nous sommes en présence de données empiriques, le data mining peut rendre de nombreux services.

[5]**CRM** : Customer Relationship management.

2.3. Tâches du data mining :

Le data mining repose sur des algorithmes et outils dédiées à différentes tâches. Ses technique sont partagées principalement, entre la classification automatique (supervisée et non supervisée) et la recherche d'associations.

2.3.1. La classification automatique supervisée:

C'est la tâche qui consiste à discriminer des données, l'objectif est de définir des règles permettant de classer des objets en sous-ensembles d'objets appelés classes qui sont connues à priori (de façon supervisée). La classification est un processus à deux étapes : une étape d'apprentissage (entraînement) et une étape de classification (utilisation).

- ✓ **L'étape d'apprentissage :** Il s'agit d'apprendre une règle de classification à partir de données annotées (étiquetées) par l'expert et donc pour lesquelles les classes sont connues.
- ✓ **L'étape de classification :** Le modèle construit dans la première étape est utilisé pour classer de nouvelles données.

La classification regroupe une multitude de techniques à savoir les arbres de décision, les réseaux de neurones, la classification bayésienne, les machines à vecteur de support…etc.

La classification est l'objet de notre travail, nous détaillions les différentes techniques au second chapitre.

2.3.2. La classification automatique non supervisée (Le clustering) :

Le clustering est un type d'apprentissage non supervisé dans lequel les points de données sont regroupés en différents groupes homogènes appelés clusters. En terme simple le clustering prend en entrée un ensemble de données et une mesure de similarité entre ces données, et produit en **s**ortie un ensemble de clusters. En tachant à minimiser la **distance intra-cluster**[6] et à maximiser la **distance inter-clusters**[7].

[6]**Distance inter-cluster :** C'est la distance entre deux clusters.

[7]**Distance intra-cluster :** C'est la distance entre les éléments d'un cluster.

Le clustering peut être reparti en deux types :

- ✓ **Hard clustering** : chaque objet appartient ou non à un cluster
- ✓ **Soft clustering**: chaque objet appartient à chaque cluster à un certain degré (par exemple, une probabilité d'appartenance au cluster).

2.3.3. Les règles d'associations :

Les règles d'association (RA) initialement étudiées en analyse de données en 1986**[3]**, ont été introduites par Agrawal en 1993 en data mining, afin de découvrir des relations significatives entre objets appartenant à une grande quantité de données. Les relations découvertes peuvent être représentées sous forme de règles d'association comme suit :

{Lait} ⟹ {pain}

Cette règle suggère qu'il existe une relation entre la vente du pain et du lait parce que de nombreux clients qui achètent du pain achètent aussi du lait.

Il existe différents algorithmes de générations des règles d'associations à savoir :

Apriori[8]**[4]**,FP-growth**[5]**…etc

2.4. Techniques du data mining :

Les techniques de data mining diffèrent en fonction des besoins de l'utilisateur, entre autres la tâche à effectuer. Chacune des tâches citées ci-dessus regroupe une multitude d'algorithmes pour construire le modèle auquel elle est associée :

[8] **ANNEXE A**

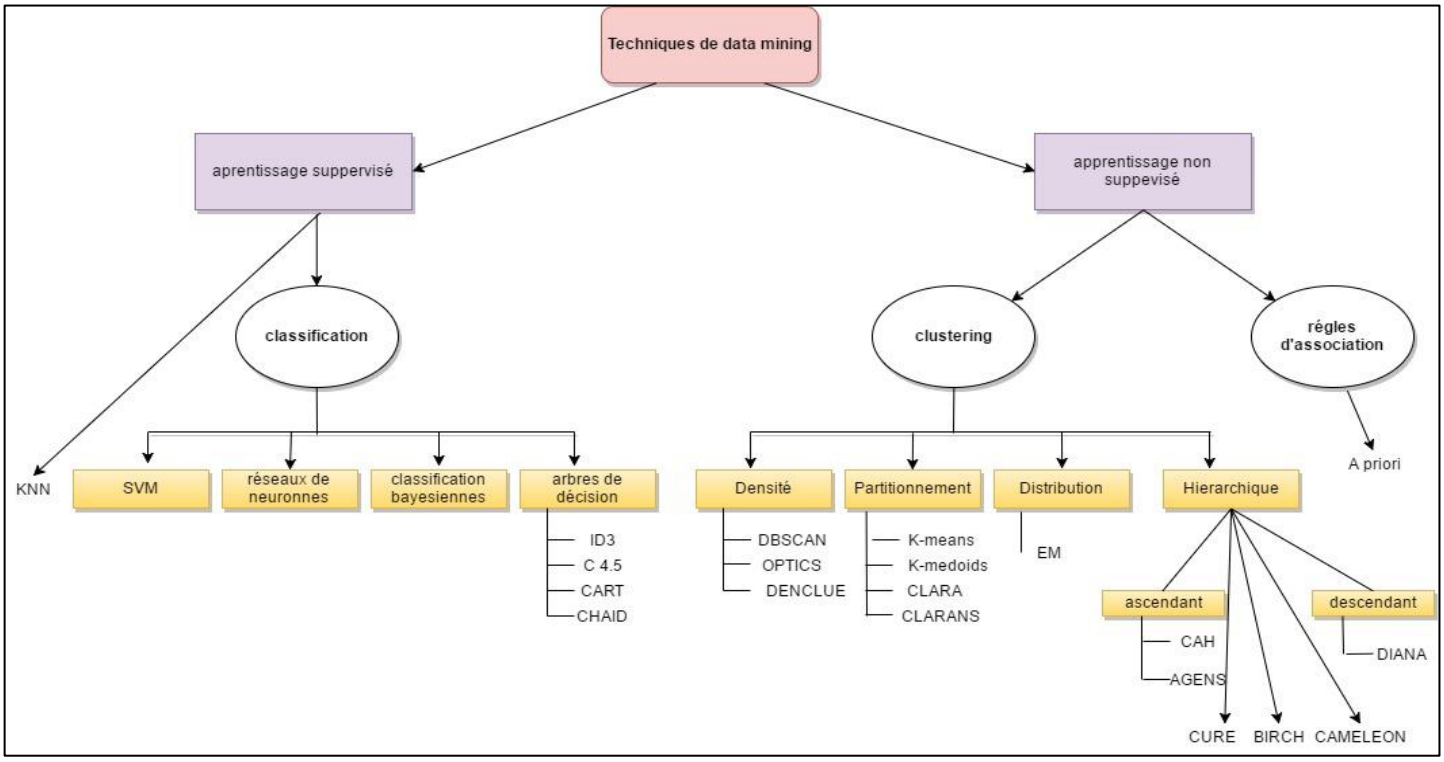

Figure 2:Techniques du data mining[6]

2.4.1. Les mesures de distances :

- **La distance de Minkowski :**

$$\mathbf{d(i,j)} = \sqrt{(|x_{i1} - x_{j1}|^q + |x_{i2} - x_{j2}|^q + .. + |x_{ip} - x_{jp}|^q)} \quad \textbf{(1.1)}$$

Où $\boldsymbol{i} = (x_{i1}, x_{i2}, x_{jp} \ldots)$ et $\boldsymbol{j} = (x_{j1}, x_{j2}, x_{jp} \ldots,)$ sont deux objets p-dimensionnels et **q** un entier positif.

- **La distance de Manhattan :**

Si **q = 1** alors la distance **d** est appelée la distance de Manhattan, c'est la somme des valeurs absolues des différences entre les coordonnées de deux points :

$$\mathbf{d(i,j)} = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + .. + |x_{ip} - x_{jp}| \quad \textbf{(1.2)}$$

- **La distance euclidienne :**

Si q = 2 alors la distance est dite euclidienne, elle calcule la racine carrée de la somme des différences carrées entre les coordonnées de deux points :

$$\mathbf{d(i,j)} = \sqrt{(|\mathbf{x_{i1}} - \mathbf{x_{j1}}|^2 + |\mathbf{x_{i2}} - \mathbf{x_{j2}}|^2 + .. + |\mathbf{x_{ip}} - \mathbf{x_{jp}}|^2)} \quad (1.3)$$

- **La distance de jaccard :**

$$\mathbf{d(i,j)} = \frac{\mathbf{r+s}}{\mathbf{r+s+t}} \quad (1.4)$$

r : nombre d'attributs valant 0 dans **X** et 1 dans **Y**.

s : nombre d'attributs valant 1 dans **X** et 0 dans **Y**.

t : nombre d'attributs valant 1 dans les deux.

2.4.2. Techniques de classification non supervisée (Clustering) :

Il existe plusieurs approches de regroupement ci-dessous quatres approches courantes :

2.4.2.1. Le clustering hiérarchique :

Dans un clustering hiérarchique, un cluster peut être divisé en sous clusters, l'ensemble des clusters étant représenté par un arbre. Un objet appartient à une et une seule feuille dans la hiérarchie, mais également à son nœud père, et ainsi de suite jusqu'à la racine. Les méthodes de clustering hiérarchique permettent d'obtenir ce type de résultats montré par la figure ci-dessous. Il existe deux types d'approches de clustering hiérarchique :

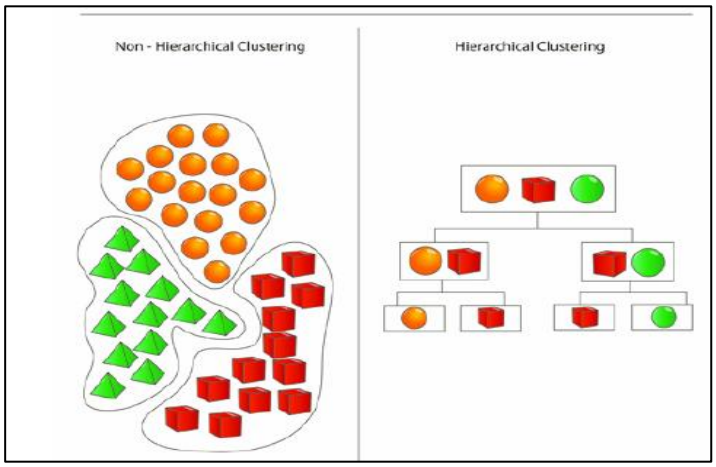

Figure 3:Clustering hiérarchique et non hiérarchique [7]

2.4.2.1.1. Le clustering hierarchique descendant :

Dans ce type de clustering, tous les éléments commencent dans le même cluster. Les différences entre les éléments du cluster sont examinées et deux ou plusieurs clusters sont créés en fonction de la plus grande différence. Une fois terminé, chaque cluster est à nouveau examiné et le processus est répété. Cela continue jusqu'à ce que chaque élément se trouve dans son propre cluster.

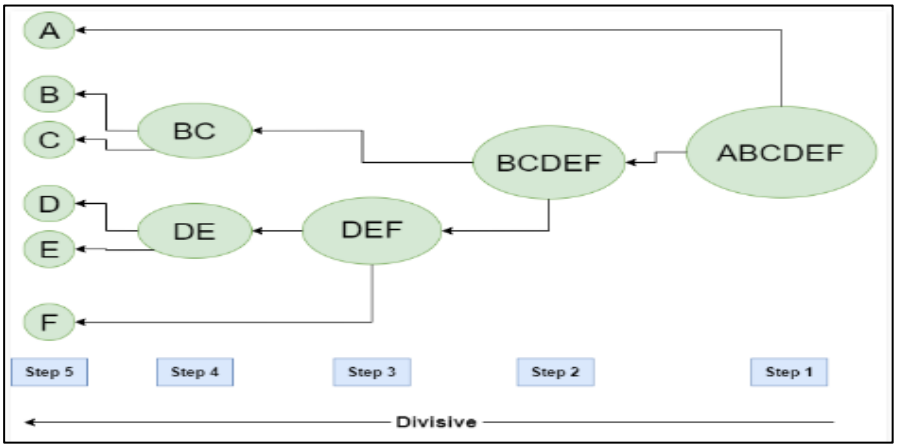

Figure 4 : Regroupement hiérarchique divisif [8]

1) L'algorithme DIANA :

DIANA est également connu sous le nom **DIvisiveANAlysis** est un algorithme de Clustering hiérarchique qui procède par division. Son fonctionnement repose sur un principe simple. Au départ, tous les objets appartiennent à un seul et unique groupe divisif Agglomératif .Un critère de partitionnement est ensuite utilisé pour diviser le groupe en deux sous-groupes jusqu'à la formation de groupes de clusters distincts les uns des autres.

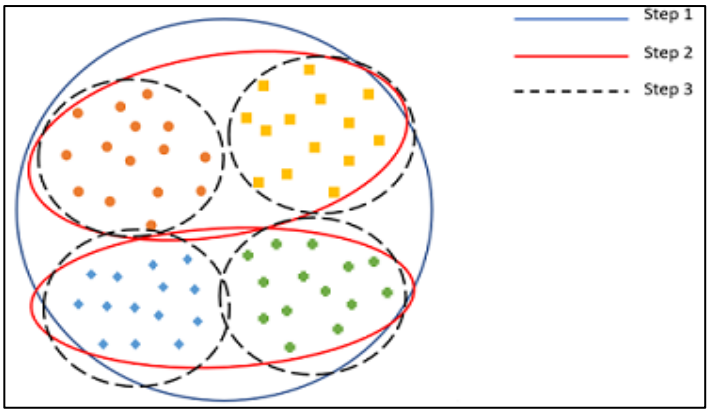

Figure 5 : Représentation fictive de DIANA [9]

2.4.2.1.2. Le clustering hiérarchique ascendant :

Méthode de classification itérative, chaque objet au départ est considéré comme une classe, le regroupement successif de ces classes produit des classes de plus en plus grandes jusqu'à l'obtention d'une classe qui regroupe tous les objets de la population (comme le montre la figure 6).Ce processus itératif produit un arbre binaire appelé **dendrogramme** représentant des partitions, on peut alors choisir une partition en tronquant l'arbre à un niveau donné.

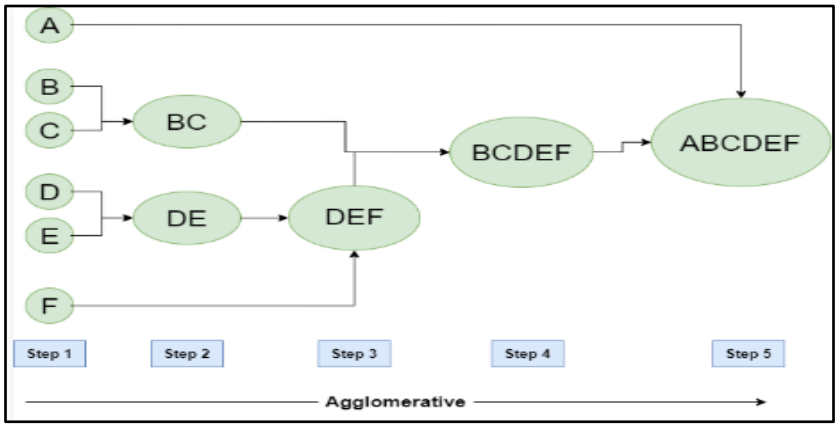

Figure 6 : Regroupement hiérarchique agglomératif [8]

2.4.2.2. Le clustering de partitionnement :

Les méthodes de partitionnement ont comme résultat un ensemble de **k** clusters (partitions) à partir d'une base de données d'objets **«n»** où chaque objet appartenant à un seul cluster et chaque groupe contient au moins un objet. Chaque cluster peut être représenté par un représentant du cluster soit un centroïde (algorithme K-means) ou par l'un des objets du cluster situé près de son centre (algorithmes k-medoïd). Dans ces modèles, le nombre des clusters nécessaires à la fin doivent être mentionné à l'avance, ce qui rend important d'avoir une connaissance préalable de l'ensemble de données.

1) K-means:

L'algorithme du k-means mis au point par MacQueen en 1967 **[10]** souvent appelé algorithme de Lloyd est le plus populaire des algorithmes de clustering de partitionnement. K-Means effectue la division des objets en **K** grappes (cluster) fixé aléatoirement à priori dont chacun d'eux représenté par son centroïde qui est une moyenne (habituellement pondérée) des points situés à l'intérieur du cluster, cette approche ne fonctionne convenablement qu'avec les attributs numériques et le résultat final peut être négativement affecté par la présence de bruits. Chaque point est assigné au cluster dont le centroïde est le plus proche.

L'algorithme k-means est un algorithme facile à implémenter il est applicable à tout type de données mais aussi il présente des limites tel que la sensibilité à la présence de bruit ou encore les résultats finaux qui dépendent fortement des choix des paramètres lors de l'initialisation (**k** , mesure de distance).

- **L'Algorithme de K-means :**

1. Sélectionner K points comme étant un nombre de cluster.
2. Former K clusters en assignant chaque point au centroïde le plus proche.
3. Recalculer le centroïde de de chaque cluster nouvellement forme.
4. Répéter les étapes 2 et 3 jusqu'à ce qu'aucun centroïde ne change.

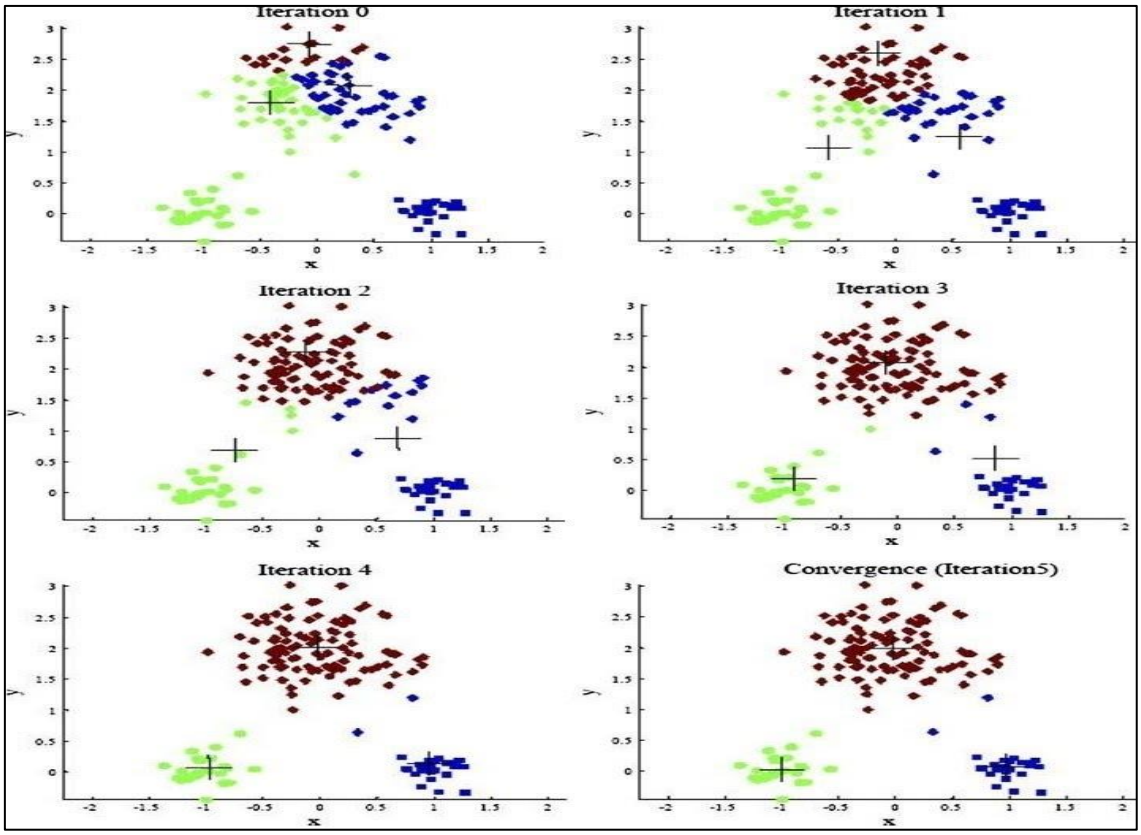

Figure 7:Exemple d'application de l'algorithme K-means avec K=3 [11]

2) k-medïods :

k-medïods **(Partitioning around Medoïds)** a été proposé en 1987**[12]** par Kaufman et Rousseeuw avec une approche similaire à celle de k-means, contrairement à ce dernier k-medïods se base sur des medïods pour créer des clusters. Le terme **médoïd** fait référence à un objet au sein d'un cluster pour lequel la dissimilarité moyenne entre lui et tous les autres membres du cluster est minimale. Il correspond au point le plus central du cluster.

L'algorithme inclus deux étapes, une étape initialisation qui consiste à choisir des points medoids arbitrairement et associer chaque point de l'échantillon à son cluster. Une seconde étape la recherche des meilleurs médoïdes qui se fait par le calcul de **coût swap**[9].

Il est souvent employé avec la distance de Manhatan, le choix des points medoïds et **k** cluster reste aléatoire, alors k-mediods aboutit souvent à des résultats différents cela reste une limite pour cet algorithme.

3) CLARA :

(**C**lustering **L**arge **A**pplications, (Kaufman and Rousseeuw 1990)**[13]** est une extension des méthodes k-medoids (PAM) pour traiter des données contenant un grand nombre d'objets (plusieurs milliers d'observations) afin de réduire le temps de calcul et de stockage .Il constitue la meilleure pratique pour l'exploration de **données spatiales.** [10] Ceci est réalisé en utilisant l'approche d'échantillonnage. Au lieu de trouver des médoïdes pour l'ensemble des données, CLARA considère un petit échantillon des données de taille fixe (**sampsize**) et applique l'algorithme PAM pour générer un ensemble optimal de médoïdes pour l'échantillon. La qualité des médoïdes résultants est mesurée par la moyenne (somme) des dissemblances (distance) entre chaque objet de l'ensemble de données et le médoïde de son cluster, défini comme la fonction de coût (coût swap).

4) CLARANS :

L'algorithme CLARANS (**C**lustering **L**arge **A**pplications **b**ased on **RAN**domized**S**earch) a été mis en place par RaymonNg et Jiawei Han1994 **[14]** dans le contexte de la classification des données spatiales. Poursuit le travail de L'algorithme PAM et CLARA.

CLARANS applique une stratégie de recherche dans un certain graphique **G(n,k)** (graphe de classification de **n** objets en **k** classes),dont chaque noeud est un ensemble de k points (k-medoïdes) appelé un **Current**. Son but est de trouver les meilleurs nœuds qui minimisent le cout de distance en remplaçant un Medoïde par un point non-Medoïde.

[9] **Coût de swap :** la somme des distances de tous les objets à leur médoïde le plus proche.
[10] **Données spatiales :** Désignent les informations liées à des objets ou éléments présents dans un espace géographique.

2.4.2.3. Le clustering basé sur la distribution :

Ces modèles de clustering sont basés sur la notion de la probabilité que tous les points de données du cluster appartiennent à la même distribution (par exemple : Normal, Gaussien).

Un exemple populaire de ces modèles est l'algorithme EM (Esperance Maximisation) qui utilise des distributions normales multi variées.

1) EM (Esperance Maximisation) :

✓ **Définition :**

Proposé par Dempster et al. (1977) **[15]** c'est un algorithme itératif qui permet de trouver les paramètres du maximum de vraisemblance d'un modèle probabiliste lorsque ce dernier dépend de variables latentes non observable .Il utilise le soft clustering.

- **Maximum de vraisemblance :** estimateur statistique, recherche les valeurs des paramètres maximisant la fonction de vraisemblance
- **Vraisemblance** : Mesure la probabilité que des observations appartiennent à une loi de probabilité donnée.

2.4.2.4. Le clustering basé sur la densité :

Ces modèles recherchent dans l'espace de données des zones de densité variée de points de données. Il isole des régions de densité différentes et attribue les points de données au sein de ces régions dans le même cluster**. DBSCAN [16] et OPTICS [17]** sont des exemples populaires de modèles de densité.

1) DBSCAN:

DBSCAN **(D**ensity-**B**ased **S**patial **Cl**ustering **O**f **A**pplications **W**ith **N**oise**)** c'est un algorithme de partitionnement de données basé sur la notion de densité proposé en 1996**[16]** par Martin Ester, Hans-PeterKriegel, Jörg Sander et Xiaowei Xu.DBSCAN utilise deux paramètres :

- ✓ **Eps (Ɛepsilon):** Rayon maximal de voisinage d'un point.

- ✓ **Minpts :** Nombre minimum de points devant se trouver dans un rayon **Ɛ** pour qu'ils forment un cluster.

➢ Fonctionnement de l'algorithme :

DBSCAN itère sur les points du jeu de données. Pour chacun des points qu'il analyse, il construit l'ensemble des points atteignables par densité depuis ce point il calcule l'epsilon-voisinage de ce point, puis si ce voisinage contient plus de minPts, les **epsilons-voisinages** [11]de chacun d'eux, et ainsi de suite, jusqu'à ne plus pouvoir agrandir de cluster, ainsi marquer le point comme étant visité.

Si le point considéré est inférieur à minPts, il sera alors étiqueté comme du bruit. Le processus terminera lorsque tous les points sont marqués au tant que visités.

Cela permet à DBSCAN d'être robuste aux données aberrantes puisque ce mécanisme les isole.

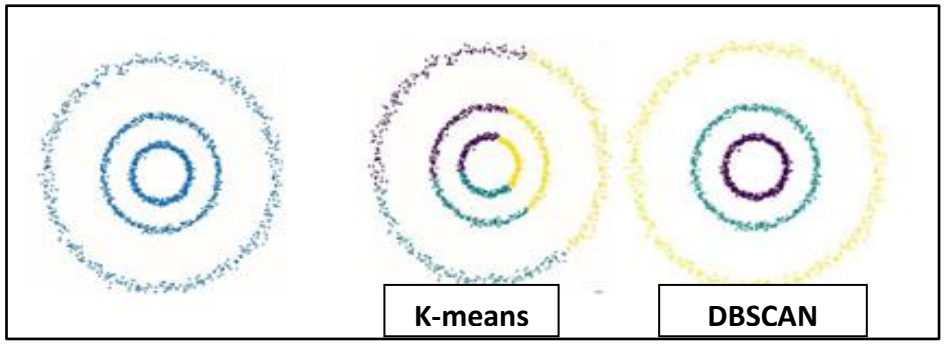

Figure 8: Illustration des résultats donnés par K-means et celle donnes par DBSCAN[18]

- [11] **Ɛ -voisinages d'un point x :** Noté **N Ɛ (x)** est l'ensemble de points **u** du jeu de données **X** qui sont à une distance de **x** inférieure ou égale à **Ɛ** tels que **: « N Ɛ (x)={ u ϵ X / d(u,x) ≤ Ɛ} »**

Conclusion :

Nous avons pu voir à travers ce chapitre, que les données brutes présentes dans des entrepôts de donnée peuvent être exploitées de manière à en extraire des connaissances ces dernières sont obtenues en passant par plusieurs phases à savoir l'acquisition des données, leur préparation, leur traitement pour en extraire des règles et enfin la validation des connaissances extraites. Nous nous sommes intéressés dans notre travail à la phase du data mining qui le cœur de ce processus qui se repose sur une diversité de techniques supervisée et non supervisée qui se distinguent chacune d'elles par son principe de fonctionnement, son algorithme et son résultat.

Chapitre2:

Les techniques de la classification supervisée

Introduction :

Le machine Learning ou apprentissage automatique en français est l'une des tendances technologiques depuis quelques années. Les entreprises prennent conscience de l'importance de leurs données et l'adoption de cette méthode devient donc incontournable pour traiter ces jeux de données et en tirer profit.

Dans le premier chapitre nous avons évoqué les deux approches de l'apprentissage automatique à savoir le supervisé et le non supervisé, nous allons consacrer celui-ci à l'apprentissage automatique supervisé aussi appelé classification automatique supervisée, et à ses différentes techniques, comment fonctionnent-elles ?

Nous verrons dans un premier temps les algorithmes d'arbres de décision et décrivant le comportement de chacun d'eux à savoir ID3, CART et C4.5, nous enchainons avec la méthode des k plus proches voisins et les réseaux de neurones, ceux-là feront l'objet de notre étude dans le troisième chapitre, nous clôturons avec la classification bayésienne et les machines à vecteurs de support.

1. Techniques de classification supervisées :

1.1 Arbres de décisions :

1.1.1 Définition :

Les AD sont un outil d'aide à la décision. C'est une représentation hiérarchique des données sous forme des séquences de décisions (tests) en vue de la prédiction d'un résultat ou d'une classe. Chaque individu (ou observation), qui doit être attribué(e) à une classe, est décrit(e) par un ensemble de variables qui sont testées dans les nœuds de l'arbre. Les tests s'effectuent dans les nœuds internes et les décisions sont prises dans les nœuds feuilles.

Les arbres de décisions sont une représentation calculable automatiquement par des algorithmes d'apprentissage supervisé. Un arbre de décision peut être représenté par une série de règles « si-alors » communément appelées règles de décisions.

Nous distinguons deux catégories d'arbres de décision, selon la nature de la classe à prédire :

- **Arbre de classification :** Pour expliquer et/ou prédire l'appartenance d'objets (observations, individus) à une classe (ou modalité ou catégorie) d'une **variable qualitative**[12], sur la base de **variables quantitatives** [13]et/ou qualitatives.
- **Arbre de régression :** Pour expliquer et/ou prédire les valeurs prises par une variable dépendante quantitative, en fonction de variables explicatives quantitatives et/ou qualitatives.

1.1.2 Structure d'un arbre de décision :

Un arbre de décision peut être représenté par une suite de règles de décisions « si-alors » ou bien graphiquement par un arbre binaire ou n-aire.

La **figure 1** montre la représentation graphique d'un arbre de décision :

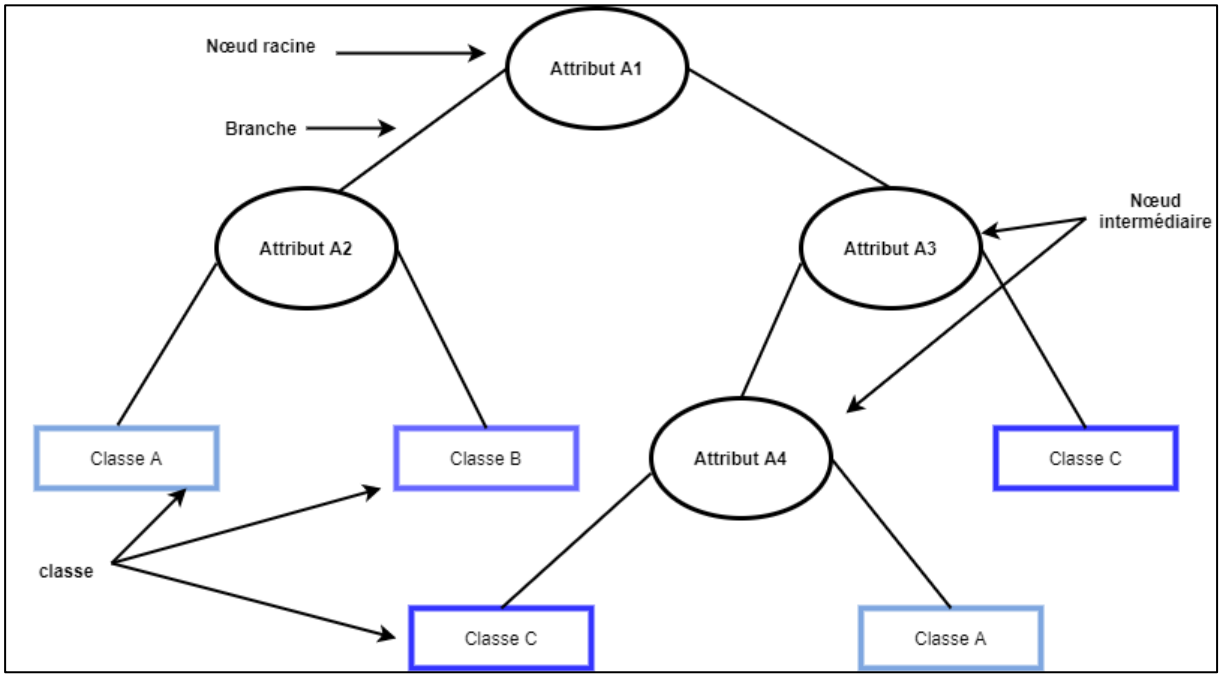

Figure 1: Structure d'un arbre de décision

[12] **Variable qualitative** : Expriment une qualité, un état comme le sexe et la couleur

[13] **Variable quantitative** : une variable quantitative est une variable qui reflète une notion de grandeur, c'est-à-dire si les valeurs qu'elle peut prendre sont des nombres comme la taille, l'âge.

Dans la figure **2** :

- Chaque nœud intermédiaire teste un attribut.
- Chaque branche correspond à une valeur d'attribut.
- Chaque feuille est étiquetée par une classe (Comme l'explique la figure suivante).

Exemple :

La figure **10** est un exemple d'un arbre de décision, généré à partir d'un ensemble de données qui a comme attributs : **"douleur", "fièvre" et "toux"** et comme classe à prédire **"Maladie"** qui peut prendre les valeurs : **"Appendicite", "Rhume", "Refroidissement", "Mal de gorge", "Rien".**

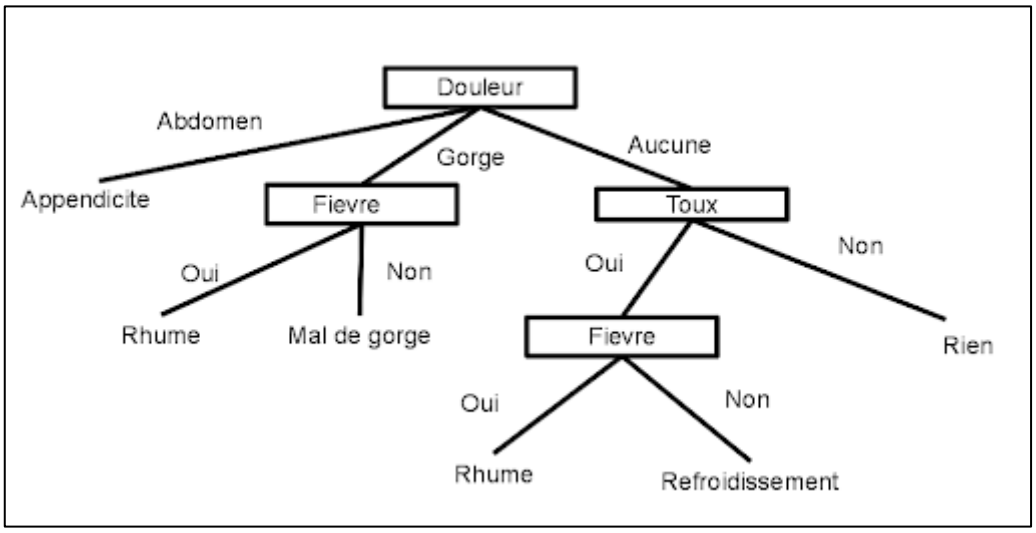

Figure2 : exemple d'arbre de décision [19]

Dans l'exemple de la figure **2**, pour classifier un nouvel objet, on suit le chemin partant de la racine (nœud initial) à une feuille en effectuant les différents tests d'attributs à chaque nœud. Les règles de décision qui lui sont associées sont les suivantes :

Si (Douleur =Abdomen) **ALORS** Maladie = Appendicite.
Si (Douleur = Gorge) **ET** (Fièvre=Oui) **ALORS** Maladie= Rhume.
Si (Douleur = Gorge) **ET** (Fièvre=Non) **ALORS** Maladie= Mal de gorge.
Si (Douleur = Aucune) **ET** (Toux=Non) **ALORS** Maladie= Rien
Si (Douleur = Aucune) **ET** (Toux=Oui) **ET** (Fièvre=Oui) **ALORS** Maladie= Rhume
Si (Douleur = Aucune) **ET** (Toux=Oui) **ET** (Fièvre=Non) **ALORS** Maladie= Refroidissement.

1.1.3 Algorithme générique d'arbres de décision :

Algorithme générique

Arbre ⟵ arbre vide, nœud_courant ⟵ racine

Répéter

Décider si le nœud courant est terminal

Si le nœud est terminal alors lui affecter une classe

Sinon sélectionner un test et créer autant de nœuds fils qu'il y a de réponses au test

Passer au nœud suivant (s'il existe)

Jusqu'à obtenir un arbre de décision consistant

Pour sélectionner un test, une heuristique appelée critère de segmentation est calculée, ce critère est spécifique à chaque algorithme, comme : Gain d'information, Entropie, Indice de Gini… etc.

1.1.4 Algorithmes d'arbres de décisions :

Il existe différents algorithmes de construction d'arbre de décision parmi eux : CHAID, ID3, C4.5 et CART, ils seront vu en détails donc ce qui suit :

1.1.4.1 Algorithme CHAID :

CHAID (**Ch**ai Squared **A**utomatic **I**nteraction **D**etector) est une technique de classification qui est bien adaptée à l'analyse de grosses volumétries de données développé vers 1980**[20]** par Kass. Il génère un arbre n-air, il traite les données qualitatives, données manquantes et les données continues. L'algorithme CHAID utilise le test du **khi-Deux(2,1)** pour déterminer la meilleure division et pour effectuer ce calcul on construit un tableau de fréquence **(tableau1)**pour chaque attribut, par classe, selon les valeurs que cet attribut peut prendre. Par exemple, si on a **K** classes et qu'un attribut peut prendre **L** valeurs, le tableau est le suivant :

Y / X	x_1	x_l	x_L	Σ
y_1				
		$\vdots$		
y_k	$\cdots$	n_{kl}	$\cdots$	$n_{.l}$
		$\vdots$		
y_K				
Σ		$n_{k.}$		n

Tableau 1: tableau de fréquences

$$\chi^2 = \sum_{k=1}^{K}\sum_{l=1}^{L} \frac{\left(n_{kl} - \frac{n_{k.} \times n_{.l}}{n}\right)^2}{\frac{n_{k.} \times n_{.l}}{n}} \qquad \textbf{(2,1)}$$

Où :

C_k : ensemble des k classes du jeu de données.

$X_{i,k}$: la valeur i de la classe **C_k**.

n_{kl} : représente le nombre d'éléments appartenant à la classe **C_k** pour la valeur **$X_{i,k}$**

n_k : représente la sommation du nombre d'éléments de toutes les classes pour une certaine valeur de l'attribut.

n_l : représente la sommation du nombre d'éléments des valeurs possibles pour l'attribut calculé pour une certaine classe.

n: représente le nombre d'exemples dans le jeu d'apprentissage.

1.1.4.1.1 Pseudo code de l'algorithme CHAID :

CHAID(A,C,S)

Entrée :

S : données d'apprentissage, **A** : ensemble d'attributs, **C** : ensemble de classes.

Sortie : arbre de décision.

Début

Si tous les enregistrements de **S** ont la même valeur pour l'attribut classe **C alors**

Retourner un seul nœud feuille avec cette valeur.

Fin si

Si A est vide **alors**

Retourner un simple nœud avec comme valeur la valeur de la classe la plus trouvées dans **S**.

Fin si

Pour tout attribut $\mathbf{A_j}$**, j :1-n faire** :

Calculer la matrice de calcul de fréquence.

Calculer le test de **khi-deux** $\mathbf{X^2}$

Fin pour

Soit Noeud l'attribut ayant le plus grand $\mathbf{X^2}$ des attributs de **A**.

Retourner un arbre avec le nœud racine **Noeud** et les arcs sont étiquetés par $\mathbf{V_1, V_2,\ldots, V_m}$ (les valeurs de l'attribut **Noeud**).

$\mathbf{Sv_i}$ **i:1-m** : les sous-ensembles de **S** constitué respectivement des enregistrements de valeur $\mathbf{V_i}$ **i:1-m** de l'attribut $\mathbf{A_j}$ **j:1-n**.

Aller faire CHAID (A-{Noeud}, C, $\mathbf{SV_1}$**) . . . . CHAID(A-{Noeud}, C,** $\mathbf{Sv_m}$**)**

Fin

1.1.4.1.2 Exemple d'application de l'algorithme CHAID :

Le tableau **1** représente un exemple d'un ensemble de données de cinq observations et à quatre attributs qui nous informent sur quel **ciel** fait-il ? **la température, l'humidité, le vent**. Et décider si c'est une bonne journée pour **jouer** au tennis ou pas ?

Ciel	Température	Humidité	Vent	Jouer
Ensoleillé	Chaud	Elevé	Faible	**Non**
Ensoleillé	Chaud	Elevé	Fort	**Non**
Nuageux	Chaud	Elevé	Faible	**Oui**
Pluvieux	Frais	Normal	Faible	**Oui**
Pluvieux	Frais	Normal	Fort	**Non**

Tableau 2:Exemple d'un jeu de données

En premier lieu nous allons calculer les tables de calcule des fréquences pour les quatre attributs dans notre exemple l=4 et K=5 :

Table de calcule de fréquence pour l'attribut ciel :

	Ensoleillé	**Nuageux**	**Pluvieux**	$\mathbf{n_{k.}}$
Oui	0	1	1	**2**
Non	2	0	1	**3**
$\mathbf{n_{.l}}$	**2**	**1**	**2**	**n**=5

Tableau 3:table de calcul de fréquence de l'attribut ciel

Le tableau 2 peut être expliqué comme suit :

- Il existe 0 enregistrement dans le jeu de données pour Ciel=Ensoleillé et Jouer=Oui
- Il existe 2 enregistrements dans le jeu de données pour Ciel=Ensoleillé et Jouer=Non
- Il existe 1 enregistrement dans le jeu de données pour Ciel=Nuageux et Jouer=Oui
- Il existe 0 enregistrement dans le jeu de données pour Ciel=Nuageux et Jouer=Non
- Il existe 1 enregistrement dans le jeu de données pour Ciel=Pluvieux et Jouer=Oui
- Il existe 1 enregistrement dans le jeu de données pour Ciel=Pluvieux et Jouer=Non

Nous allons faire la même table pour les trois attributs restants, ensuite calculer le test de khi-deux **(2,1)** pour chacun d'eux :

Calculer le test de khi-deux pour l'attribut ciel :

$$\mathbf{X^2} = \frac{(0-\frac{2*2}{5})^2}{\frac{2*2}{5}} + \frac{(1-\frac{2*1}{5})^2}{\frac{2*1}{5}} + \frac{(0-\frac{2*2}{5})^2}{\frac{2*2}{5}} + \frac{(1-\frac{2*2}{5})^2}{\frac{2*2}{5}} + \frac{(2-\frac{3*2}{5})^2}{\frac{3*2}{5}} + \frac{(0-\frac{3*1}{5})^2}{\frac{3*1}{5}} + \frac{(1-\frac{3*2}{5})^2}{\frac{3*2}{5}} = 4{,}18$$

Attribut	Ciel	Température	Humidité	Vent
Khi-deux	4,18	0 ,66	0,14	2,93

Tableau 4:Test de khi-deux pour les quatre attributs du jeu de données

- L'attribut racine est l'attribut avec la valeur de khi-deux la **plus élevée**, donc **ciel** sera l'attribut mis dans **la racine** de l'arbre.
- Créer autant de branches autant qu'il y a de valeurs pour l'attribut **ciel**.
- Nous remarquons que pour **ciel=Ensoleillé** tous les enregistrements prennent la valeur **jouer=Non**, donc la branche sera étiquetée par **Oui** (voir **figure 3**)
- Nous remarquons que pour **ciel=Nuageux** tous les enregistrements prennent la valeur **jouer=Non**, donc la branche sera étiquetée par **Non** (voir **figure 3**)

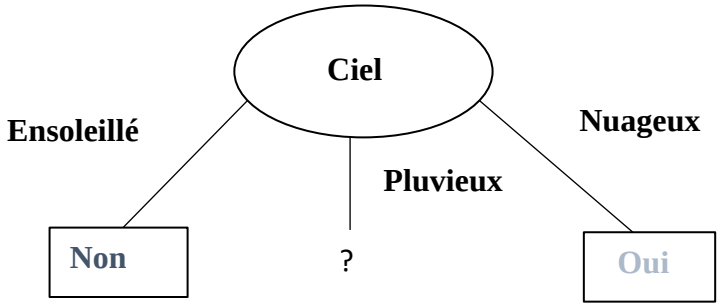

Figure3:Arbre de décision après la première itération

- Pour la branche **ciel=pluvieux** nous referons le processus avec la base d'apprentissage suivante :

Ciel	Température	Humidité	Vent	Jouer
Pluvieux	Frais	Normal	Faible	Oui
Pluvieux	Frais	Normal	Fort	Non

Tableau 5:Ensemble d'apprentissage pour ciel=pluvieux

- Nous calculons les **tables de calcul de fréquences** pour les attributs humidité, vent et température, ensuite calculer le test de **khi-deux**
- À cet étape l'attribut qui a la plus grande valeur de khi-deux est l'attribut **Vent**,
- Nous créons autant de branches autant qu'il y a de valeurs pour l'attribut **Vent**
- La branche avec la valeur **Vent=Faible** sera étiquetée avec la classe **Oui,** quant à la branche avec la valeur **Vent=Fort** elle sera étiquetée avec la classe **Non.**

La **figure 4** représente l'arbre final que nous obtiendrons :

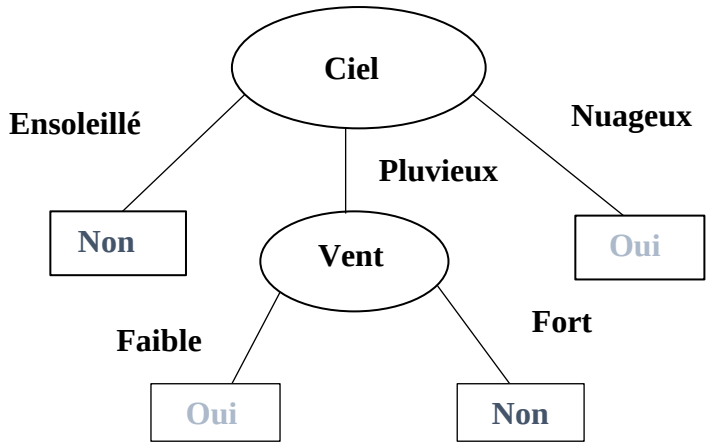

Figure 4 : Arbre de décision final

1.1.4.1.3 . Avantages de l'algorithme CHAID :

- Ne nécessite pas beaucoup de calculs.
- Facile à interpréter.
- Performant sur des grands jeux de données.

1.1.4.1.4 . Inconvénients de l'algorithme CHAID :

- Nécessite un échantillon d'apprentissage de grande taille
- Des arbres de décision avec tendance au sur apprentissage.

1.1.4.2 Algorithme ID3 :

1.1.4.2.1 Définition :

ID3 (**itératif Dichotomiser 3)** développé par Quilan Ross en 1986**[21]**, est un algorithme de classification supervisée, qui permet de construire récursivement un arbre de décision n-aire. Il traite des données qualitatives et quantitatives discrètes mais pas les données manquantes et numériques continues. À chaque étape de la récursion, ID3 calcule l'entropie et les gains d'information de chaque attribut. De cette manière, l'attribut le plus dominant est mis sur l'arbre de décision comme une racine. Par la suite, les deux scores seraient à nouveau calculés pour les autres attributs. Ainsi, l'attribut suivant le plus dominant est trouvé. Enfin, cette procédure se poursuit jusqu'à ce qu'une décision soit prise.

ID3 génère l'arbre de décision en se basant sur les deux mesures : l'entropie de Shannon et le Gain d'information définis ci-après :

- **L'entropie de Shannon E(s) :** L'entropie de **S** est la quantité d'information qu'elle peut apporter. Elle peut être définit comme une mesure d'incertitude régnant dans une collection d'objet. Si tous les objets appartiennent à la même classe, il n'y a pas de désordre Elle est donnée par l'équation suivante :

$$\textbf{Entropie(s)} = \sum_{i=1}^{n} -p\,(ci)\,\log 2\; p\,(ci) \quad (2, 2)$$

Sachant que : ∀ **S 0 ≤ Entropie(S) ≤ 1** Où :

S : l'ensemble de données.

p(ci): proportion de la classe ci (probabilité) dans **S.**

l**og2** : logarithme de base 2.

- **Gain d'information :** fournit la valeur de l'information gagné après la séparation du nœud parent en nœud fils.

Le gain d'informations de **S** par rapport à une partition $\mathbf{S_v}$ donnée, est la variation d'entropie causée par la partition de **S** selon **Sv (**différence d'entropie entre avant et après le partitionnement selon un attribut).

$$\textbf{Gain (S, A) = E(S) - Σ ((| } S_v \textbf{ | / | S |) * Entropie (} S_v\textbf{)) (2,3)}$$

Où :

Sv = Sous-ensemble de **S** pour lequel l'attribut **A** a la valeur **v**.

$|S_v|$ = nombre d'éléments dans S_v.

| S | = nombre d'éléments dans **S**.

1.1.4.2.2 Pseudo code de l'algorithme ID3 :

ID3 (A, C, S)

S : données d'apprentissage, **A** : ensemble d'attributs, **C** : ensemble de classes.

Sortie : arbre de décision.

Début

Si tous les enregistrements de **S** ont la même valeur pour l'attribut classe **C alors**

Retourner un seul nœud feuille avec cette valeur.

Fin si

Si A est vide **alors**

Retourner un simple nœud avec comme valeur la valeur de la classe la plus trouvées dans **S**.

Fin si

Calculer l'entropie **E(S)** de **S**

Pour tout attribut $\mathbf{A_j, j:1\text{-}n}$ **faire** :

Calculer le gain **Gain** ($\mathbf{A_j}$, **S)**

Fin pour

Soit Noeud l'attribut ayant le plus grand **Gain (A, S)** des attributs de **A**.

□□Retourner un arbre avec le nœud racine **Noeud** et les arcs sont étiquetés par $\mathbf{V_1, V_2, \ldots, V_m}$ (les valeurs de l'attribut **Noeud**).

□□$\mathbf{Sv_i\ i:1\text{-}m}$: les sous-ensembles de **S** constitué respectivement des enregistrements de valeur $\mathbf{V_i\ i:1\text{-}m}$ de l'attribut $\mathbf{A_j\ \ j:1\text{-}n}$.

□□**Aller faire ID3 (A-{Noeud}, C,** $\mathbf{SV_1}$**) . . . . ID3(A-{Noeud}, C,** $\mathbf{SV_m}$**)**

Fin

1.1.4.2.3 Exemple d'application de l'algorithme ID3 :

Dans ce qui suit nous allons vous montrer les étapes du déroulement de l'algorithme id3 pour le même ensemble de données que nous avons pris pour l'algorithme CHAID (tableau 1) :

Étape01 :

1) Le calcul de l'entropie de Shanon pour l'ensemble de données :

Noter : Entropie(s)=E(s)

E(s) = - probabilité (oui) $\log_2$ probabilité (oui) - probabilité (non)$\log_2$ probabilité (non)

Où la probabilité **(oui)** et la probabilité de **(non)** signifient respectivement le nombre d'enregistrement qui ont comme valeur de classe **(oui)** et **(non)** divisé par la taille de l'ensemble de données.

E(s) = -2/5 $\log_2$ (2/5) -3/5 $\log_2$ (3/5)=**0.97**

2) Le calcule de gain pour chaque attribut :

a. Pour l'attribut ciel :

Gain (ciel) = **Entropie(s) –($|Ensoleillé|/|S|$ x Entropie($S_{Ensoleillé}$) -$|Nuageux|/|S|$ x Entropie($S_{Nuageux}$) - $|Pluvieux|/|S|$ x Entropie($S_{Pluvieux}$))** tels que **:**

- $|Ensoleillé|$**,** $|Nuageux|$ **,** $|Pluvieux|$ signifient respectivement le nombre de valeur Ensoleillé , Nuageux, pluvieux de l'attribut **ciel** .
- $|S|$ **:** Le nombre d'enregistrement de l'ensemble d'apprentissage**.**
- **S (oui)** : Une partie de l'ensemble d'apprentissage ou la valeur d'attribut (**ciel**) est égale a **« oui »** voir le tableau .
- **S (non)** : Une partie de l'ensemble d'apprentissage ou la valeur d'attribut (**ciel**) est égale a **« non»** voir le tableau

Ciel	Température	Humidité	Vents	Jouer
Ensoleillé	Chaud	Élevé	Faible	**Non**
Ensoleillé	Chaud	Élevé	Fort	**Non**

Tableau 6: $S_{Ensoleillé}$

Ciel	Température	Humidité	Vents	Jouer
Nuageux	Chaud	Élevé	Faible	Oui

Tableau 7: $S_{Nuageux}$

Ciel	Température	Humidité	Vents	Jouer
Pluvieux	Frais	Normal	Faible	Oui
Pluvieux	Frais	Normal	Fort	Non

Tableau 8: $S_{Pluvieux}$

On utilisant les tableaux 1,2 ,3 et la formule d'entropie, nous pourrons calculer l'entropie pour les trois valeurs **Ensoleillé, Nuageux, Pluvieux** de la façon suivante :

Entropie(Ensoleillé) = $-2/2 \log_2 2/2 =$ **0.**

Entropie(Nuageux) = $-1/1 \log_2 1/1 =$ **0.**

Entropie(Pluvieux)= $--1/2 \log_2 1/2 - 1/2 \log_2 1/2 =$ **1.**

D'où :

Le Gain(ciel) = $0.97-(2/5 \times 1-0-0) =$ **0.57.**

De la même façon nous avons calculé le Gain pour les autres attributs nous avons obtenus les résultats suivant :

Attributs	Ciel	Température	Humidité	Vent
Gain	**0.57**	0.03	0.03	0.43

Tableau 9:Résultats du Gain pour tous les attributs

D'après les résultats donnés par le tableau nous remarquons que l'attribut **Ciel** qui a le plus grand gain d'information et donc d'après l'algorithme ID3 il va être en racine de l'arbre de décision comme le montre **la figure 5**

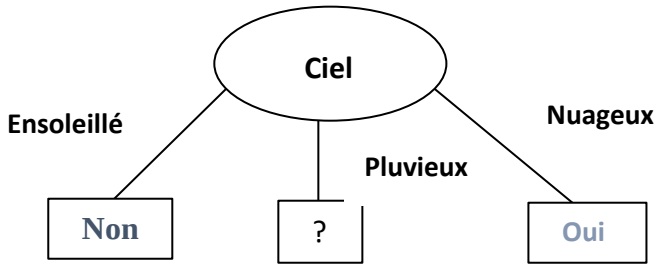

Figure5: Arbre de décision résultant de la première étape

Étape 02 :

Pour continuer la construction de l'arbre dans la **figure5**. Nous avons procédé de la même façon de l'étape 1, nous avons alors obtenu le même arbre final de l'algorithme CHAID(voir la **figure4**)

1.1.4.2.4 Les avantages de l'algorithme ID3 :

- Simple et facile à mettre en œuvre.

1.1.4.2.5 Les inconvenients de l'algorithme ID3 :

- Ne traite pas les valeurs manquantes et continues.
- Les arbres tendent à etre de grande taille, donc probléme d'interprétabilité

1.1.4.3 Algorithme C4.5 :

1.1.4.3.1 Définition :

C4.5 a été proposée en 1993**[22]**, toujours par Ross Quinlan, c'est une amélioration de ID3 venu pour pallier à ces limites en ce qui est du traitement des valeurs manquantes, valeurs continues ainsi l'élagage des arbres de décisions.

Il repose complètement sur ID3, l'algorithme inclus deux phases, la phase de la génération de l'arbre dite d'expansion ainsi que l'élagage de l'arbre.

1.1.4.3.2 La phase d'expansion :

C'est la phase de la génération de l'arbre de décision à la différence de ID3, C4.5 utilise en plus comme mesures de segmentation les fonctions **SplitInfo** et le **Gain Ratio**.

- **SplitInfo :**

Le gain d'information présente un inconvénient puisque cette dernière favorise les attributs **à un grand** nombre de valeurs. Cette notion de **Split Info** a été introduite pour remédier ce problème.

$$\mathbf{SplitInfo(S,A)} = -\sum_{i=1}^{n} \frac{|Si|}{|S|} * \log 2 \left(\frac{|Si|}{|S|}\right) \qquad \textbf{(2,4)}$$

Où :

S : Ensemble d'apprentissage.

A : Valeur d'un attribut.

n : Le nombre de valeur de l'attribut **A**

S_i : Un sous-ensemble de **S** pour lequel l'attribut **A** a la valeur **i**.

- **Gain Ratio :** C'est le rapport entre gain et SplitInfo.

$$\textbf{Gain Ratio} = \frac{\textbf{Gain}(S, A)}{\textbf{SplitInfo}(S, A)} \qquad (2,5)$$

Où :

S : Ensemble ou échantillon d'attributs.

A : Valeur d'un attribut.

a. Pseudo code de l'algorithme C4.5 :

C4.5(A, C, S)

Entrée :

S : données d'apprentissage, **A** : ensemble d'attributs, **C** : ensemble de classes.

Sortie : arbre de décision utilisant l'algorithmeID3.

Début

Si tous les enregistrements de **S** ont la même valeur pour l'attribut classe **C alors**

Retourner un seul nœud feuille avec cette valeur.

Fin si

Si A est vide **alors**

Retourner un simple nœud avec comme valeur la valeur de la classe la plus trouvées dans **S**.

Fin si

Calculer l'entropie **E(S)** de **S**

Pour tout attribut $\mathbf{A_j}$**, j :1-n faire** :

Calculer le gain **Gain Ratio (**$\mathbf{A_j}$**)**

Fin pour

Soit Noeud l'attribut ayant le plus grand **Gain Ratio** des attributs de **A**.

□□Retourner un arbre avec le nœud racine **Noeud** et les arcs sont étiquetés par $\mathbf{V_1, V_2, \ldots, V_m}$ (les valeurs de l'attribut **Noeud**).

□□$\mathbf{Sv_i}$ **i:1-m** : les sous-ensembles de **S** constitué respectivement des enregistrements de valeur $\mathbf{V_i}$ **i:1-m** de l'attribut $\mathbf{A_j}$ **j:1-n**.

□□**Aller faire C4.5 (A-{Noeud}, C,** $\mathbf{SV_1}$**) . . . . C4.5(A-{Noeud}, C,** $\mathbf{Sv_m}$**)**

Fin.

---

b. Exemple d'application de l'algorithme C4.5 :

Soit l'ensemble d'apprentissage du **tableau 2.**

La génération de l'arbre de décision pour cet algorithme a été faite comme suit :

1) **le calcul de l'entropie et le gain d'information** : (voir l'exemple d'application de l'algorithme ID3)
2) **Le calcul du splitInfo pour chaque attribut :**

- **Pour l'attribut Ciel :**

SplitInfo (ciel)= - |S Ensoleillé| / |S|x log $_2$ |S Ensoleillé| / |S| -|S Nuageux| / |S|x log $_2$ |S Nuageux| / |S|- |S Pluvieux| / |S|x log $_2$ |S Pluvieux| / |S|.

SplitInfo (ciel)= -2/5 $\log_2$ 2/5 -1/5 $\log_2$ 1/5 -2/5 log2 2/5 = **1.52**.

De la même manière le **splitInfo** des autres attributs sont calculé, les résultats trouvés sont les suivants :

Split Info (Température)=**0.97,** SpliInfo (Humidité)=**0.97,** SplitInfo (Vent)= **0.97**

3) **Le calcul du Gain ratio pour chaque attribut :**

Gain Ratio (ciel) = Gain (ciel)/ splitInfo(ciel)

= 0.57/1.52= **0,375**

GainRatio(Température) = GainRatio(Humidité)=**0.291,** GainRatio(Vent)=**0.04.**

Nous remarquons d'après les résultats du **Gain Ratio** que l'attribut **ciel** qui a eu la plus grande valeur est donc nous déduisant que cet attribut va être la racine de l'arbre de décision, nous continuerons la construction de l'arbre pour une deuxième itération et à la fin de cette dernière que nous finirons la construction de notre arbre de décision.

▪ **Remarque :** L'arbre résultant de la première et la deuxième itération sont les mêmes que nous avons obtenus dans la partie exemple d'application d'arbre de décision des algorithmes CHAID et ID3 (voir la **figure3** et **la figure4**).

1.1.4.3.3 La phase d'élagage :

Consiste à réduire la taille des arbres de décision complexes, retrancher ce qui est inutile et marginal tout en supprimant les nœuds les moins significatifs de l'arbre et les remplacer par une feuille. Il existe deux types d'élagage :

- **Le pré-élagage** : Fixer un critère d'arrêt qui permet de stopper la construction de l'arbre lors de la phase de construction, se fait pendant la génération de l'arbre. Pour un arbre C4.5 le pré-élagage se fait soit :
  - ✓ Choisir un seuil pour le nombre d'itération de la récursivité de l'arbre.
  - ✓ Arrêter quand la recréation d'un nœud n'a pas de valeur statistique.
- **Le post-élagage :** Se fait après la génération de l'arbre. Il consiste à supprimer les sous-arbres qui n'améliorent pas l'erreur de classification. Ci-dessous le pseudo-code pour le réaliser.

Élagage C4.5

Entrée :
nœud = l'arbre (ou sous-arbre) généré par l'algorithme **C4.5.**
exemples = set d'exemples pré-étiquetés (non utilisés pour l'élaboration de l'arbre)
Début
Pour chaque classe C, FAIRE
Si taux d'erreur (nœud remplacé par **C**, exemples) < taux d'erreur (nœud, exemples) **alors**
Remplacer nœud par **C**
Fin si
Fin pour chaque
Pour chaque fils de nœud **f, faire**
S = sous-ensemble d'exemples ayant pour racine le nœud **f**
Élaguer (f, S)
Fin pour chaque
fin

- **L'erreur apparente :**

L'erreur apparente d'une feuille représente le nombre d'exemples mal classifiés par cette feuille, celui de l'arbre est la somme des erreurs de toutes les feuilles. On calcule le taux d'erreur de classification avec la formule suivante :

Taux =nombre d'erreurs/nombre d'exemples

- **L'erreur réelle :**

C4.5 utilise une estimation de l'erreur réelle de l'arbre, cette estimation est produite à partir de **l'erreur apparente** de l'arbre.

On utilise une approche ascendante. Avec le paramètre de confiance **CF**. Pour chacune des feuilles de l'arbre, notons **N** le nombre d'exemples qu'une feuille couvre et **E** le nombre d'erreurs de classification qu'elle induit dans l'échantillon. Soit **p**, la probabilité pour qu'un nouvel exemple soit mal classé par cette feuille. La valeur de **p** est trouvée à l'aide de la fonction suivante, où **p** est une valeur entre **0** et **1** :

$$\mathrm{Ucf(N,E)} = P\left(\sum_{i=0}^{E}\binom{N}{i}p^i(1-p)^{N-i} \geq \mathrm{CF}\right)$$

Par la suite, on remonte jusqu'à la racine, en calculant l'estimation de l'erreur réelle de cet arbre en faisant une somme pondérée des estimations des erreurs réelles de ses fils.

1.1.4.3.4 Traitement de valeurs manquantes :

L'algorithme C4.5 a rendu possible la gestion des valeurs manquantes, deux possibilités se proposent pour remédier à ce manque :

- Éliminer l'enregistrement pour lequel la valeur est absente, et calculer le gain ou le gain ratio pour les enregistrements qui ne présentent pas de valeurs manquantes.
- Classer les enregistrements qui ont des valeurs inconnues en estimant la probabilité des différents résultats possibles.

1.1.4.3.5 Traitement des valeurs continues :

Le traitement des valeurs continue par C4.5 se fait par la discrétisation de ces dernières.
Si un attribut C_i a une intervalle de valeurs organisées en ordre(croissant ou décroissant) **A1,A2,…,Am** dans l'ensemble d'apprentissage, la discrétisation se fait en partitionnant les enregistrements, entre ceux qui ont une valeur inférieure ou égale à A_J et ceux ayant une valeur supérieur à A_J, nous obtiendrons ainsi des catégories **A** de la sorte :A_J **<A ,** A_J **>= A** , nous calculons le gain ou le gain ratio pour tout A_J , **J=1,m** et la partition prise est celle qui maximise le gain.

1.1.4.3.6 Avantages de l'algorithme C4.5 :

- Traitement des valeurs manquantes et valeurs continues.
- Permet l'élagage de l'arbre.

1.1.4.3.7 Inconvénients de l'algorithme C4.5 :

- Ne peut pas prédire les valeurs continues(régression).

1.1.4.4 L'algorithme CART :

1.1.4.4.1 Définition :

CART (Classification And Regression Tree) développé par Leo Breiman en 1984**[23]**, c'est un algorithme qui génère des arbres de décision binaire. L'acronyme CART correspond à deux situations bien distinctes, selon la nature de la variable à prédire, si qualitative alors l'algorithme permet de faire la classification, si quantitative alors régression.

Le déroulement de CART passe par deux phases, celle de la construction de l'arbre maximal, appelé aussi phase d'expansion et une phase de l'élagage de l'arbre.

1.1.4.4.2 La phase d'expansion :

Les mesures de segmentation de la construction de l'arbre différent selon les deux situations, classification ou régression, nous allons voir ces deux mesures et expliquer comment se déroule ce processus.

a. La classification :

Un arbre de classification CART est un arbre binaire utilisé pour prédire des valeurs qualitatives. La création d'un arbre de décision commence par le nœud racine et le choix de l'attribut adéquat élu selon un critère de segmentation, et de manière récursive refaire le processus pour les nœuds fils. Pour ce faire CART utilise les deux critères de segmentation, **Indice de Gini** et **Critère de Gini**, leurs formules seront présentées ci-dessous. Le nœud prend alors l'attribut qui maximise la valeur du critère de Gini.

L'arbre que l'algorithme CART génère est binaire, autrement dit chaque nœud accepte au plus deux fils. Dans un jeu d'apprentissage un attribut **A** peut avoir plus de deux valeurs **Ai**, cela nécessite donc une division dichotomique de l'ensemble d'apprentissage et enfin choisir la division qui maximise le **critère de Gini**.

- ✓ **Indice de Gini** : c'est une mesure de segmentation qu'on utilise pour le calcul du critère de Gini pour savoir l'attribut qui réalise la meilleure séparation, Il se calcule avec la formule suivante :

$$\mathbf{Gini(S) = 1 - \sum_{j=1}^{n} freq\ (Cj, S)^{2}} \quad \mathbf{(2,6)}$$

Avec :

Cj: La valeur **j** de la Classe **C**.

S : Ensemble d'apprentissage.

freq (**Cj**, **S**): La frequence de la classe **Cj** dans l'ensemble **S**

- ✓ **Critère de Gini :** Le critère de Gini est la mesure de segmentation de l'algorithme. Il mesure le niveau d'inégalité de la répartition d'une variable dans la population.

Il se calcule comme suit :

$$\mathbf{CritèreGini(p) = Gini(p) - (Pgauche * Gini\ (p1) + Pdroite * Gini\ (P2))} \quad \mathbf{(2,7)}$$

Tels que :

Gini(P) : indice de Gini de l'attribut **P**.

Pgauche : proportion de l'ensemble de gauche (**p1**) dans **S**.

Pdroite): proportion de l'ensemble de droite (**p2**) dans **S**.

Gini (p1), Gini (P2) : respectivement indice de Gini de **p1** et indice de Gini de **p2**

Dans un jeu d'apprentissage un attribut **A** peut avoir plus de deux valeurs $\mathbf{A_i}$, cela nécessite donc une division dichotomique de l'ensemble d'apprentissage et enfin choisir la division qui maximise le critère de Gini.

b. Pseudo code de l'algorithme CART :

CART (A, C, S)

Entrée :

S : données d'apprentissage, **A** : ensemble d'attributs, **C** : ensemble de classes.

Sortie : arbre de décision.

Début

Si tous les enregistrements de **S** ont la même valeur pour l'attribut classe **C alors**

Retourner un seul nœud feuille avec cette valeur.

Fin si

Si A est vide **alors**

Retourner un simple nœud avec comme valeur la valeur de la classe la plus trouvées dans **S**.

Fin si

Calculer indice de Gini **Gini(S)** de **S**

Pour tout attribut $\mathbf{A_j}$**, j :1-n faire** :

Si $\mathbf{A_j}$ n'est pas binaire **Alors** former toutes les combinaisons possibles de binarisation de $\mathbf{A_j}$

Prendre la combinaison qui a la plus grande valeur du **critère de Gini**

Sinon calculer le **critère de Gini** de $\mathbf{A_j}$

Fin pour

Soit Noeud l'attribut ayant le plus grand **critère de Gini** des attributs de **A**.

Retourner un arbre avec le nœud racine **Noeud** et les arcs sont étiquetés par $\mathbf{V_1, V_2, \ldots, V_m}$ (les valeurs de l'attribut **Noeud**).

$\mathbf{Sv_i}$ **i:1-m** : les sous-ensembles de **S** constitué respectivement des enregistrements de valeur $\mathbf{V_i}$ **i:1-m** de l'attribut $\mathbf{A_j}$ **j:1-n**.

Aller faire CART (A-{Noeud}, C, $\mathbf{SV_1}$**) . . . . CART(A-{Noeud}, C,** $\mathbf{SV_m}$**)**

Fin

c. Exemple d'application de l'algorithme CART :

Dans cet exemple nous allons dérouler l'algorithme CART pour l'ensemble de données du **tableau 2**:

- **1ére Itération :**

1) **Le calcul de l'indice de Gini(Gini) pour l'ensemble de données S :**

Indice de Gini (S) = 1-(freq(oui)^2 +freq(non)^2)

$$= 1 - \left(\left(\frac{3}{5}\right)^2 + \left(\frac{2}{5}\right)^2\right) = \mathbf{0.48}$$

Où : freq(oui), freq(non) représentent respectivement le nombre d'enregistrement ou la valeur de la classe **« Jouer» =** oui ,non diviser par la taille de l'ensemble de données .

2) **Le calcul de critère de Gini pour chaque attribut de l'ensemble de données :**

- **Pour l'attribut Ciel :**

Critère de Gini (ciel) : Gini (s) **-((Pgauche × Gini(S gauche)) +(Pdroit × Gini(S droit))**

Nous allons binariser l'attribut ciel, nous aurons alors trois combinaisons de binarisation :

Division	**Gauche**	**Droit**
1ére division	Ensoleillé	(Nuageux, Pluvieux)
2éme division	Nuageux	(Ensoleillé, Pluvieux)
3éme division	Pluvieux	(Ensoleillé, Nuageux)

Tableau 10: Binarisation de l'attribut ciel

La prochaine étape consiste à calculer le **critère de Gini** pour chaque combinaison de la façon suivante :

Le critère de Gini (1ére division) =

Gini (s) -((P Ensoleillé x Gini(S Ensoleillé) + (P(Nuageux, Pluvieux) x Gini(**s(Nuageux, Pluvieux)**)

- **P Ensoleillé =2/5**(le nombre d'occurrence de la valeur **Ensoleillé** pour l'attribut **ciel** diviser par la taille **S Ensoleillé**).
- **P (Nuageux, Pluvieux)=3/5**(le nombre d'occurrence de la valeur **Nuageux** ou **Pluvieux** pour l'attribut **ciel** diviser par la taille de **s (Nuageux, Pluvieux))**.

Gini(S Ensoleillé) = 1- $((2/2)^2)$ = **0**

Gini(s ($_{\text{Nuageux, Pluvieux}}$) = $1-((2/3)^2+(1/3)^2)$ **= 0.44**

Après remplacement nous trouvons **:**

Le critère de Gini (1$^{\text{ére}}$ division) = 0.48 -((2/5 x 0) + (3/5 x 0.44)) =**0.216**

De la même manière **le critère de Gini** des deux autres divisions sont respectivement :

Le critère de Gini (2$^{\text{éme}}$ division) =**0.18**, **Le critère de Gini (3$^{\text{éme}}$ division)** = **0.01**.

- **Pour l'attribut Température :**

Le critère de Gini (Température)=

Gini(s) - ((P $_{\text{chaud}}$ x Gini(S $_{\text{chaud}}$)) + (P$_{\text{frais}}$ x Gini(s($_{\text{frais}}$))

Gini(S $_{\text{chaud}}$)= $1-((2/3)^2+(1/3)^2)$ **= 0.44, P$_{\text{chaud}}$ = 3 /5.**

Gini(s($_{\text{frais}}$)= $1-((1/2)^2+(1/2)^2)$ = **0.5**, **P$_{\text{frais}}$** = **2/5**.

Après replacement :

Le critère de Gini (Température) = 0.48 - ((3 /5 x 0.44)+(2/5 x 0.5))= **0.016**

De la même manière de calcul nous avons trouvé le **critère de Gini** d'autres attributs les résultats sont les suivant :

Le critère de Gini (Humidité)=0.016.

Le critère de Gini (Vent)=0.216.

L'attribut **ciel (1$^{\text{ére}}$ division)** et l'attribut **vent** maximise **le critère de Gini** ,un choix entre ces deux attributs est effectué pour monter en racine .Dans notre cas nous choisissons l'attribut **ciel (1$^{\text{ére}}$ division**) la **figure6** montre l'arbre de décision résultant de cette première itération .

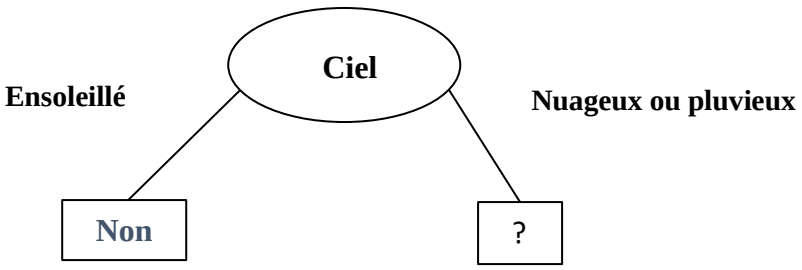

Figure6: Arbre de décision résultant de première itération.

- **2 éme Itération :**

Nous referons le processus de la première itération avec l'ensemble de données où la valeur de l'attribut **ciel** est égale à **Nuageux ou Pluvieux,** à la fin de cette dernière nous obtiendrons l'arbre de décision final montré dans la **figure7.**

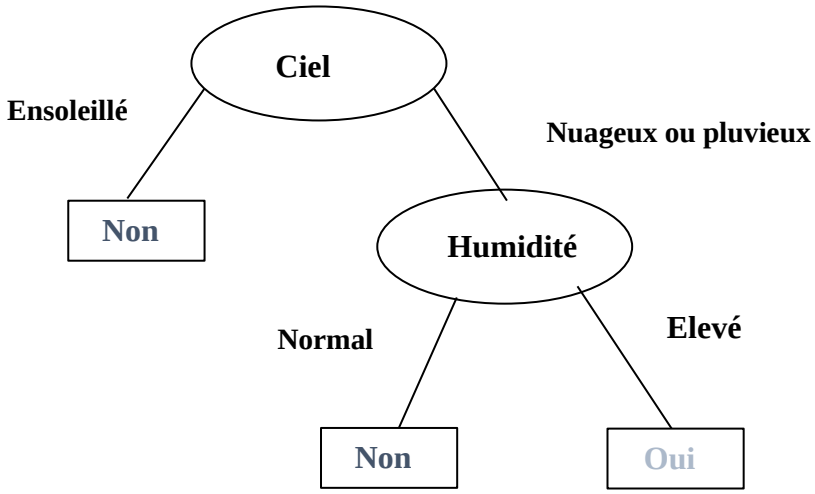

Figure7: Arbre de décision final résultant de la 2 éme itération.

d. La régression :

Un arbre de régression CART est un arbre utilisé pour expliquer des variables quantitatives, les nouveaux critères de segmentation sont la variance et la variance intra-segment.

- **Variance (y) = Var =** $\frac{\sum_{i=1}^{n}(yi - \bar{y})^2}{n}$ **(2,7)** / $\bar{y} = \frac{\sum_{i=1}^{n} yi}{n}$ **(2,8)**

Tels que :

y : la variable à expliquer

n : le nombre d'individus.

$\bar{y}$ **:** La moyenne de y.

- **La variance intra-segment :**

$$\frac{(\mathbf{var(fg)\times n1)+(var(fd)\times n2)}}{\mathbf{n}} \quad \mathbf{(2,9)}$$

Où **:**

Fg, fd : Représente respectivement fils gauche et fils droit (les valeurs d'attributs).

$\mathbf{n_1}, \mathbf{n_2}$ **:** Représente respectivement le nombre d'enregistrements qui ont la valeur **fg** et le nombre de d'enregistrements qui ont la valeur **fd**.

n : le nombre d'enregistrement dans la base d'apprentissage.

A chaque itération, l'attribut sélectionné est celui qui minimise la **variance intra-segment**, nous obtiendrons une feuille lorsque pour un sous ensemble $\mathbf{S_v}$ (les enregistrements pour lesquels un attribut **A** égale à la valeur **v**) pour chaque attribut **X**, les enregistrements de l'ensemble d'apprentissage prennent la même valeur **i** d'un attribut **X**, la feuille prendra la valeur de la moyenne de $\mathbf{A_v}$.

1.1.4.4.3 La phase d'élagage :

L'achèvement de la phase d'expansion produit un arbre maximal, cet arbre tend à être excessivement raffiné, mais l'objectif étant la construction d'un modèle robuste et parcimonieux, ce dernier est réalisé par une procédure d'élagage de l'arbre (**purning** en anglais).

La démarche consiste à construire une suite d'arbre $\mathbf{T_0, T_1, \ldots, T_k}$ ($\mathbf{T_0}$ étant l'arbre maximal), par élagage successif jusqu'à ce que $\mathbf{T_k}$ soit une feuille, les nœuds banni de l'arbre sont ceux qui minimise une fonction d'erreur g, l'algorithme qui suit est un algorithme dédié à l'élagage d'un arbre CART :

Élagage CART

Entrée : l'arbre de décision obtenu dans la phase d'expansion.

On construit ne suite d'arbres $\mathbf{t_0, t_1 \ldots, t_K}$

Soit :

T_0 : arbre obtenue dans la phase d'expansion.

T_k : une feuille

A l'étape t_j :

Pour chaque t_j/ j=0, K :

Calculer l'erreur apparente sur l'ensemble p.

Pour chaque t_j pour **j=0, k** calculer l'erreur apparente sur l'ensemble **T**.

Pour toute position **P** de $\mathbf{t_j}$ on calcule **g(p)**

Choisir la position **P** qui minimise la fonction **g (tk,p).**

Élaguer l'arbre à la position **p**

L'arbre $\mathbf{t_j}$ **+1** est un élagué de $\mathbf{t_j}$ en position **P**.

Sortie : L'arbre de la suite dont l'erreur apparente est minimale.

- **La fonction g(p)** : mesure le critère pour choisir le nœud qu'on va être élagué.

$$G(p) = \frac{A-B}{C \times (D-1)} \quad (2,10)$$

A : le nombre d'exemples mal classés du jeu d'apprentissage par le nœud **p** de l'arbre T_k quand on fait l'hypothèse qu'il a été transformé en feuille

B : le nombre d'exemples mal classés par les feuilles du nœud T_k situé sous le nœud **p**

C : le nombre de feuilles de l'arbre

D : le nombre de feuilles du sous arbre de T_k situé sous le nœud

1.1.4.4.4. Avantages de l'algorithme CART :

- Simple à comprendre, interpréter et visualiser.
- Pas de complexité de paramétrage
- La mixité des variables.

1.1.4.4.5. Inconvénients de l'algorithme CART :

- Binarisation pas toujours approprié.
- Cout de calcul : assez lent pour les gros ensembles de données
- les arbres de décision peuvent être instables.

1.2. La classification Bayésienne :

Les classifieurs bayésiens sont les classifieurs statistiques basés sur le **théorème de Bayes**, ils peuvent prédire la probabilité qu'un tuple donné appartienne à une classe particulière.

On distingue deux techniques de classification bayésienne : la classification naïve bayésienne et les réseaux bayésiens.

1.2.1. Théorème de Bayes :

Une référence théorique pour les approches statistiques de résolution des problèmes de classification. Soit **X** un échantillon de données dont la classe est inconnu et qu'on voudrait déterminer.

Soit **C** une classe. On cherche à déterminer **P(C|X)** la probabilité de vérification de **C** après l'observation de **X**.

$$p(C/X) = \frac{p(X/C) * p(C)}{P(X)} \quad (2,11)$$

Tels que :

- P ($\boldsymbol{C}|\boldsymbol{X}$) **: (la probabilité à posteriori**) la probabilité après la connaissance de **X** (la probabilité d'appartenance de X à la classe **C**).
- P ($\boldsymbol{C}$) **: (La probabilité à priori)** la probabilité d'appartenance de la classe **C** dans la population (la fréquence de la classe **C**).
- P($\boldsymbol{X}|\boldsymbol{C}$) : **(la probabilité conditionnel)** la probabilité d'apparence de chaque valeur des attributs de **X** dans les attributs des échantillons appartenant à la classe **C** :

$$\mathbf{P}(\boldsymbol{X}|\boldsymbol{C}) = \prod_{i=1}^{n} \mathbf{P}(\mathbf{ai} = \mathbf{vi} \,|\mathbf{C}) \quad \textbf{(2,12)}$$

Tels que **: ai:** $\boldsymbol{i}^{\boldsymbol{éme}}$ attribut de **X** et **vi:** sa valeur**.**

1.2.2. La classification naïve bayésienne :

C'est un type de classification bayésienne probabiliste simple basée sur le théorème de Bayes et les probabilités conditionnelles. Utilisé pour les problématiques de classement avec des données qualitatives, quantitatives continues et textuelles supposé indépendantes d'où le terme « naïve ».

1.2.2.1. Pseudo code de la classification bayésienne :

Classifieur naïve bayesien (A, C, S, X)

Entrée :

S : données d'apprentissage, **A** : ensemble d'attributs, **C** : ensemble de classes, **X** : élément à classer.

Sortie : Décision.

Début

Pour chaque classe de C_i i :1-n de **C faire :**

Calculer la probabilité à priori **P (C_i)**

Pour chaque attribut **A_j j=1, m** de **A**

Pour chaque valeur **V_k k=1, k** de **A_j**

Calculer les probabilités conditionnelles **P (V_k /C_i)**

Calculer le numérateur de bayes pour l'hypothèse **X** appartient à la classe **C_i (P(X/C_i) *P(C_i)**

Affecter X à la classe avec le plus grand **numérateur de bayes.**

Fin.

1.2.2.2. Exemple d'application de la classification naïve bayésienne :

Nous prenons l'ensemble d'apprentissage du **tableau 1**, **page 5**

Étape 1 : Calculer les probabilités à priori pour les deux classes (oui, non) :

P(Oui) = 2/5 , **P(Non)** = 3/5

Étape 2 : Calculer les probabilités conditionnelles pour chaque attribut :

Ciel			
P (nuageux /oui)	1/2	P (nuageux /non)	0
P (pluvieux/oui)	1/2	P (pluvieux/non)	1/3
P (ensoleillé/oui)	0	P (ensoleillé/non)	2/3

Température			
P (chaud /oui)	1/2	P (chaud /non)	2/3
P (frais/oui)	1/2	P (frais /non)	1/3

Humidité			
P (normal /oui)	1/2	P (normal /non)	1/3
P (élevé/oui)	1/2	P (élevé /non)	2/3

Vent			
P (fort /oui)	0	P (fort /non)	2/3
P (faible/oui)	1	P (faible /non)	1/3

Étape 3 : trouver la classe d'un élément **X<nuageux, frais, élevé, fort> :**

- Calculer le numérateur de bayes pour l'hypothèse **: X** appartient à la classe **oui :**

  P(x/oui) *P(oui)= P (nuageux /oui) * P (frais/oui) * P (élevé/oui) * P (fort /oui) * P(oui) **= 0**

- Calculer le numérateur de bayes pour l'hypothèse **: X** appartient à la classe **oui :**

  P(x/non) *P(non)= P (pluvieux /non) * P (frais/non) * P (élevé/non) * P (fort /non) * P(non)=0.02

- Comparer les deux résultats précédents : **P(x/non) *P(non)> P(x/oui) *P(oui)**
- **Décision** : **X** appartient à la classe **non.**

1.2.2.3. Avantages de la classification bayésienne :

- Robuste aux attributs manquants.
- Peut prendre en compte tous les types d'attributs (discrets et continus).
- Vitesse de la classification et de l'apprentissage.

1.2.2.4. Inconvénients de la classification bayésienne :

- Nécessité de discrétiser les attributs ayant des valeurs numériques pour pouvoir calculer les probabilités.
- Faible résistance au sur-apprentissage.

1.2.3. Les réseaux bayésiens :

Les réseaux Bayésiens constituent un ensemble de méthodes statistiques utilisées pour modéliser des problèmes, extraire de l'information et prendre des décisions. Ils sont un formalisme de raisonnement probabiliste de plus en plus utiliser dans plusieurs domaines tels que l'industrie, la finance et le traitement d'images.

C'est un graphe orienté acyclique, ou les nœuds représentent des variables aléatoires et les arcs représentent soit :

- Des dépendances probabilistes entre variables.
- Des distributions probabilistes conditionnelles pour chaque variable.

Pour chaque nœud on associe une table de probabilité conditionnelle, qui donne sa probabilité étant donné les valeurs de ses parents. Si un nœud n'a pas de parents on parle de probabilité inconditionnelle dans le cas des variables continues les probabilités sont définies par des fonctions de densité de probabilité. Voir **la figure 8.**

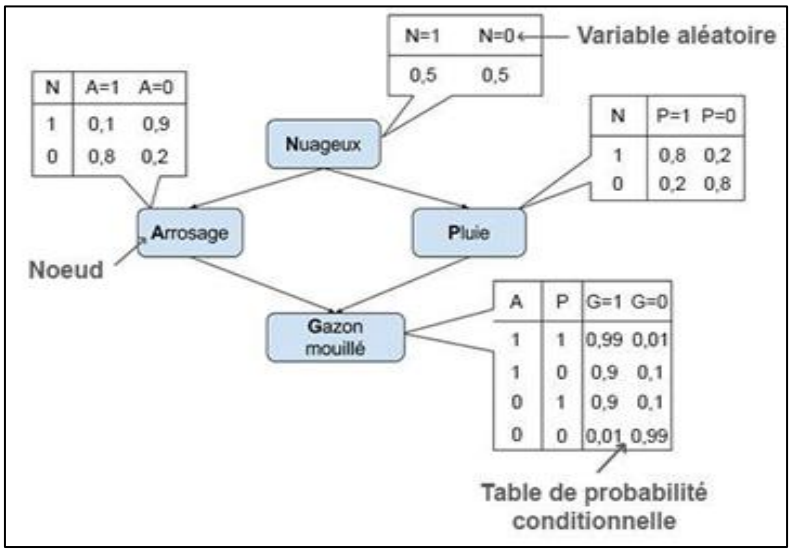

Figure8: Exemple d'un réseau bayésien. [24]

1.3. Machine à vecteur de support (SVM) :

1.3.1. Définition :

Les machines à vecteurs de support **(**également les réseaux à vecteurs de support**)** sont des modèles d'apprentissage supervisé linéaires binaires proposée par Corinna Cortes et Vapnik en 1993**[25].** Il peut être utilisé à la fois pour des problèmes de classification et de régression.

1.3.2. Principe de SVM :

Un modèle SVM est une représentation des données sous forme de points dans l'espace, cartographiée de manière à ce que les points des catégories distinctes soient divisés par un espace clair aussi large que possible.

SVM étudie les données d'apprentissage étiquetées puis classe toutes les nouvelles données d'entrée en fonction de ce qu'il a appris au cours de la phase d'apprentissage.

1.3.3. Concept de base :

a. Hyperplan :

Dans un espace vectoriel de dimension finie **n**, un hyperplan est un sous-espace Vectoriel de dimension **n-1**. Ainsi, dans un espace de dimension **2** un hyperplan sera une droite, dans un espace de dimension **3** un hyperplan sera un plan. Dans ce cas c'est un hyperplan de séparation (une ligne qui sépare deux classes différentes) comme nous pouvons le voir sur le graphique ci-dessous. Son équation est donnée comme suit **< w. x > +b=0** tels que :

w : définit la pente de l'hyperplan (perpendiculaire à l'hyperplan).

b : biais (permet de translater l'hyperplan parallèlement à lui-même).

< w, xi > : c'est le produit scalaire entre les deux vecteurs w et xi.

W et b sont les paramètres à estimer de la fonction de décision **h(x).**

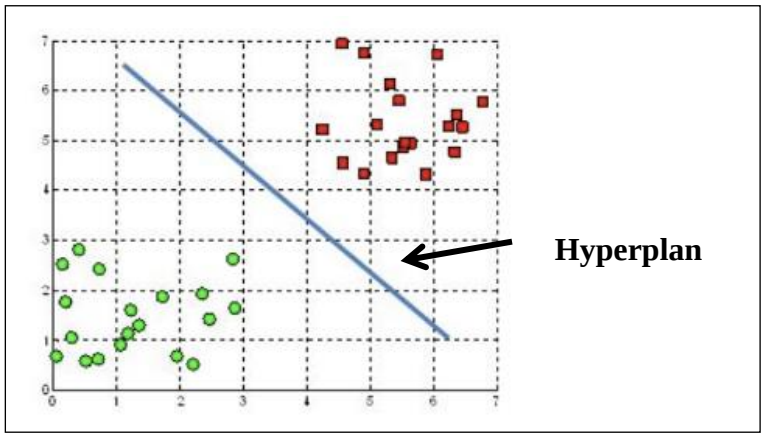

Figure9: Hyperplan de séparation de deux classes « cubes en rouge » et « les boules vertes » [26]

b. Une fonction de décision :

La fonction de décision **h(x)** c'est une fonction qui sépare les données en deux catégories données comme suit :

$$h(x) = \langle w, xi \rangle + b = \sum_{i=1}^{n} < w, xi > + b$$

- **<w, xi> +b > 0** si **Yi= +1**
- **<w, xi> +b < 0** si **Yi= -1**

c. Les vecteurs de supports :

Les vecteurs de supports sont des points de données plus proches de l'hyperplan. En utilisant ces vecteurs supports, nous maximisons la marge du classifieur. La suppression des vecteurs de supports modifiera la position de l'hyperplan. Ce sont les points qui nous aident à construire notre SVM.

1.3.4. Le choix de l'hyperplan optimal :

Le choix de l'hyperplan si vous regardez la **figure10**, plusieurs hyperplans peuvent séparer les points de données en deux classes. L'hyperplan optimal est donc celui qui divise très bien les points de données (les données des deux classes ne sont pas mélangées) qui est dessiné sur la base de ses vecteurs de support et qui aura une distance maximale de chacun des vecteurs de support.

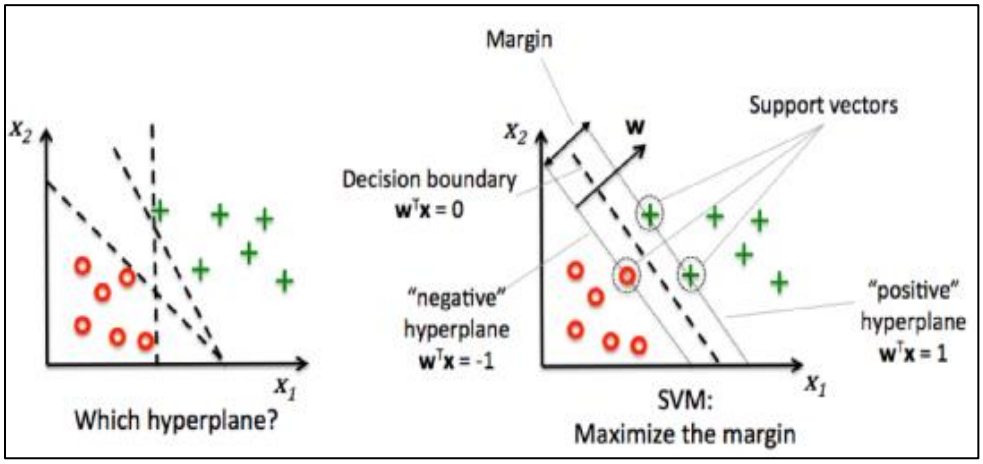

Figure10: le choix d'un hyperplan optimal qui maximise la marge.[27]

1.3.4.1 La marge maximale :

- La marge est la distance entre l'hyperplan et les points les plus proches.
- Dans les SVM, l'hyperplan est choisi comme celui qui maximise la marge.
- La marge maximale est égale à : **marge =1/** $\|w\|^2$
- Maximiser la marge revient à minimiser la norme du vecteur de paramètres **w.**

1.3.5. Discrimination linéaire :

On dit que les données sont linéairement séparables s'il existe un hyperplan qui classe correctement tous les exemples de l'échantillon de données **S**. Tandis qu'il arrive souvent que même si le problème est linéaire, les données sont affectées par un bruit et les deux classes se retrouvent mélangées autour de l'hyperplan de séparation. Dans ce cas, le meilleur hyperplan, c'est l'hyperplan qui va introduire l'erreur minimum.

1.3.6. Discrimination non linéaire *:*

Dans le cas où les données sont non linéairement séparables c'est-à-dire qui il n'existe pas d'hyperplan capable de séparer correctement les deux catégories.
L'idée des SVM est de changer l'espace des données, la transformation non linéaire des données peut permettre une séparation linéaire des exemples dans un nouvel espace.
On va donc avoir un changement de dimension. Cette nouvelle dimension est appelée « **Espace de redescription ».**

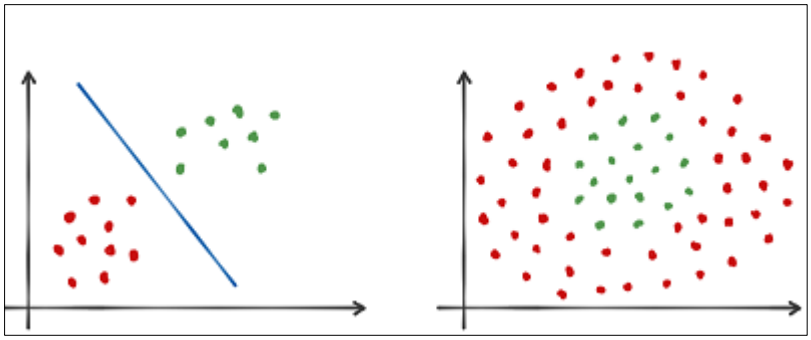

Figure 11:Un schéma qui montre la discrimination linéaire et non linéaire.[28]

1.3.7. SVM multi classe :

Grâce aux nombreuses améliorations apportées au fil des ans, il existe actuellement plusieurs méthodes pour effectuer une classification multi-classes avec des SVM. Citons les plus utilisé:

- **Un-contre-tous :**

Également appelé **«One-versus-the-rest»** le but c'est de distinguer une classe de toutes les autres, pour une classe donnée nous construisons **K** classifieurs binaires différents. Ainsi les exemples positifs sont tous les points de la classe et les exemples négatifs sont tous les points qui sont hors classe (voir la figure ci-dessous).

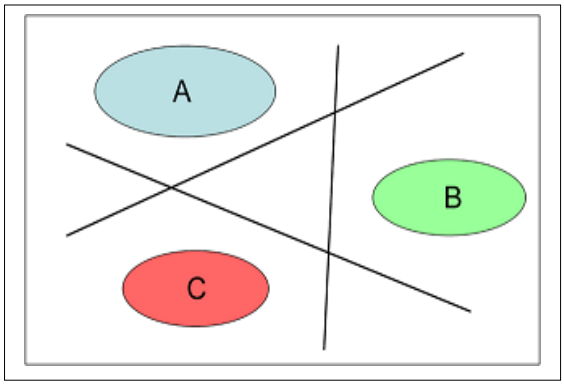

Figure 12: SVM multi classe -One-against-all-[29]

- **Un-contre-un:**

Également appelée **«One-against-one»** dans cette approche,nous cherchons à distinguer une classe d'une autre. En conséquence, nous formons un classificieur par paire de classes, ce qui conduit à **K (K-1) / 2** classifiieurs pour les classes K. (voir **la figure 13**).

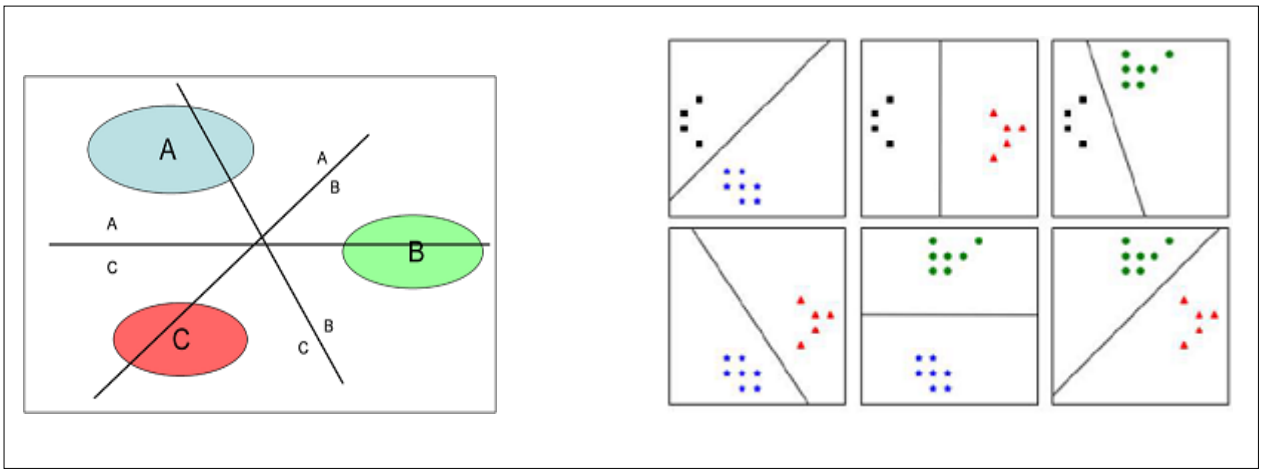

Figure 13: SVM multi classes -One-against-one-[29]

Les applications courantes de l'algorithme SVM sont le système de détection d'intrusion, la reconnaissance de l'écriture manuscrite, la prédiction de la structure des protéines etc...

1.3.8. Les avantages des SVM :

- Résistance au sur-apprentissage.
- Ne nécessite pas un grand nombre de paramètres à régler.

1.3.9. Les inconvénients des SVM :

- Le traitement des problèmes multi-classes reste une question ouverte.
- L'efficacité dépend du choix de la fonction noyau.
- Les SVM ne résiste pas aux valeurs manquantes.

1.4. Algorithme des K-plus proche voisins :

1.4.1 Définition :

Le **K-PPV** est le diminutif de **K plus proches voisins** en anglais **K-nearest neigbors (K-NN)** proposé par Thomas Cover **[30]** est un algorithme de classification peut être utilisé aussi pour résoudre des problèmes de régression. Son principe consiste en effet à choisir les K données les plus proches (voisins) du point étudié afin de prédire sa classe. Il est non paramétrique (seule k qui doit être fixé) et il nécessite le choix d'une mesure de distance auparavant. L'algorithme K-NN est qualifié comme paresseux (Lazy learning) car il n'apprend rien pendant la phase d'entrainement.

Pour prédire la classe d'un nouvel exemple « **X** », l'algorithme calcule la distance de **X** avec chaque exemple de la base d'apprentissage afin de trouver les « **K** » plus proches voisins de **X**. Enfin, la classe majoritaire parmi les **N** classes, sera attribuée à **X.**

1.4.2 . Pseudo code :

Algorithme K-NN

Données en entrée :

Un ensemble de données **D** .

Une fonction de distance **d** .

Un nombre entier **K.**

Un objet **x.**

Données en sortie :

Les classes avec le nouvel élément.

Début :

Soit **D = {(x′, c), c : classe}** l'ensemble d'apprentissage.

Soit **x** l'exemple dont on souhaite déterminer la classe.

Pour chaque ((x′, c) ∈ D) faire

Calculer la distance **dist(x, x′)**

fin

pour chaque {x′ ∈ kppv(x)} faire

compter le nombre d'occurrence de chaque classe

fin

Attribuer à **x** la classe majoritaire (qui contient le plus de voisins);

Fin.

La figure 14 illustre un exemple d'application de K-NN, dont le but est de trouver la valeur de la classe d'un exemple inconnu x des trois classes **w1,w2 ,w3** . En utilisant la distance Euclidienne et **k=5** voisins, des 5 plus proches voisins ,4 appartiennent à **w1** et 1 appartient à **w3**, donc x sera affecté à **ω1** qui est la classe majoritaire.

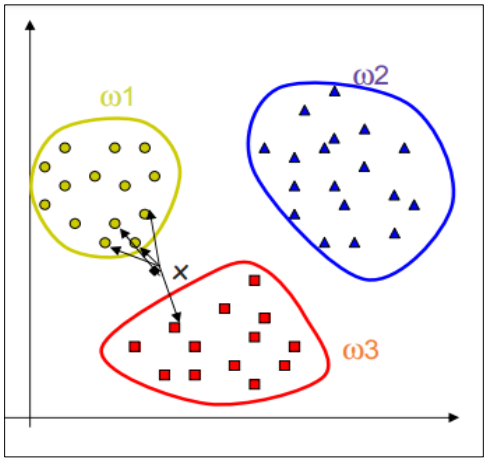

Figure 14:Exemple d'application de l'algorithme KNN [31]

1.4.3. Le choix de la valeur K :

La valeur **K** à utiliser pour effectuer une prédiction avec K-NN, varie en fonction du jeu de données. En règle générale, moins on utilisera de voisins (un nombre **K** petit) plus on sera sujette au sous apprentissage (underfitting). Par ailleurs, plus on utilise de voisins (un nombre K grand) plus, on sera fiable dans notre prédiction. Toutefois, si on utilise **K** nombre de voisins avec **K=N** et **N** étant le nombre d'observations, on risque d'avoir un sur apprentissage (overfitting) et par conséquent un modèle qui se généralise mal sur des observations qu'il n'a pas encore vu.

1.4.4. Les avantages de l'algorithme K-NN :

- Simple et facile à mettre en œuvre.
- Adapté tout type de données.

1.4.5. Les inconvénients de l'algorithme K-NN

- Le choix de la valeur K et de la mesure de distance reste une limite pour cet algorithme.

1.5. Réseaux de neurones :

Le concept des réseaux de neurones artificiels (Artificial Neural Network) fut inventé en 1943**[32]** par deux chercheurs de l'Université de Chicago : le neurophysicien Warren McCullough, et le mathématicien Walter Pitts. Son fonctionnement est calqué sur celui des neurones du cerveau humain.

Il s'agit là d'une variété de technologie **Deep Learning (apprentissage profond)**[14], qui représente un ensemble de neurones formels interconnectés permettant la résolution de problèmes complexes tels que la reconnaissance des formes ou le traitement du langage naturel. Il existe différentes variantes de réseaux de neurones qui se distinguent l'une de l'autre selon l'architecture du réseau parmi eux, Feedforward, réseau de neurones récurrent (RNN), réseau de neurones convolutifs (CNN) nous allons en détailler quelques-uns dans le deuxième chapitre.

1.5.1 Motivation biologique :

La motivation derrière le réseau neuronal est le cerveau humain. De nombreux chercheurs ont pensé à fabriquer une machine qui fonctionnerait dans la perspective du cerveau humain.

[14] **Deep learning:** (ou apprentissage profond) est un ensemble de méthodes d'apprentissage automatique conçues sur la base de réseaux de neurones profonds, visant à mimer la « profondeur » des couches d'un cerveau.

Le cerveau humain contient des milliards de neurones qui sont connectés à de nombreux autres neurones pour former un réseau, de sorte que s'il voit une image, il reconnaît l'image et traite la sortie (voir la figure ci-dessous).

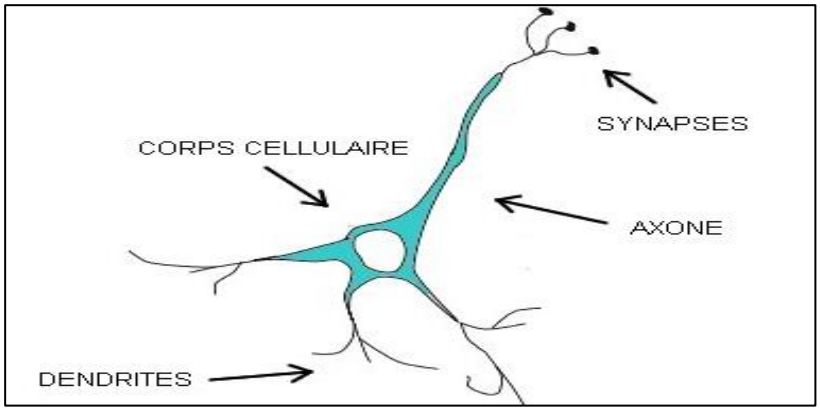

Figure15:Neurone biologique [33]

neurones Biologique	neurones Artificiels
Synapse	Poids des connexions
Axones	Signal de sortie
Dendrites	Signal d'entrée
Noyau ou Somma	Fonction d'activation

Tableau 11 : Analogie entre le neurone biologique et le neurone formel.[34]

1.5.2 Concepts de base :

Avant d'expliquer ce qu'est un neurone artificiel et comment il fonctionne, nous allons définir ces concepts :

- **Entrées :** Données sources alimentées dans le réseau neuronal, dans le but de prendre une décision ou une prédiction sur les données. Les entrées d'un réseau neuronal sont généralement un ensemble de valeurs réelles ; chaque valeur est introduite dans l'un des neurones de la couche d'entrée.
- **Ensemble d'entraînement :** Un ensemble d'entrées pour lesquelles les sorties correctes sont connues, utilisées pour entraîner le réseau neuronal.

- **Les sorties :** Les réseaux de neurones génèrent leurs prédictions sous la forme d'un ensemble de valeurs réelles ou de décisions booléennes. Chaque valeur de sortie est générée par l'un des neurones de la couche de sortie.
- **Neurone / perceptron :**

L'unité de base du réseau neuronal. Accepte une entrée et génère une prédiction.

- **Poids** :C'est un paramétre qui transforme les données d'entrée dans un réseau neuronal, dans chaque nœud se trouve un ensemble d'entrée,de poids et valeur de biais,chaque entrée d'un neurone est multiplié par un poids , la sortie résultante est soit observée, soit transmise à la couche suivante du réseau**.**
- **Biais :**

Le biais reflète l'adéquation du modèle avec l'ensemble d'entraînement. Un biais élevé signifie que le réseau neuronal n'est pas en mesure de générer des prédictions correctes, même pour les exemples sur lesquels il s'est entraîné.

- **Fonction d'activation :**

Une fonction d'activation est une équation mathématique qui détermine la sortie de chaque élément (perceptron ou neurone) dans le réseau neuronal. Il prend l'entrée de chaque neurone et la transforme en sortie, généralement entre **(1)** et **(0)** ou entre **(-1)** et **(1)**.

Dans un réseau neuronal, les entrées, qui sont généralement des valeurs réelles, sont introduites dans les neurones du réseau. Chaque neurone a un poids, et les entrées sont multipliées par le poids et introduites dans la fonction d'activation. La sortie de chaque neurone est l'entrée des neurones de la couche suivante du réseau, et donc les entrées se succèdent à travers de multiples fonctions d'activation jusqu'à ce que finalement, la couche de sortie génère une prédiction.

1.5.3 Neurone artificiel :

Le perceptron, encore appelé neurone artificiel ou neurone formel, cherche à reproduire le fonctionnement d'un neurone biologique. Il existe différents niveaux d'abstraction, suivant la précision de la modélisation voulue. Considérons notre neurone sur **la figure16** et explicitons son fonctionnement :

Des signaux **$x_1, x_2, x_3, \ldots x_n$** arrivent à notre neurone, chaque lien qui amène le signal est pondéré, respectivement **$w_1, w_2, w_3, .. w_n$** c'est ce poids **(weight)** qui va être adapté tout au long de l'apprentissage pour permettre au réseau de prédire efficacement (en général il reste entre **0** et **1** ou **-1** et **1** ensuite il calcule la somme de tous ces signaux pondérés **($\sum_{i=0}^{n}$ wi · xi)** et il ajoute un certain biais **b**. Ce biais peut être vu comme un neurone externe supplémentaire qui envoie systématiquement le signal **1** de poids **b** au neurone. Grâce à lui, la fonction d'activation va être décalée et le réseau aura donc de plus grandes opportunités d'apprentissage.

Une fois cette somme calculée, on applique une fonction d'activation (activation function) pour obtenir notre signal de sortie. La formule de sortie d'un neurone caché sera donc toujours de la forme : **y = fonction d'activation** ($b + \sum_{i=0}^{n}$ **wi · xi**).

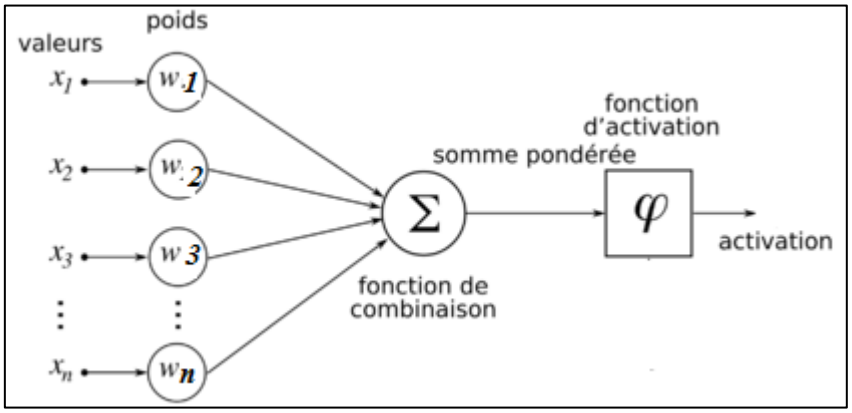

Figure 16:Structure d'un neurone artificiel [35]

1.5.4. L'apprentissage des réseaux de neurones :

Lorsque nous commençons avec le réseau de neurones, nous initialisons les poids au hasard. Évidemment, cela ne donnera pas de très bons résultats. L'ajustement d'un réseau de neurones implique l'utilisation d'un ensemble de données d'entraînement pour mettre à jour les pondérations du modèle afin de créer un bon mappage des entrées aux sorties.

Les techniques d'apprentissage des réseaux de neurones formels sont des algorithmes d'optimisation qui cherchent à minimiser l'écart entre les réponses réelles du réseau et les réponses désirées en mettant à jour les pondérations, ainsi un changement dans les performances du réseau à chaque itération. La formation des réseaux de neurones est très difficile. Le meilleur algorithme général connu pour résoudre ce problème est **la descente de gradient stochastique**[15], où les poids du modèle sont mis à jour à chaque itération en utilisant **l'algorithme de retropropagation de l'erreur**.

1.5.5. L'algorithme de rétropropagation :

Le principe de rétropropagation est de modéliser une fonction donnée en modifiant les pondérations internes des signaux d'entrée pour produire un signal de sortie attendu. Peut être utilisée à la fois pour les problèmes de classification et de régression.

Le réseau de neurone est entrainé à l'aide d'une méthode d'apprentissage supervisé, dans laquelle l'erreur entre la sortie du réseau et une sortie attendue connue est présentée au système et utilisée pour modifier son état interne.

1.5.6. Le sur-ajustement et le sous-ajustement dans les réseaux de neurones :

Lors de la phase d'entrainement des réseaux de neurones deux problèmes peuvent survenir :

- **Le sur-ajustement (Overfitting)** : c'est une situation ou le réseau neuronal est bon pour apprendre son ensemble d'entraînement, l'erreur sur cet ensemble est très petite, mais lorsque des nouvelles données sont présentées au réseau, l'erreur est importante donc le réseau a bien mémorisé les exemples de formation, mais il n'a pas appris à généraliser à des nouvelles données.
- **Le sous-ajustement (Underfitting) :** Se produit lorsque le réseau neuronal n'est pas en mesure de prédire avec précision pour l'ensemble d'apprentissage, ni sur celui de validation (nouvelles données). Cela veut dire que le réseau n'a pas appris assez et il réalise de mauvaises prédictions sur le jeu d'entrainement.

[15] **la descente de gradient stochastique :** C'est une méthode de descent de gradient (itérative) utilisée pour la minimisation d'une fonction objectif qui prendre la forme d'une somme de fonctions différentiables.

1.5.7. Les types des réseaux de neurones :

1.5.7.1. Les réseaux de neurones artificiels (ANN) :

En anglais **A**rtificial **n**eural **n**etwork (ANN), sont constitués d'un grand nombre de neurones (perceptron). Chaque neurone peut prendre des décisions simples et alimente ces décisions vers d'autres neurones, organisés en couches interconnectées comme le montre la figure ci-dessous. ANN est également connue sous le nom de réseau neuronal Feed-Forward car les entrées sont traitées uniquement dans le sens direct : c'est un réseau neuronal « peu profond » ne comporte que trois couches de neurones :

- **Une couche d'entrée (Input layer) :** C'est la première couche du réseau neuronal. Elle prend les signaux d'entrée (variables) elle les transmet à la couche suivante, elle n'applique aucune opération sur les entrées.

- **Une couche cachée (Hidden layer) :** La couche cachée se compose de neurones qui appliquent différentes transformations aux données d'entrée elle contient la fonction de sommation et d'activation.

- **Une couche de sortie (Output layer)** : Cette couche est la dernière couche du réseau et reçoit en entrer la sortie de la dernière couche cachée, et en fin elle génère des prédictions.

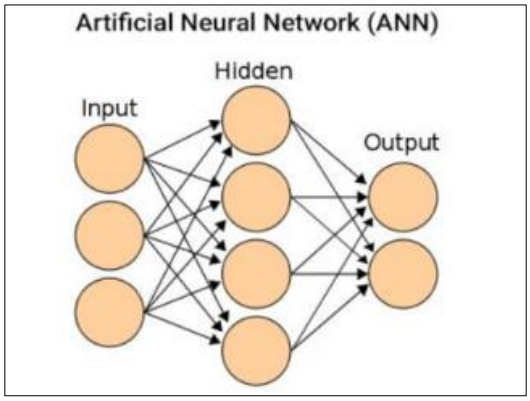

Figure 17:structure d'un réseau de neurone artificiel ANN [36]

1.5.7.2. Le perceptron multi couche :

(**PMC** ou **multi-layer perceptron**) C'est est un groupe de perceptrons, organisés en plusieurs couches, qui peuvent répondre avec précision à des questions complexes. Chaque perceptron de la première couche (à gauche) envoie des signaux à tous les perceptrons de la deuxième couche, etc. Un PMC contient une couche d'entrée, au moins une couche cachée et une couche de sortie.

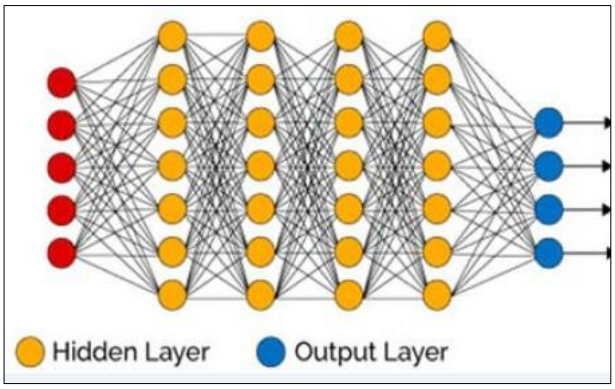

Figure 18:Structure d'un perceptron multicouche [37]

1.5.7.3. Les réseaux de neurones convolutifs :

En anglais **c**onvolutional **n**eural **n**etwork **CNN**, est une classe de réseaux de neurones profonds qui est le plus couramment appliquée à l'analyse de l'imagerie visuelle, il consiste en un empilage multicouche de perceptrons, dont le but est de prétraiter de petites quantités d'informations. Contrairement aux architectures traditionnelles de perceptrons multicouches, il utilise deux opérations appelées « Convolution » et regroupement « Pooling » pour réduire une image en ses caractéristiques essentiels, et utilise ces caractéristiques pour comprendre et classer l'image. Il existe quatre types de couches pour un réseau de neurones convolutif : **La couche de convolution**, **la couche de mise en commun (pooling)**, **la couche de correction ReLU** et **la couche entièrement connectée (fully-connected)** voir la **figure 19**

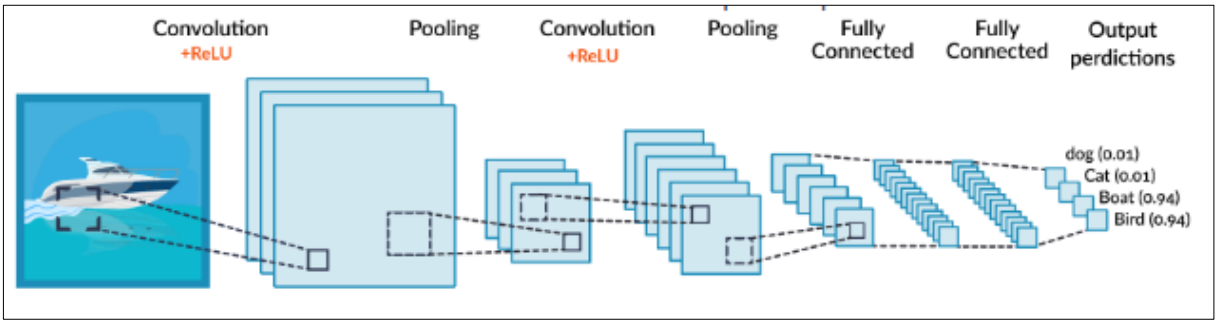

Figure 19:Illustration des étapes d'un réseau de neurone convolutif [38]

- **La couche de convolution :**

La couche de convolution est la composante clé des réseaux de neurones convolutifs, son but est d'extraire un ensemble des caractéristiques (features) des images reçues en entrée, elle permet de préserver la relation entre les différentes parties de l'image. Pour cela, on réalise un filtrage par convolution, le principe est de faire glisser un filtre (matrice de pixel de données d'entrée) sur l'image, et de calculer le produit de convolution entre le filtre et chaque portion de l'image balayée.

Considérons une image **5 x 5** dont les valeurs de pixel ne sont que de **0** et **1** la convolution de l'image **5 x 5** et de la matrice **3 x 3**(filtre) avec une foulée **1x1**(décalage de **1** pixel à chaque étape) peut être calculée comme indiqué dans **la figure20,**et que l'on nomme **« Image convoluée** ou **carte de caractéristiques**».

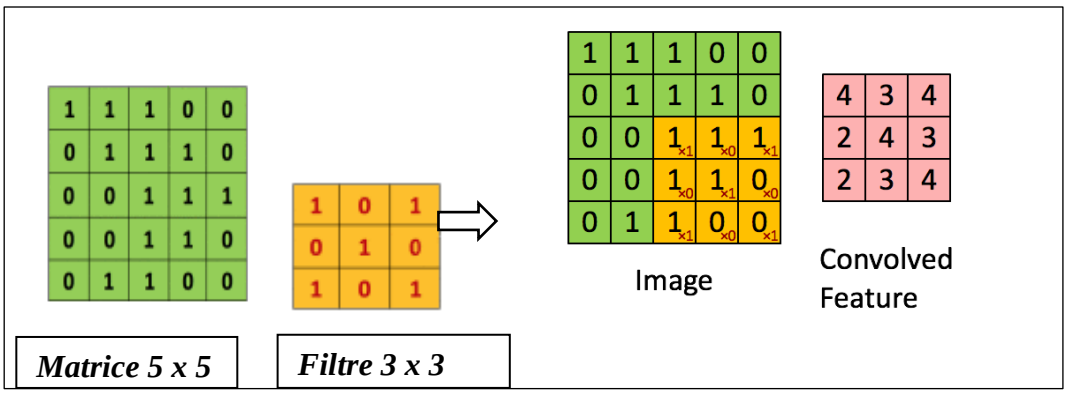

Figure 20:Opération de convolution [39]

- **La couche de mise en commun :**

Ce type de couche est souvent placé après la couches de convolution, elle reçoit en entrée plusieurs cartes de caractéristiques « **feature maps**» **(** matrice résultante de l'opération de convolution***)***, et applique à chacune d'entre elles l'opération de mise en commun (pooling) qui consiste à réduire la taille des images en sélectionnant les valeurs maximales, moyennes ou somme à l'intérieur de ces pixels, tout en préservant leurs caractéristiques importantes. Pour cela, on découpe l'image en cellules régulière. En pratique, on utilise souvent des cellules carrées de petite taille pour ne pas perdre trop d'informations. **Max Pooling**, l'une des techniques de pooling les plus courantes

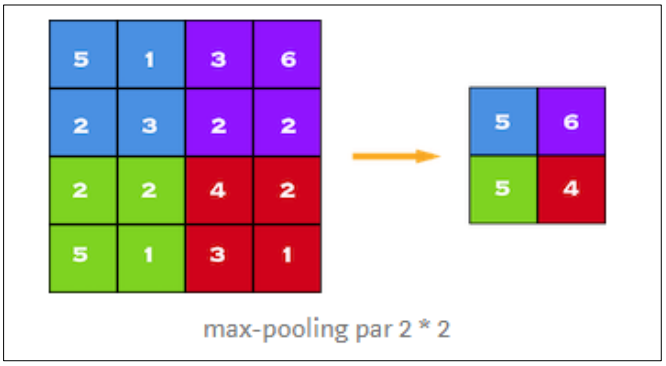

Figure 21:Opération de mise en communs (Pooling)[40]

- **La couche ReLU :**

ReLU (unité linéaire rectifiée) est un type de fonction d'activation. Mathématiquement, elle est définie comme **y = max (0, x)** Pour améliorer l'efficacité du traitement en intercalant entre les couches de traitement la couche de correction ReLU remplace donc toutes les valeurs négatives reçues en entrées par des zéros.

- **La couche entièrement connectée :**

La couche entièrement connectée **(**En anglais **fully-connected**) constitue la dernière couche du réseau de neurones, permet de classifier l'image en entrée du réseau : elle renvoie un vecteur de taille **N**, où **N** est le nombre de classes dans le problème de classification d'images. Chaque élément du vecteur indique la probabilité pour l'image en entrée d'appartenir à une classe.

1.5.8. Les domaines et exemple d'applications des réseaux de neurones :

Aujourd'hui les réseaux de neurones sont utilisés dans divers domaines tels que :

- **Médecine** : (Diagnostics médicaux).
- **Finance** : (la prévision des marchés financiers, détection de fraude).
- **Traitement d'image** : (la reconnaissance de caractères, reconnaissance de formes).
- **Traitement du signal** :(Traitement de la parole).
- Classification d'espèces animales étant donnée une analyse ADN.

1.5.9. Les avantages des réseaux de neurones :

- Efficacité pour résoudre des problèmes de haute dimensionnalité ou avec des relations complexe entre variables.
- La sortie peut être une valeur discrète, continue ou un vecteur de plusieurs valeurs discrètes ou continues.

1.5.10. Les inconvénients des réseaux de neurones :

- Apprentissage lent.
- Théoriquement complexe
- Difficile à mettre en œuvre, nécessitant un réglage fin et intensif en calcul.
- Le choix de valeurs initiales

Conclusion :

Nous avons consacré ce chapitre à l'étude de certains algorithmes de classification supervisée, la classification bayésienne, les machines à vecteurs de support, l'algorithme KNN, certains algorithmes dédiés à la génération des arbres de décision, et une structure un peu plus complexe qui est les réseaux de neurones.

Malgré l'aboutissement des algorithmes d'arbres de décision à la même structure de données qui est un arbre (binaire ou n-aire), nous avons pu constater la différence entre ces derniers en ce qui est des mesures de segmentation utilisées, le types de données traitées, l'élagage et la gestion des valeurs manquantes et continues.

Les réseaux de neurones présentent plusieurs types, dans ce chapitre deux types sont présenté, ANN et CNN, le premier ayant une structure simple, et le dernier ayant une structure plus sophistiquée.

Le prochain chapitre sera consacré à la partie pratique de notre mémoire. Nous allons présenter notre application son architecture, expliquer son fonctionnement ainsi la liste des algorithmes implémentés.

Chapitre3:

Description de l'application

Introduction :

Après avoir fait le tour d'horizon sur les techniques de classification supervisée, nous consacrons ce troisième chapitre à l'aspect pratique de notre travail.

L'objectif de notre travail est de développer une application qui permet d'exécuter et évaluer les techniques de classification supervisée : ID3, CART, C4.5, KNN, ANN et CNN.

Nous commençons d'abord par présenter l'architecture générale de l'application, ensuite nous détaillerons le rôle de chacun des modules et sous modules la composant, enfin un aperçu sur l'ensemble des interfaces et un guide d'utilisation sur le fonctionnement générale de l'application.

L'application porte le nom **M&E Classification**, où **M** correspond à "Metrics", **E** correspond à" Evaluation" et le mot "Classification" fait référence à la classification supervisée.

1. Concepts de base :

Pour maîtriser l'apprentissage supervisé, il faut absolument comprendre les deux concepts de base suivant :

- **Un Jeu de données :**

(Dataset en anglais) c'est une collection de données connexes, associées à un type spécifique d'informations, organisé en **structure de données** [16](tableau à deux dimensions), réparti en colonnes et en lignes, où : Les colonnes (**Xi)** représentent les attributs **(features)**, les lignes représentent les observations **(Enregistrements)** et **Y** signifie la classe **(target)** qui est la valeur que l'on cherche à prédire.

Le **tableau 1** montre un exemple d'un jeu de données de deux enregistrements et deux attributs qui décrivent le **" Nom» et** la **" Moyenne "** d'un étudiant où la classe à prédire est **" Admis "** ou **"Ajourné "**.

[16] **Structure de données** : Est un format spécial destiné à organiser, traiter, extraire et stocker des données.

Les attributs (Features) La classe (target)

Observations /attributs	X1=Nom	X2=Moyenne	Y=Décision
1	Asma	10	Admise
2	Ali	9	**Ajourné**

Tableau1 : Exemple d'un jeu de données

- Le modèle : **C'est une** implémentation d'un algorithme d'apprentissage automatique sur lequel est entraîné un jeu de données, avec lesquels il sera en mesure d'apprendre, raisonner et effectuer des prédictions, autrement dit lui faire apprendre la relation qui relie **x** (attributs) à **y**(classe), en trouvant une fonction **f / f(x)=y**.

NB : le modèle est appelé classifieur afin d'éviter la confusion avec le modèle du MVC.

2. L'architecture de l'application :

En programmation informatique, les problèmes complexes sont souvent découpés en modules, le patron de conception vient alors pour faire un arrangement caractéristique de ces modules en constituant des couches interactives de manière à simplifier la gestion de chacune et une meilleure gestion de l'évolutivité. Il existe plusieurs types de patron de conception, pour réaliser notre application nous avons utilisé le patron de conception appelé : le modèle **MVC**.

Le MVC est un patron de conception (design pattern) qui propose une solution générale au problème de la structuration d'une application. L'objectif global du MVC est de séparer les aspects traitement, données et affichage, et de définir les interactions entre ces trois aspects, cet objectif est réalisé en en divisant le code en trois parties distinctes qui sont : **Modèle**, **Vue** et **Contrôleur** ; nous allons expliquer chacune de ces parties dans ce qui suit mais avant nous allons donner un petit aperçu de notre architecture MVC sur laquelle se base notre application (voir **figure 1**) :

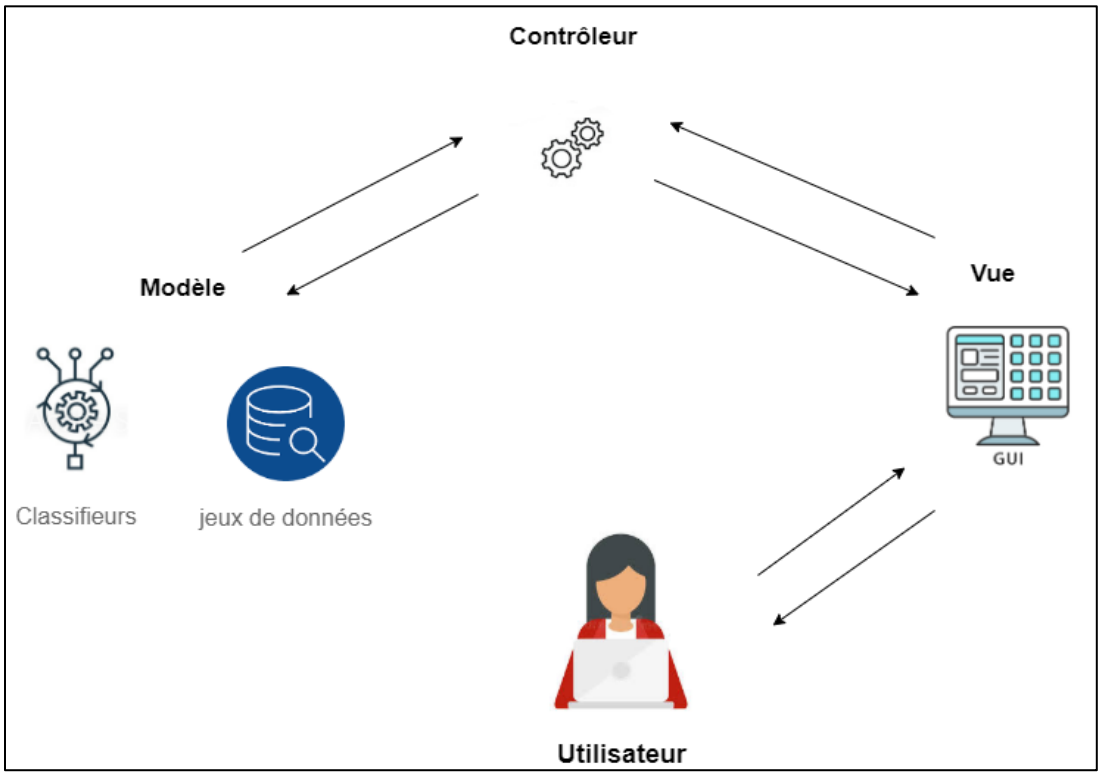

Figure1: Architecture de l'application

La **figure 1** montre l'architecture de l'application, les différentes connexions entre les trois parties : Modèle, Vue, Contrôleur, le rôle de chacune est expliqué ci-après :

- **Le modèle** : c'est la partie consacrée aux deux aspects, données et traitement. Dans notre application elle regroupe un ensemble de fonctions qui effectuent les opérations (traitements) sur l'ensemble des jeux de données et sur l'ensemble des classifieurs.

  Les traitements effectués retournent des résultats (données) qui seront transmis au contrôleur, nous verrons plus en détails cette partie (**voir la section 3. L'architecture de la partie modèle)**.

- **La vue :** Cette partie concerne l'aspect affichage, elle correspond à l'interface avec laquelle l'utilisateur interagit, elle n'effectue aucun traitement**,** elle se contente d'afficher les résultats des traitements effectués par le modèle. Elle communique avec le contrôleur pour récupérer les résultats qui répondent à la requête de l'utilisateur.

- **Le contrôleur :** Le contrôleur joue le rôle d'un chef d'orchestre, il dirige le trafic entre la vue et le modèle (intermédiaire), il récupère les données depuis le modèle et les transmets aux vues, de plus Il décide l'interface de à afficher, qui réponds à la requête de l'utilisateur.

2.1. Architecture interne de la partie modèle du MVC :

La partie modèle est organisée en modules interconnectés, chaque module est conçu pour répondre à une fonctionnalité précise comme le montre **la figure2**

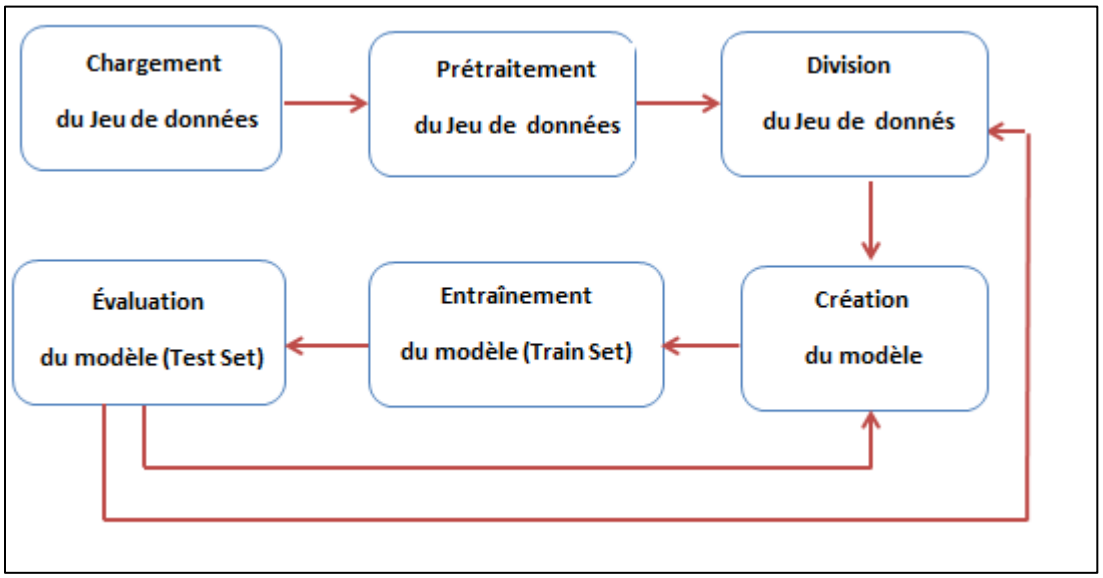

Figure2 : L'architecture de la partie modèle du MVC

La **figure 2** illustre le lien entre les modules :

- Chargement du jeu de données,
- Prétraitement du jeu de données.
- Division du jeu de données
- Création du modèle (classifieur).
- Entraînement du modèle (classifieur)
- Évaluation du modèle.

Avant de détailler chaque module, nous allons présenter au préalable les bibliothèques que nous avons utilisé pour réaliser les différentes tâches se rapportant à chaque module, présentées ci-après :

- **Pandas :** En programmation informatique, pandas est une bibliothèque de logiciels écrite pour le langage de programmation Python pour la manipulation et l'analyse de données.
- **Scikit-learn :**Connu aussi sous le nom (Sklearn) ,est une bibliothèque d' apprentissage automatique de logiciels gratuits pour le langage de programmation Python . Il propose plusieurs types de classification , la régression et le clustering , est conçu pour interagir avec les bibliothèques numériques à savoir NumPy .

- **Matplotlib :** Est une bibliothèque de traçage , elle crée des visualisations statiques animées et interactives en langage de programmation Python en tant que composant de **NumPy** . Il fournit une API orientée objet pour incorporer des tracés dans des applications à l'aide de boîtes à outils GUI à usage général telles que Tkinter .

- **Chefboost :** C'est un Framework qui permis la construction des arbres de décision y compris (ID3, C4.5, CART, CHAID) ainsi les arbres de régression sous forme de règles de décisions stockés dans le répertoire **« outputs/rules»** ajouter à cela l'implémentation en python des forêts aléatoire (dérivés d'arbre de décision ou l'ensemble de données serait séparé en sous-ensembles la décision sera alors prise par la plus grand nombre de résultats de ces derniers)**.** La commande dédiée à son installation est :

  $$**pip install chefboost** .

- **TensorFlow :** Est le premier Framework d'apprentissage en profondeur open source créé et développé par Google.

- **Keras :** Est une API (**A**pplication **P**rogramming **I**nterface) de réseaux de neurones de haut niveau, écrite en python interfaçable avec TenserFlow, elle a été développée pour permettre des expérimentations rapides, avoir des résultats avec le fiable délai.

- **Graphviz :** Diminutif de Graph Visualisation, c'est une bibliothèque qui manipule des graphes définis à l'aide de scripts suivant le langage DOT.

Après avoir vu l'ensemble des bibliothèques utilisées pour le codage des modules cités dans la figure **2** nous allons présenter ces derniers ci-après :

2.1.1. Le module chargement de jeu de données :

C'est le module qui s'occupe du chargement des jeux de données, importés depuis :

➢ Des bibliothèques de machine Learning, dans notre cas nous avons utilisé :

- **Scikit learn:** Via le module **« sklearn.datasets»** (cas de **Breast cancer** et **Iris** voir chapitre 4 section 1).
- **Keras:** Via le module **«tensorflow.keras.datasets»** (cas de **Mnist** voir chapitre 4 section1).

Ou bien :

- À partir des fichiers téléchargés au préalable sous forme .**csv** ou encore .xsls (cas de **Bank marketing** voir chapitre 4). Le code en Python qui permet de charger le fichier est : **pandas.read_(csv/xsls)(** "chemin du fichier ").

2.1.2. Le module prétraitement de jeux de données :

Les algorithmes de classification apprennent à partir des données qui leur sont fournies, par conséquent si ces données sont de mauvaises qualité (erronées, incomplètes…) alors l'algorithme qui en résulte sera aussi mauvais puisqu'il est censé refléter ce qu'il voit dans les données. C'est pour cette raison qu'il est impératif de bien préparer les données avant le passage à l'apprentissage de l'algorithme, c'est l'objectif de ce module qui traite les jeux de données chargés (voir la section **1**) , en passant par les quatre étapes , analyse et visualisation des jeux de données, vérification des valeurs manquantes, normalisation des jeux de données, traitement des valeurs catégorielles, expliquées ci-après :

2.1.2.1. Analyse et visualisation des jeux de données :

Dans un premier temps nous avons analysé les données chargées, vérifié leur type, les attributs, les classes et le nombre d'enregistrements par classe, en utilisant un certain nombre de fonctions :

- **describe()** :Elle génère une description statistique de jeu de données pour chaque colonne (la moyenne, le minimum des valeurs …).
- **value_counts() :**Cette fonction nous renvoi le nombre de lignes (enregistrements)pour chaque valeur de classe.
- **dtypes()** :Elle renvoie le jeu de données avec le type de données de chaque colonne.

Nous avons aussi visualisé nos données dans des nuages de point en utilisant la bibliothèque **matplotlib.**

2.1.2.2. Vérification des valeurs manquantes : Dans cette étape nous avons vérifié la présence des valeurs manquantes dans les jeux de données en utilisant la fonction **isnull()**.

2.1.2.3. Normalisation des jeux de données :

Il est aussi important de **normaliser** nos données, c'est-à-dire les mettre sur une même échelle pour rendre l'apprentissage de la machine plus rapide et aussi plus efficace. Pour ce faire nous avons utilisé la méthode de normalisation **MinMaxScaler** (voir la section 1.1.4.Normalisation MinMax du chapitre 1) que nous avons importé du module **sklearn.preprocessing** de la bibliothèque sklearn.

2.1.2.4. Traitement des valeurs catégorielles :

Il consiste à vérifier la présence des valeurs catégorielles dans nos jeux de données et en faire l'encodage, nous avons utilisé à cet effet le module **LabelEncoder** de la classe **sklearn.preprocessing** de la bibliothèque sklearn.

2.1.3. Le module division de jeu de données :

Un bon classifieur en machine Learning, est un classifieur qui généralise, autrement dit, il a la capacité de faire des prédictions non seulement sur les données utilisées pour le construire, mais surtout sur de nouvelles données. Pour obtenir ces deux types de données, la procédure consiste à effectuer un fractionnement en deux sous-ensembles sur le jeu de données :

- **Ensemble de données d'apprentissage :** Communément appelé **train set**, utilisé pour entraîner le classifieur, l'ensemble des observations (enregistrements) est appelé **x_train** et le vecteur de classes (target) correspondantes aux observations (enregistrements) est décrit par **y_train.**

- **Ensemble de données test :** Communément appelé **test_set**, l'ensemble des observations (enregistrements) est appelé **x_test** et le vecteur de classes correspondantes aux observations est appelé **y_test.** Ce sont les données qui n'ont pas servi à l'entraînement, utilisées comme une entrée du modèle pour tester sa performance en comparant les prédictions données par le classifieur **(y_pred**) et les valeurs attendues **(y_test).**

Le module "division de jeu de données" s'occupe de ce fractionnement en ensemble de test et ensemble d'apprentissage**,** Dans notre code **80%** du jeu de données est consacré pour l'ensemble d'apprentissage quant aux **20%** restantes elles forment l'ensemble de test, pour ce faire nous avons utilisé la fonction **train_test_split** de Sklearn .

2.1.4. Le module création du modèle :

Dans ce module nous avons créé six modèles (classifieurs) qui implémente les algorithmes de classification supervisée suivant : L'algorithme KNN, CART, ID3, C4.5, Les réseaux de neurones (ANN), Les réseaux de neurone convolutif (CNN).

2.1.4.1. L'algorithme KNN :

Nous avons implémenté l'algorithme KNN à l'aide du modèle **KNeighborsClassifier** présent dans la bibliothèque sklearn, accompagné d'un ensemble de paramètres à spécifier afin d'obtenir le modèle le plus optimal possible. Ces paramètres sont les suivants :

- **n_neighbors :** Le nombre de voisins pris par le modèle c'est le paramètre (**k**) de l'algorithme, par défaut **n_neighbors=5**.

- **metric:** La métrique de distance à utiliser (par défaut : **Minkowski**), la liste des métriques disponible est consultable dans la documentation de **"sklearn.neighbors.DistanceMetric"**.

2.1.4.2. L'algorithme CART :

Nous avons implémenté cet algorithme par le modèle **DecisionTreeClassifier** importé de la bibliothèque sklearn, à noter ce modèle utilise une version optimisée de CART qui ne prend pas en charge les variables catégorielles, il fournit un ensemble de paramètres configurables pour la création d'un modèle optimal, par exemple le paramètre **max_depth** pour la profondeur de l'arbre**, criterion** (**gini** ou entropy) pour choisir le critère de segmentation. Le modèle génère un arbre et des règles de décisions comme résultat.

2.1.4.3. L'algorithme ID3 :

Nous avons implémenté l'algorithme ID3 par le modèle **ID3Estimator** qui est un package compatible avec la bibliothèque sklearn installable via la commande **« $$ pip install decision-tree-id3 »**. Comme résultat ce modèle génère l'arbre de décision associé sous format **.dot**, l'arbre peut être lu sous forme **pdf** ou **png** avec l'outil **GraphViz**.

ID3Estimator compte certains paramètres pour contrôler la croissance de l'arbre comme pour **max_depth** (Configuration de la profondeur maximale de l'arbre).

2.1.4.4. L'algorithme C4.5 :

Cet algorithme a été implémenté à l'aide du Framework **Chefboost**. Nous avons eu comme résultat les règles de décision (**if _else**) qui ont été mise dans le chemin **« outputs/rules »** .

.

2.1.4.5. Le réseau de neurones artificiels (ANN) :

Le réseau de neurones artificiels a été implémenté par le modèle **Sequential** importé du module **tensorflow.keras.models.** Ce modèle représente un réseau de neurone simple.Nous avons créé deux type du réseau de neurones un pour les données à une dimension et l'autre pour les données multidimensionnel (images) présentés ci-après :

- **Cas de jeux de données à une dimension :**

  Le modèle du réseau neuronale créé se compose d'une couche d'entrée où l'entrée est de taille **n** (**n** représente le nombre d'attribut du jeu de données), une seule couche cachée **(Dense)** de **128** neurones et une couche de sortie à **m** neurones où **m** représente le nombre de classes du jeu de données, avec une fonction d'activation **sigmoid**[17] pour les jeux de données binaires (cas de **breast cancer** et **bank marketing** définit dans le **chapitre 4 section 1.1Breast Cancer,1.2 Bank Marketing**) et **softmax**[18] pour les jeux de données multi classes (cas de Iris, voir **chapitre 4 section1.3 Iris).** La **figure 3** montre l'architecture du modèle créé.

[17] **sigmoid** : Une fonction d'activation utilisée pour les jeux de données binaires, elle renvoie une probabilité pour chaque sortie comprise entre de **0 et 1.**

[18] **Softmax :** Une fonction d'activation utilisée pour les jeux de données multi classes, elle renvoie une probabilité pour chaque classe comprise entre **0** et **1** sachant que la somme de toutes les probabilités est égale à **1**.

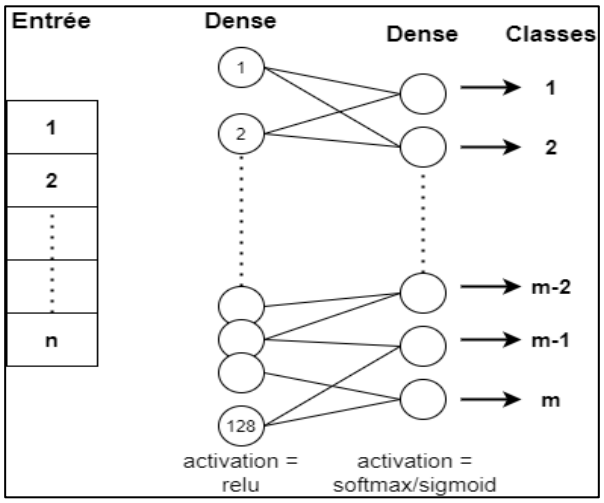

Figure 3: Architecture du modèle ANN pour les jeux de données à une dimension

- **Cas de jeux de données à deux dimensions (images) :**

Le modèle créé est composé d'une couche d'entrée, **une couche Flatten**[19], une couche cachée (Dense) de **128** neurones et une couche de sortie à dix neurones où chaque neurone correspond avec une fonction d'activation softmax. L'image en entrée est de taille **(28x28). La figure 4** représente l'architecture du modèle.

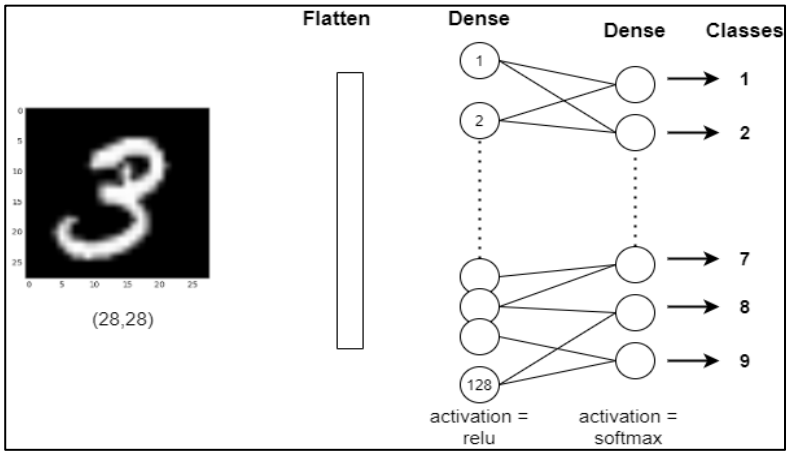

Figure4:Architecture du modèle ANN pour le cas des données multidimensionnels.

[19] **La couche Flatten :** C'est une couche qui transforme les images en entrée en un vecteur d'une dimension (aplatissement de l'image).

2.1.4.6. Le réseau de neurone convolutif (CNN) :

Le réseau de neurone convolutif a été implémenté par le modèle **Sequential** de l'API Keras. Dans ce qui suit nous expliquons les modèles crées pour les deux cas, jeux de données à une dimension et jeux de données à deux dimensions :

- **Cas de jeux de données à une dimension :**

Le modèle est organisé en six couches, une couche d'entrée, deux couches de convolution à une dimension (**Conv1D),** une couche Flatten et une couche de correction **relu** de **64** neurones et une couche de sortie (couche entièrement connecté), avec la fonction d'activation sigmoid pour les jeux de données binaire et softmax pour les jeux de données multi classes **(voir** la **figure5).**

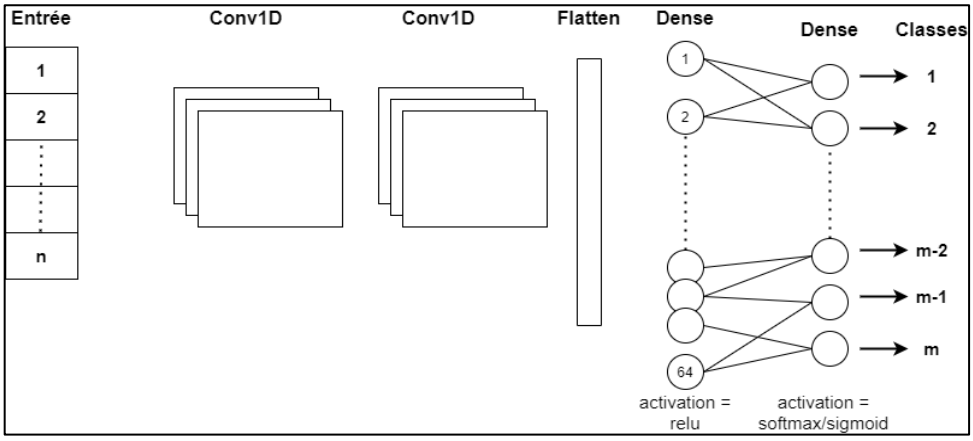

Figure5: Architecture du modèle CNN cas de jeux de données à une dimension

- **Cas de jeux de données à deux dimensions (images) :**

Le modèle est formé d'une couche d'entrée, une couche de convolution **Conv2D** et une couche de mise en communs **MaxPooling2D,** une couche Flatten et une couche de correction **relu** de **200** neurones et enfin une couche entièrement connectée, Pour ce modèle nous avons utilisé la fonction d'activation softmax.

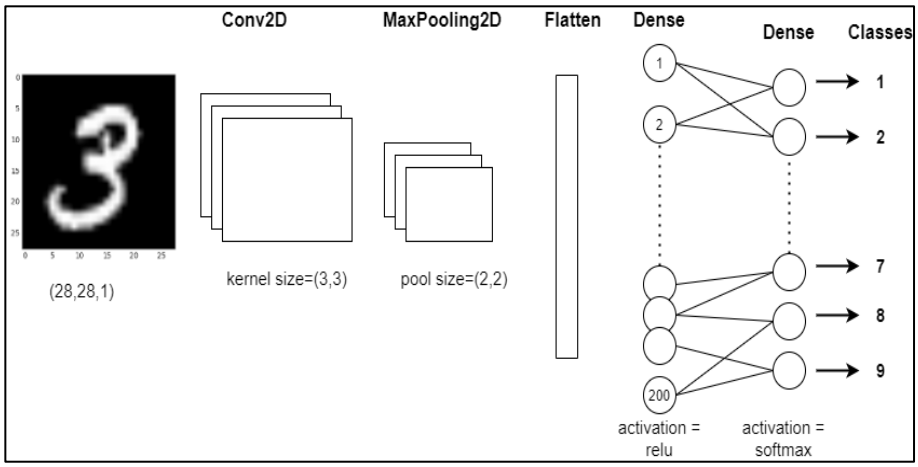

Figure 6: Architecture du modèle CNN cas de jeux de données à deux dimensions

2.1.5. Le module entraînement du modèle :

Dans cette partie nous avons entraîné les modèles créés dans la phase précédente sur l'ensemble d'apprentissage du jeu de données **Tain Set(x_tain,y_train)** en utilisant la fonction **fit(x_train,y_train).**

2.1.6. Le module évaluation du modèle :

Ce module permet d'évaluer les modèles créés en utilisant des métriques d'évaluation **(voir le chapitre 4 section 3.1 Métriques d'évaluation)** qui sont importées du module **metrics** de la bibliothèque sklearn. L'évaluation des modèles est faite sur l'ensemble de test des jeux de données **Test Set(x_test,y_test**) juste après l'étape d'entraînement. Les modèles prédisent la classe de chaque enregistrement de l'ensemble de test **(x_test)**, un vecteur de classes (**y_pred**) est retourné, l'évaluation des performances des modèles se fait en comparant le vecteur de classes retournées **(y_pred)** et celui de classes d'origine **(y_test)** de l'ensemble de test **(x_test).**

3. Le fonctionnement général de l'application :

Le lancement de l'application dirige l'utilisateur vers la première interface appelée **"accueil",** qui est une interface statique et à travers laquelle l'utilisateur peut voir le nom, le logo et l'objectif de l'application. Voir la **figure 7** :

Figure 7:la page d'accueil

- Le clic sur le bouton " **Suivant** " de la page d'accueil déclenche l'évènement de l'appel de la fonction **passerjeudonnees()** présente dans le contrôleur qui donne accès à la deuxième interface nommée **"Jeux de données",** voir **figure 8,** sur cette interface l'utilisateur peut choisir sur quatre listes un jeu de données, sa description sera ensuite affichée dans la partie droite de l'interface :

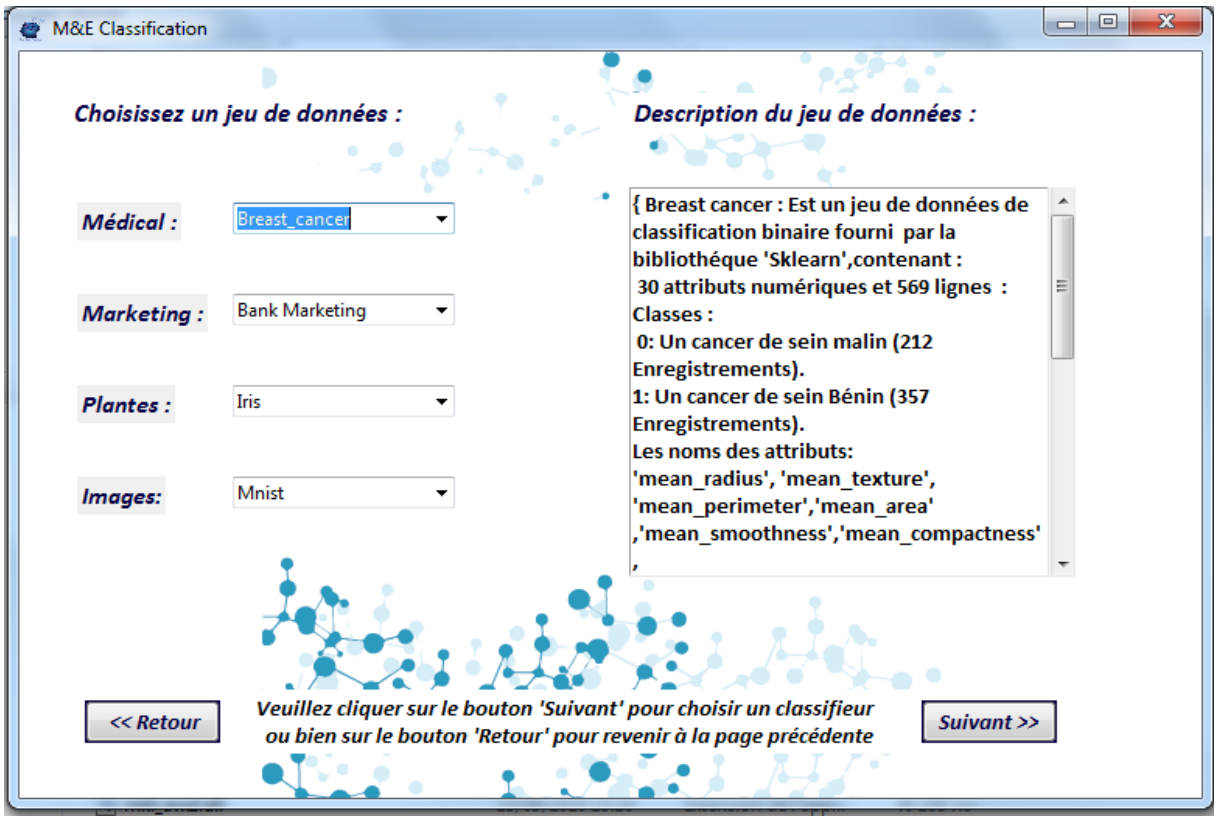

Figure 8:L'nterface jeu de données

- Le clic sur le bouton **"Suivant"** fait appel au contrôleur qui va exécuter la fonction **passerclassifieurs**() pour diriger l'utilisateur vers l'interface **"classifieur", voir figure 9** :

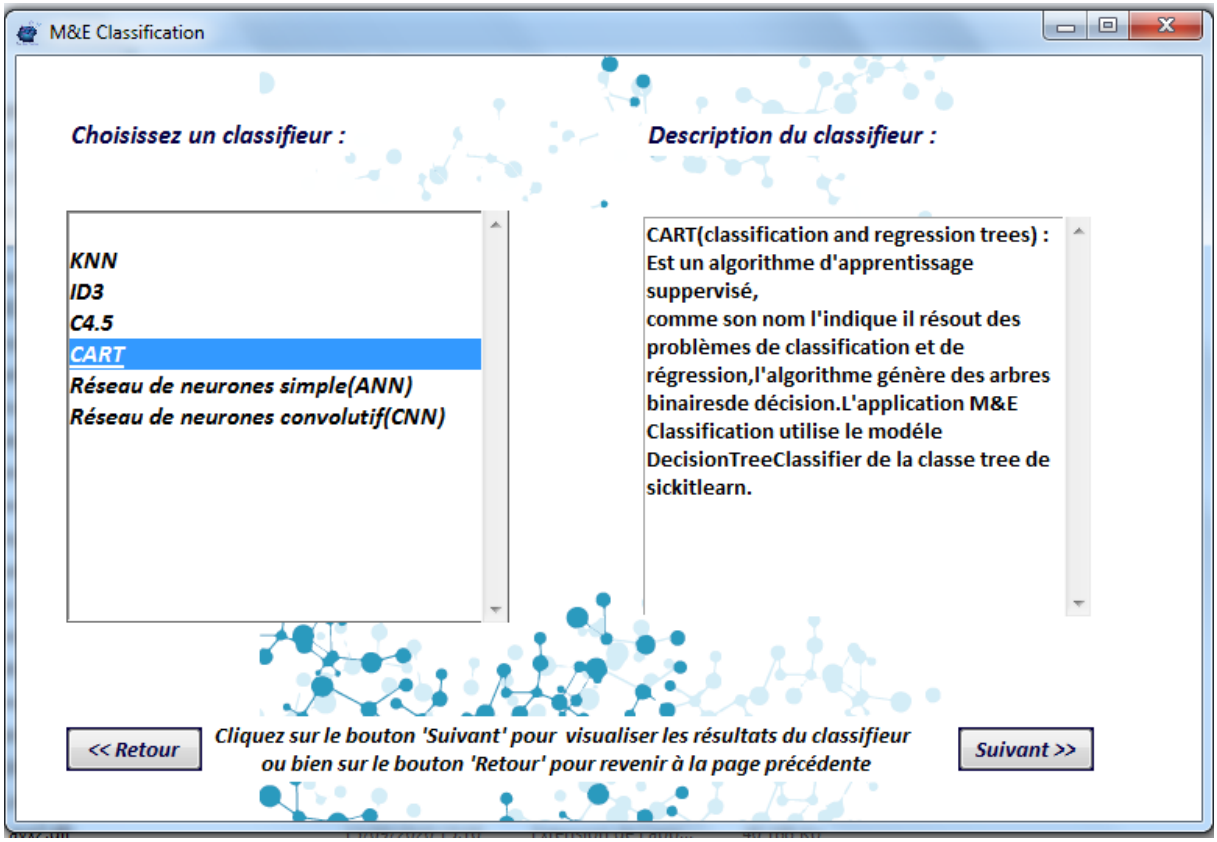

Figure 9:l'interface classifieur

- L'interface **"Classifieur"** donne la possibilité à l'utilisateur de choisir un classifieur parmi une liste de six algorithmes, si un classifieur est sélectionné, sa description est affichée dans la partie droite de la même interface si l'utilisateur.
- Le bouton **"Suivant"** de l'interface **"Classifieur"** donne accès à l'interface évaluation appelé par le contrôleur qui à son tour appelle la fonction. **passerevaluation(combovalue,classifieurvalue)** avec les deux paramètres : **combovalue** qui représente le nom du jeu de données choisie et **classifieurvalue** qui est la valeur de classifieur choisie par l'utilisateur
- La fonction **passerevaluation(combovalue,classifieurvalue)** appelle la fonction **evaluation_model(combovalue,classifieurvalue)** localisé dans la partie modèle de notre architecture MVC
- La fonction **evaluation_model(combovalue,classifieurvalue)** appelle les six modules : chargement, le prétraitement et la division du jeu de données, ensuite création, entrainement et évaluation du classifieur(voir la section 2.1Architecture interne de la partie modèle du MVC)

- A ce stade le contrôleur récupère les résultats de l'évaluation des modèles et la liste des schémas délivrés par chaque modèle, le contrôleur envoie ces résultats à l'interface évaluation là où elles seront affichées, voir **figure 10**.

Les affichages répondent à la requête de l'utilisateur exprimée par les deux valeurs : jeu de données sélectionné et le classifieur choisi par l'utilisateur.

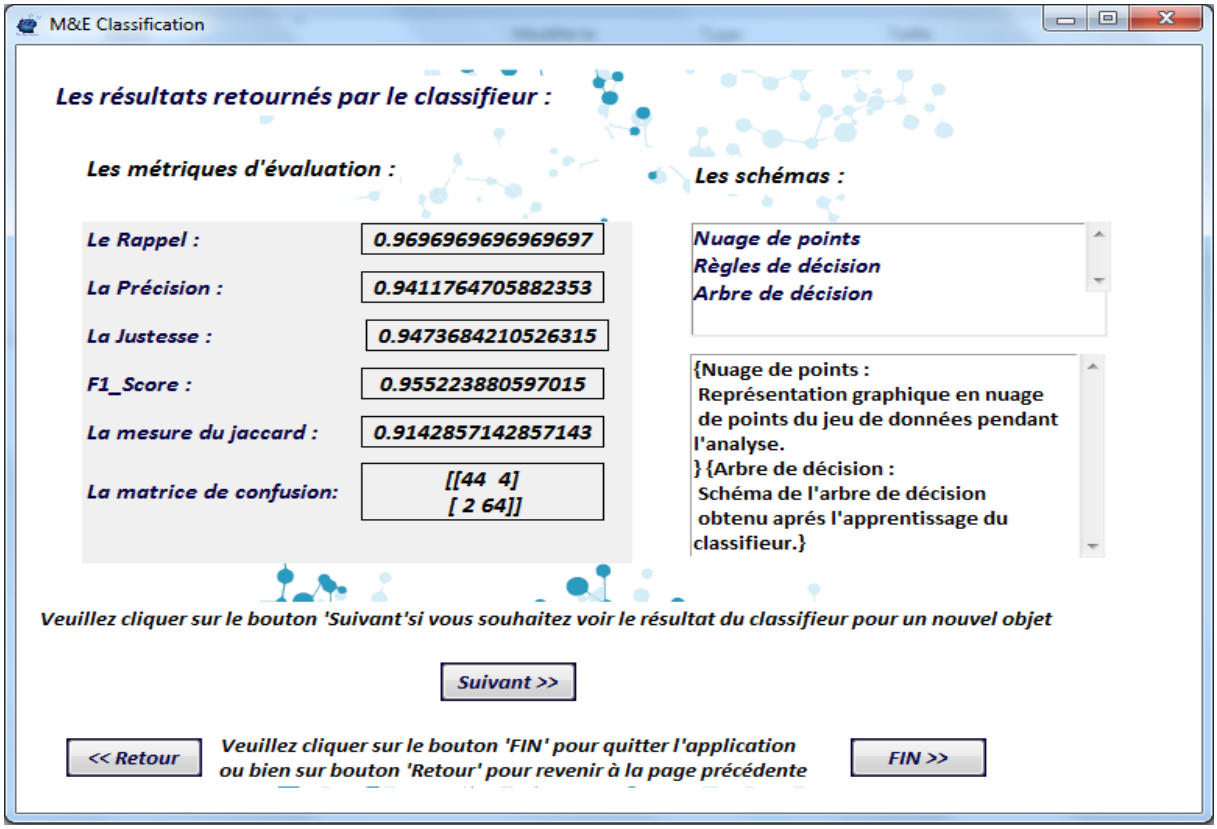

Figure10: l'interface évaluation

- Dans l'interface évaluation l'utilisateur pourra voir à la fois les résultats du classifieur et visualiser un schéma de son choix de la liste des schémas affiches dans le côté droit de l'interface évaluation voir **figure 10**.
- L'utilisateur pourra atteindre l'interface **"fin"** en cliquant sur le bouton **"fin",** voir la **figure 12.**

- Pour visualiser le résultat fourni par le classifieur pour un nouvel objet pris du **Test Set,** le bouton **"Suivant"** de l'interface évaluation appelle le contrôleur qui à son tour exécute la fonction **passernouvelobjet(combovalue,classifieurvalue)** chargée par l'affichage de l'interface **nouvelobjet**, voir **figure 11 .**

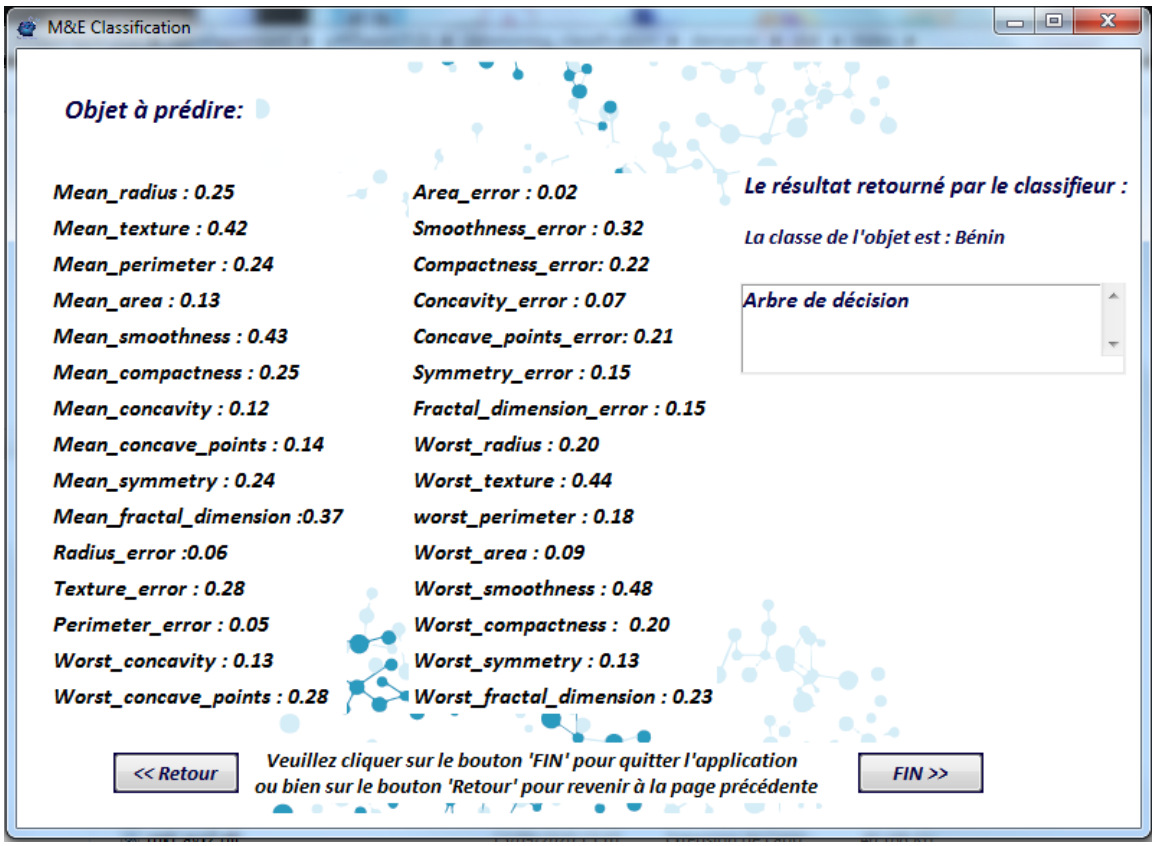

Figure11: l'interface nouvel objet

- L'interface **nouvelobjet** affiche l'enregistrement de l'objet à prédire et le résultat de prédiction retourné par le clasifieur pour ce même objet sur la partie droite de l'interface.
- Si l'utilisateur veut quitter l'application il pourra cliquer sur le bouton « **FIN »,** voir **figure 11** l'interface fin s'affichera le bouton **"Fermer"** de la même interface permet de fermer l'application. Voir **figure 12 .**

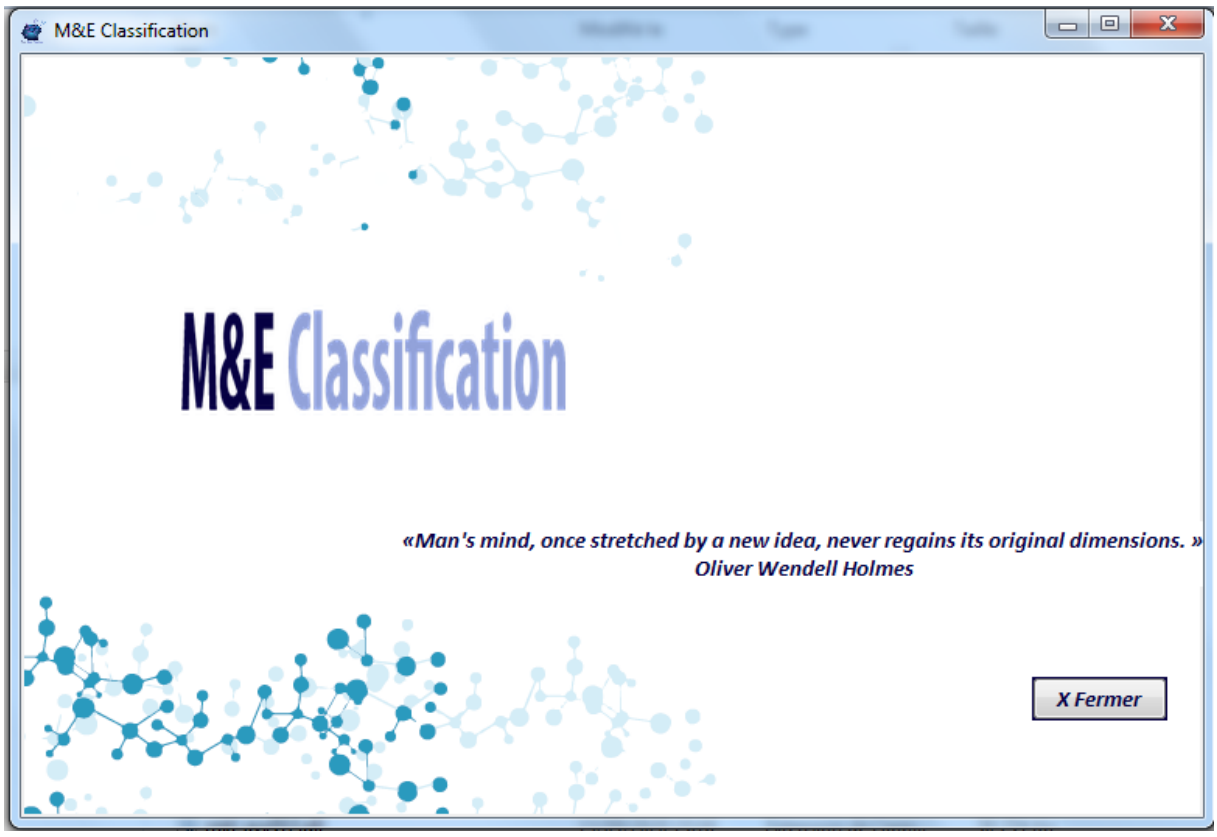

Figure12: l'interface fin

- **Remarque :** Dans chaque interface l'utilisateur pourra revenir à l'interface précédente par le bouton **"Retour"**.

4. Environnement et outils de développement :

Dans ce qui suit nous allons présenter l'environnement du développement ainsi que les outils que nous avons utilisés pour réaliser notre travail.

4.1. Environnement matériel :

- PC: Lenovo ThinkPad: Intel (R) Core(TM) i5-4300U CPU @ 1.90 GHz.
- PC: Lenovo: Intel (R) Core(TM) i5-5200U CPU @ 2.20 GHz.

4.2. Environnement logiciel :

- Windows 7 professionnel.
- Windows 10 professionnel.

4.3. Environnement de développement :

- **Anaconda:**

Anaconda est une distribution libre et open source des langages de programmation Python et R appliqué au développement d'applications dédiées à la science des données et à l'apprentissage automatique. Anaconda contient tous les outils et librairies dont vous avez besoin pour faire du Machine Learning : **Numpy**, **Matplotlib**, **Sklearn**, etc.

- **Spyder**:

Acronyme signifiant **S**cientific **PY**thon **D**evelopment **E**nvi**R**onment, c'est un IDE (**I**ntegrated **D**evelopment **E**nvironment) multiplateforme open source, entièrement écrit en python. Il intègre des bibliothèques d'usage scientifique comme : MATPLOTLIB, Numpy…etc

4.4. Langage de programmation :

- **Python:**

Un langage de programmation **open source**, répandu par sa facilité, permet une approche simple mais efficace de la programmation orientée objet, il est doté d'une gestion automatique de la mémoire par ramasse-miettes, d'un typage dynamique et d'un système de gestion d'exceptions. Grâce à ses bibliothèques et packages de data science, il est devenu le langage le plus populaire pour les algorithmes de machine learning, data science et le big data, mais ses cas d'utilisation s'étendent encore pour le scripting, l'automatisation, la création de services web ou de REST API, ou encore pour le méta programmation et pour la génération de code.

4.5. Les bibliothèques utilisées :

- **Tkinter :**(**T**ool **k**it **I**nterface), module présent dans python pour la création des interfaces graphiques.
- **Fpdf:** C'une bibliothèque qui permis la génération des fichiers pdfs sous Python.
- **Pillow:** Python Imaginig Library est une bibliothèque de traitement d'image, elle est conçue d'une manière à offrir l'accès rapide aux données contenus dans une image elle offre un support pour différents formats fichiers tels que PNG et JPEG. Elle dispose d'une capacité de traitement d'images puissantes. En ce qui est affichage des images sous windows les deux modules **PIL.ImageTk** ou **PIL.ImageWin** qui sont appelés.
- **Auto-py-to-exe:** Elle permet la conversion du fichier python « **.py** » vers un fichier exécutable « .**exe** »

- **Pyinstaller**: C'est une bibliothèque qui regroupe une application Python en toutes ces dépendances dans un seul package**.**
- **Opencv** :(**O**pen **C**omputer **V**ision) est une bibliothèque graphique spécialisé dans le traitement d'images en temps réel**.**
- **Seaborn:** Est une bibliothèque de visualisation de donnes en Python basé sur matplotlib. Il fournit une interface de haut niveau pour dessiner des graphiques statistiques attrayants et informatif.
- **Pathlib :** Traite des tâches liées aux chemins, telles que la construction de nouveaux chemins à partir des noms de fichiers et d'autres chemins, la vérification de diverses propriétés des chemins et la création de fichiers et de dossiers sur des chemins spécifiques.
- **Webbrowser:** C'est une bibliothèque qui fournit une interface de haut niveau permettant d'afficher des documents Web aux utilisateurs via la **open()** fonction.
- **NumPy :** Il fournit une API orientée objet pour incorporer des tracés dans des applications à l'aide de boîtes à outils GUI à usage général telles que Tkinter .
- **GIT :** Est un logiciel de gestion de versions décentralisé pour suivre les modifications du code source pendant le développement des logiciels, conçu pour coordonner le travail entre les programmeurs.
- **Adobe Illustrator :** Un logiciel de création graphique vectorielle.
- **Draw.io :** Service en ligne permettant de dessiner les diagrammes.
- **MockFlow :** Service en ligne de création des maquettes.

Conclusion :

Nous avons présenté dans ce chapitre l'architecture MVC de l'application avant de détailler la structure et le fonctionnement des différents modules et présenter la mise en œuvre de nos algorithmes de classification supervisée.

Nous avons décrit par la suite le fonctionnement de l'application accompagné des captures de ses interfaces, nous avons clôturé par la présentation de l'environnement de développement et des outils que nous avons utilisé pour l'implémentation de notre application.

Comme l'indique le thème de notre mémoire "implémentation et évaluation des algorithmes de classification supervisée", après avoir implémenté les algorithmes, nous allons dans le chapitre suivant présenter l'évaluation de ces algorithmes en utilisant un ensemble de métriques.

Chapitre4:

Tests et résultats

Introduction :

Nous avons vu dans le premier chapitre qu'il existe plusieurs techniques en data mining, nous nous sommes intéressées aux techniques de classification supervisée dans le second chapitre, nous avons implémenté quelques-unes au cours du troisième.

Pour tester la qualité des algorithmes implémentés, au cours de ce chapitre nous avons présenté un ensemble de métriques et un ensemble de jeux de données que nous avons utilisé, sur lesquels nous avons effectué des tests afin d'évaluer et comparer ces derniers.

En premier lieu nous avons présenté les jeux de utilisés pour tester les algorithmes implémentés, ensuite nous avons aborder les tests effectués sur ces algorithmes dans le but de déterminer l'algorithme optimal pour chaque jeu de données.

Á la fin du ce chapitre nous avons présenté les différentes métriques d'évaluation mesurées pour évaluer chaque algorithme, les résultats de chaque métrique obtenus chaque jeu de données et discuter ces résultats.

1. Présentation des jeux de données :

L'application "**M&E Classification**" que nous avons conçu organise les jeux de données sous quatre catégories, à savoir : médicale, marketing, plantes et enfin images.

Pour tester et évaluer les algorithmes implémentés, nous avons utilisé quatre jeux de données, un pour chaque catégorie, ils s'agissent de Breast Cancer, Bank Marketing, Iris et Mnist, que nous que nous détaillerons dans ce qui suit :

1.1. Breast Cancer (UCI ML Breast Cancer Wisconsin Diagnostic) :

C'est un ensemble de données sur le diagnostic du cancer de sein bénin ou malin, crée par Dr William H. Wolberg, médecin à l'hôpital de l'Université du Wisconsin à Madison aux États-Unis en novembre 1995. Importé depuis la bibliothèque de scikit-learn [26] sous format **csv**[20].

L'ensemble de données est binaire avec **569** lignes, où chaque objet est décrit par **30** attributs numériques.

[20] **CSV** : sigle de « comma-separated values », est un format texte ouvert représentant des données tabulaires sous forme de valeurs séparées par des virgules .

Les valeurs d'attributs sont calculées à partir d'une image numérisée d'un aspirât à l'aiguille fine **BAF**[21] d'une masse mammaire. Ils décrivent les caractéristiques des noyaux cellulaires présents dans l'image. Dix caractéristiques sont calculées pour chaque noyau cellulaire qui sont énumérés dans le **tableau 1** :

Caractéristique	**Signification**
Radius	Moyenne des distances du centre aux points sur le périmètre
Texture	Écart type des valeurs de l'échelle de gris
Perimeter	Périmètre
Area	Surface
Smoothness	Variation locale des longueurs de rayon
Compactness	Périmètre ^ 2 / surface – 1
Concavity	Gravité des parties concaves du contour
Concave_points	Nombre de parties concaves du contour
Symmetry	Symétrie
Fractal_dimension	"approximation du trait de côte" – 1

Tableau1:Caractéristiques des attributs du jeu de données Breast Cancer

De l'ensemble des 10 caractéristiques du **tableau 1** sont calculées : la moyenne, l'erreur standard et la "pire" ou la plus grande (moyenne des trois valeurs les plus mauvaises/ les plus élevées) de ces caractéristiques pour chaque image, ce qui donne **30** attributs (voir ANNEXE B)

1.2. Bank Marketing :

C'est un jeu de données binaire liées aux compagnes de marketing d'une institution bancaire portugaise, la compagne markéting est basée sur des appels téléphoniques avec des clients, à la fin de cet appel est décidé si le client souscrira un dépôt bancaire à terme (classe **oui**) ou pas (classe **non**), Il est disponible dans le lien suivant **[27] .**

Le jeu de données décrit 41 188 observations, chacune est définie sur 20 attributs, 10 catégoriels définis dans le **tableau 2**, et 10 numériques définis dans le **tableau 3**.

[21] **BAF** : acronyme signifiant La biopsie à l'aiguille fine, c'est une procédure chirurgicale de diagnostic.

Attribut	Signification	Valeurs
Job	Type d'emploi	admin. blue-collar, entrepreneur, housemaid, management, retired, selfemployed, services, student, technician, unemployed, unknown
Éducation	Niveau éducatif	basic.4y, basic.6y, basic.9y, high.school, illiterate, professional.course,university. degree, unknown
Marital	État matrimonial	divorced, married, single, unknown
Default	Le crédit est-il par défaut	Yes, no
Housing	Un prêt au logement	No, yes, unknown
Loan	Prêt personnel	No, Yes, Uknown
Contact	Type de communication	cellular, telephone
month	Mois du dernier contact	Jan, feb,……,dec
Day of week	Jour du dernier contact	Lun, tue, ….,fri
poutcome	Résultat de la campagne marketing précédente	failure, nonexistent, success

Tableau2:Attributs catégoriels du Bank Marketing

Attribut	Signification
Age	L'âge du client
Duration	Durée du dernier contact en secondes
Campaign	Nombre de contacts effectués pendant cette campagne et pour ce client
Pdays	Nombre de jours qui se sont écoulés après le dernier contact du client d'une campagne précédente
previous	Nombre de contacts réalisés avant cette campagne et pour ce client
emp.var.rate	Taux de variation de l'emploi, indicateur trimestriel
cons.price.idx	Indice des prix à la consommation, indicateur mensuel
cons.conf.idx	Indice de confiance des consommateurs, indicateur mensuel
Euribor3m	Taux euribor 3 mois
Nr_employed	Nombre de salariés

Tableau3:attributs numériques du Bank Marketing

1.3. Iris :

C'est un jeu de données multi classes connu aussi sous le nom de Iris de fisher , présenté par Ronald fisher en 1963 ,parfois aussi nommé jeu de données d'iris d'anderson du nom d'Edgar Andrson qui a collecté ces données afin de quantifier la variation de morphologie des fleurs d'iris de trois espèces à savoir Setosa , Versicolour et Verginica qui représente les trois classes respectivement. L'ensemble de données est chargé de la bibliothèque scikit-learn contient :

- Quatre attributs numériques mesurés en centimètre **:**
  - Largeur de **sépale**[22]
  - Longueur de sépale
  - Largeur de **pétale**[23]
  - Longueur de pétale
- **150** lignes (50 enregistrements pour chaque classe) il est disponible dans le lien suivant **[28].**

Figure 9:Les trois catégories de fleurs d'iris (Versicolor, Setosa,Verginica)

[22] **Sépale :** En botanique, un sépale correspond à l'ensemble des structures foliacées observées à la base de la corolle, sous les pétales. Ils sont généralement en couleur verte ils forment le calice de la fleur.
[23] **Pétale :** En botanique, un pétale est une pièce florale qui entoure le système reproducteur des fleurs constituant l'un des éléments foliacés dont l'ensemble compose la corolle d'une fleur, il correspond à une feuille modifiée.

1.4. Mnist :

Mnist (**M**odified ou Mixed **N**ational **I**nstitute of **S**tandards and **T**echnology) est un jeu de données multi classes, issues d'une base de données appelée Nist de classification d'images de chiffres manuscrits, regroupant **60 000** images d'apprentissage et **10 000** images de test et **10** classes qui représentent les dix chiffres [0,1,2,3,4,5 ,6,7 ,8,9], ces images sont en noir et blanc normalisées centrés de 28x28 pixels, téléchargé depuis l'API de Tensorflow, accessible à partir du lien **[29].**

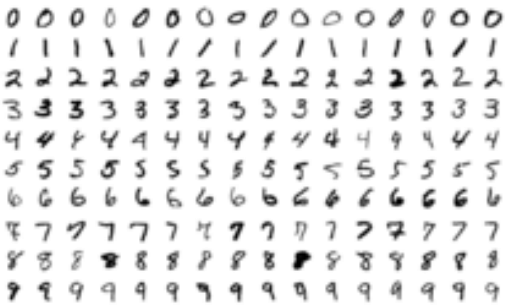

Figure2:Exemple de chiffres de la base de données MNIST

2. Tests sur les algorithmes de classification :

Après avoir implémenté les algorithmes (chapitre 3), et sélectionné l'ensemble des jeux de données sur lesquels nous allons faire le test (Section 1.), dans cette partie notre travail consiste à adapter les classifieurs aux jeux de données, autrement dit ajuster les paramètres de chaque classifieur de façon à ce que les résultats donnés par ces derniers soient favorables aux prédictions futures.

Dans ce qui suit nous allons tester KNN sur les deux paramètres : métrique de distance et la valeur K, pour les arbres de décision le paramètre à considérer est la profondeur de l'arbre, et enfin observer la variation de la justesse et de la **perte** [24]en fonction du nombre d'entrainement dans le cas des réseaux de neurones.

[24] **La perte** : erreurs de prédiction du réseau de neurones, la méthode pour la calculer s'appelle la fonction de perte**,** divers fonction existe, comme Binary crossentropy pour la classification binaire et categorical crossentropy pour la classification multi classes.

2.1. L'algorithme K-NN :

En apprentissage automatique il ne suffit pas juste de créer un modèle mais il faudra créer le plus optimal, pour améliorer la qualité du modèle KNN nous utilisons la classe **GridSearchCV** du module **model_selection** de la bibliothèque sklearn.

La classe GridSearchCV fonctionne en entraînant le modèle plusieurs fois sur une plage de paramètres que nous spécifions (le nombre des voisins **k** et la métrique de distance dans notre cas). De cette façon, nous pouvons tester notre modèle avec chaque paramètre et déterminer les valeurs optimales pour obtenir les meilleurs résultats de précision.

Selon GridSearchCV les paramètres qui ont donné les meilleurs résultats de précision sont : Métrique de distance = **Euclidienne** et **K=9**

2.2. Les arbres de décisions :

Les arbres de décision très profond risquent de surapprendre les données d'entrainement ce qui implique l'incapacité du classifieur à effectuer de bonnes prédictions sur un ensemble test, pour surveiller la profondeur de l'arbre nous utilisons le paramètre max_depth pour suivre la variation de la précision.

Les graphes de la **figure 3** et la **figure 4**, représentent la variation de la précision du classifieur **CART** implémenté, en fonction du paramètre **"max_depth"** pour les jeux de données Breast Cancer et Iris, où la courbe bleue se réfère à la variation de la précision pour l'ensemble d'entrainement quant à la courbe orange elle se réfère à la variation de la précision pour l'ensemble de test :

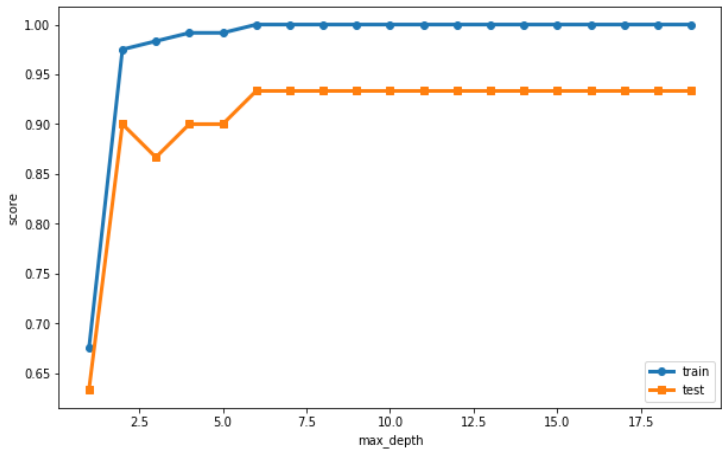

Figure3:Variation du score en fonction du paramètre max_depth pour Breast cancer

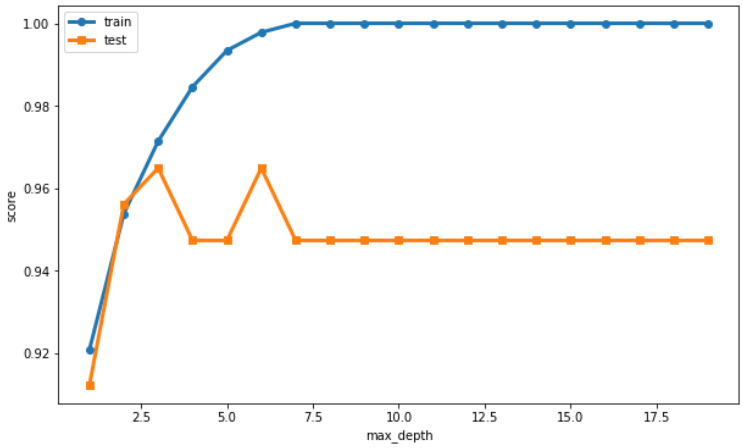

Figure4:Variation du score en fonction du paramètre max_depth pour Iris

Nous nous interessons ici à l'ensemble test pour juger la performance et valider la valeur du parametre max_depth.

Nous remarquons sur les graphiques illustrés par les **figure 3** et **figure 4**, qu'il y a une relation proportionnelle entre la profondeur de l'arbre et la precision du classifieur, ceci s'explique par le fait que la precision augmente tant que le la profondeur de l'arbre augmente, jusqu'à ce qu'il atteigne une précision dans la plage [0.94,0.96] et reste constante à partir de là pour une valeur de **max_depth=7**, mais pour Iris le graphique atteint le pic d'une precision de 0.96 pour une profondeur aux alentours de **2** et **6 et** rechute après ces valeurs.

la profondeur max_depth=7 permet d'atteindre de bons resultats pour l'ensemble test, nous fixerons alors ce parametre à la valeur 7.

2.3. Les réseaux de neurones :

2.3.1. Les réseaux de neurones simples (ANN) :

Les graphes illustrés dans la **figure 5**, **figure 6** et la **figure 7** montrent la variation de la perte en utilisant le classifieur ANN en fonction du nombre d'epochs[25] pour les jeux de données breast cancer, bank marketing et iris :

[25] **Epochs** : nombre d'entrainement du réseau de neurones

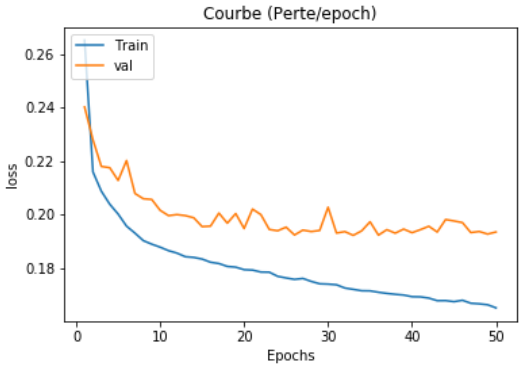

Figure5: courbe perte/epochs de Bank Marketing

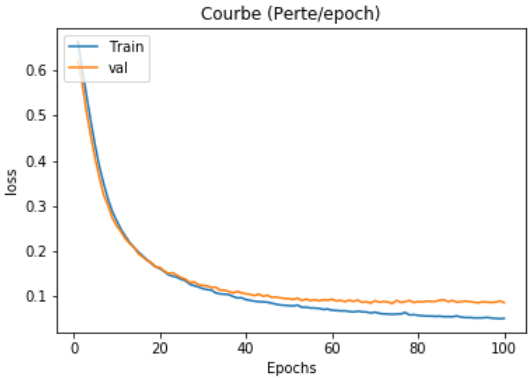

Figure 6: courbe perte/epochs de Breast Cancer

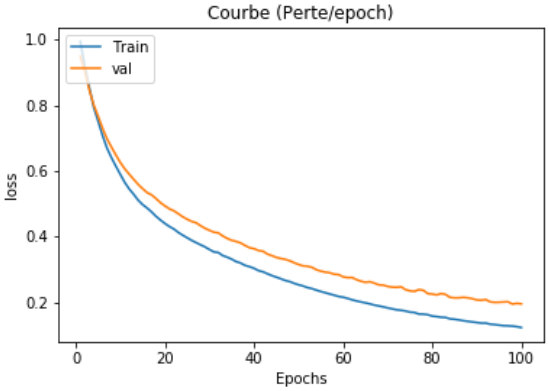

Figure7: courbe perte/epochs de Iris

L'ensemble des courbes montrent que la perte est en baisse en augmentant le nombre d'entrainement (epochs), en s'approchant de **100** epochs la perte devient minimale pour les jeux de données : Iris et Breast cancer, quant au Bank marketing au bout de **50** epochs, la perte devient considérablement petite.

La valeur du perparamètre « nombre d'epochs » utilisé par notre classifieur est **100** pour Iris et Breast cancer, **50** pour Bank Marketing.

➢ **ANN Mnist :**

Les graphes ci-dessous représentent l'évolution de la perte et la justesse en fonction des nombres d'epochs pour le jeu de données Mnist :

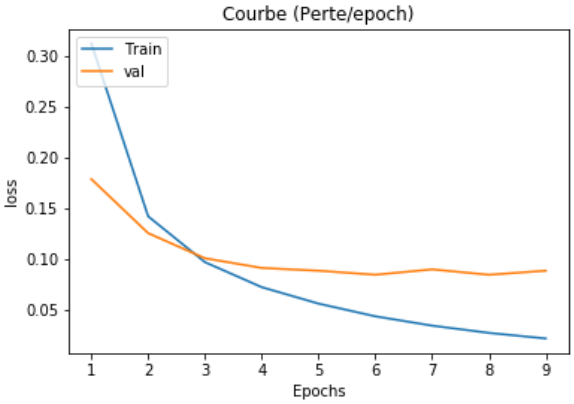

Figure 8:courbe perte/epochs ANN

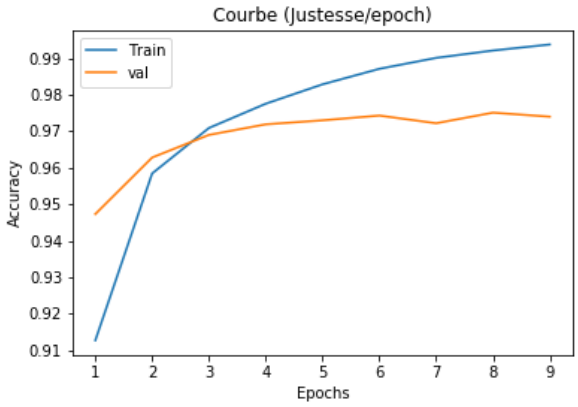

Figure 9:courbe justesse/epochs ANN

Nous pouvons conclure depuis les figures **8** et **9**, que la justesse augmente tant que le nombre d'epochs augmente et qu'à partir de 9 epochs la justesse atteint un seuil important, quant à la perte, elle atteint un seuil peu considérable aux environs de l'epochs **9**, de là l'hyperparamètres « nombre d'epochs » pour ANN Mnist est réglé à **9**.

2.3.2. Les réseaux de neurones convolutif :

Les graphes **10, 11 et 12** montrent la variation de la perte en utilisant le classifieur CNN en fonction du nombre d'epochs pour les jeux de données breast cancer, bank marketing et iris :

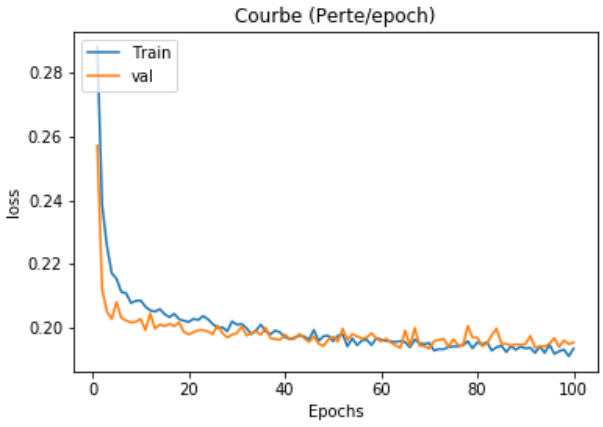

Figure 10:courbe perte/epochs Bank Marketing

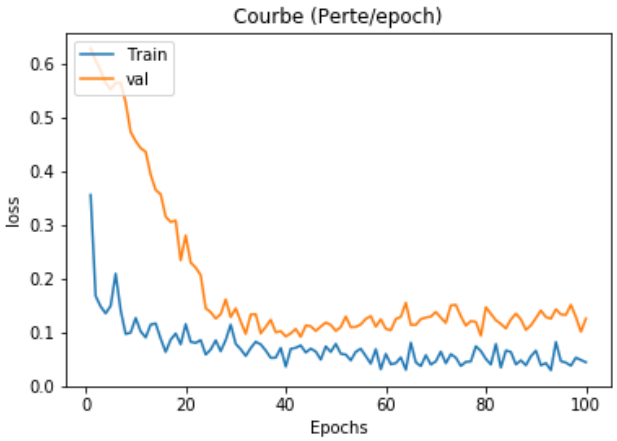

Figure 11:courbe perte/epochs Breast Cancer

Figure 10:courbe perte/epochs Iris

Nous remarquons d'après les figures **10,11 ,12** que la perte est considérablement petite en s'approchant de 100 epochs pour les jeux de données : Bank Marketing et Breast cancer et 140 Pour Iris.

La valeur du perparamètre « nombre d'epochs » utilisé par notre classifieur est **100** pour

- **CNN Mnist :**

Les courbes ci-dessous étudient la variation de la perte et de la justesse en fonction du nombre d'epochs pour le jeu de données Mnist :

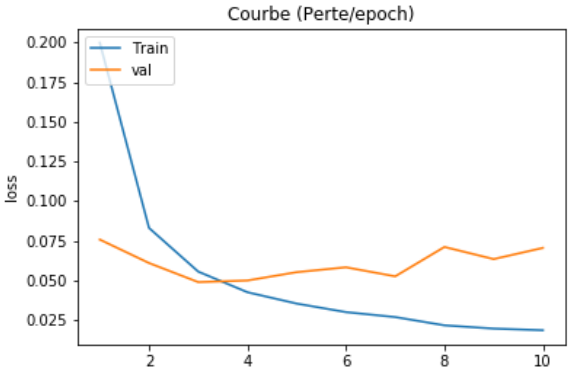

Figure13:courbe perte/epochs pour CNN cas du Mnist

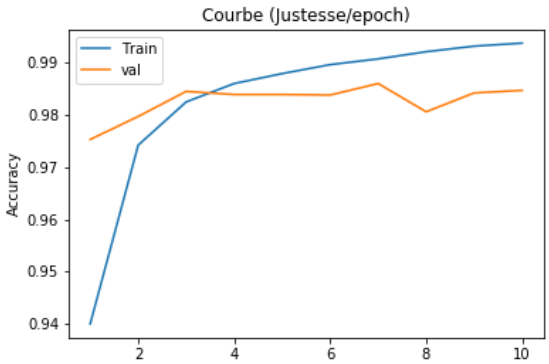

Figure 14:courbe justesse/epochs pour CNN cas du Mnist

Les figures **13 et 14** montrent la variation de la perte et la justesse du classifieur CNN pour le jeu de données Mnist. Le comportement du classifieur à l'epochs 10 fait preuve de bonnes performances, la valeur **10** serait alors le nombre d'epochs pour le classifieur CNN.

3. Evaluation des algorithmes de classification supervisée :

Afin de valider une technique de classification, cette dernière doit vérifier des qualités de précision et de robustesse. Pour mesurer ces qualités il existe plusieurs métriques en pratique, nous avons choisi dans notre travail un ensemble de six métriques : matrice de confusion, rappel, précision, f1-score, justesse et enfin l'indice de Jaccard.

Dans ce qui suit nous allons faire une étude comparative entre les algorithmes de classification supervisée implémentés sur les quatre jeux de données en utilisant les métriques citées, mais nous allons au préalable donner un aperçu sur le principe des métriques utilisées.

3.1. Métriques d'évaluation :

3.1.1. La matrice de confusion :

C'est un outil permettant de mesurer les performances d'un modèle de machine learning en vérifiant notamment quelles prédictions sont exactes par rapport à la réalité dans les problèmes de classification. C'est une matrice **N*N,** L'un des axes de la matrice de confusion est l'étiquette prédite par le modèle, et l'autre axe représente l'étiquette réelle. **N** correspond au nombre de classes.

		Classe réelle	
		-	+
Classe prédite	-	True Negatives *(vrais négatifs)*	False Negatives *(faux négatifs)*
	+	False Positives *(faux positifs)*	True Positives *(vrais positifs)*

- **Vrais positifs (VP) :** Les cas où la prédiction est positive, et où la valeur réelle est effectivement positive.
- **Vrais négatifs (VN) :** les cas où la prédiction est négative, et ou la valeur réelle est effectivement négative.
- **Faux positifs (FP) :** les cas où la prédiction est positive, et ou la valeur réelle est négative.
- **Faux négatifs (FN) :** les cas où la prédiction est négative, et ou la valeur réelle est positive.

3.1.2. La précision :

La fréquence à laquelle le modèle prédit correctement la classe positive.

Précision =VP / VP+FP / tq : 0<=précision<=1 **(4,1)**

3.1.3. Le rappel (recall) :

Le nombre de classes positives que le classifieur a prédit correctement, calculé avec la formule est la suivant :

Rappel =VP /VP+FN /tq : 0<=rappel<=1 **(4,2)**

3.1.4. F1_score :

Une métrique d'évaluation peut être interprété comme une moyenne pondérée de la précision et du rappel ou elle atteint sa meilleure valeur a 1 et son pire score a 0.la formule est :

F1_score = 2*(précision*rappel) / (précision+ rappel) /tq : 0<=f1_score<=1 **(4, 3)**

3.1.5. La Justesse (Accuracy) :

La proportion des prédictions correctes effectuées par le modèle.

Justesse= nbr prédictions correctes/ nbr total des prédictions (4,4)

- Cas de la classification binaire : la formule précédente peut s'écrire comme suit :

Justesse = VP+VN/VP+VN+FP+FN / tq : 0<=accuracy<=1 **(4,5)**

3.1.6. Le Coefficient de Jaccard :

Calcule la similarité entre deux ensembles de données, il est donné par la formule **:**

Coefficient de Jaccard = VP /VP +FP+FN (4,6)

3.2. Résultats des classifieurs :

Dans cette section seront présentés les résultats des métriques sur les algorithmes implémentés pour chaque jeu de données :

3.2.1. Breast cancer :

Tableau4:Résultas des classifeurs pour le jeu de données Breast Cancer

Classifieurs	Rappel	Précision	Justesse	F1_score	Mesure de Jaccard	Matrice de confusion
KNN	1.0	0.9428	0.9649	0.9706	0.9428	[[44 4] [0 66]]
ID3	1.0	0.9296	0.9561	0.9635	0.9296	[[43 5] [0 66]]
C4.5	0.9428	0.9296	0.921	0.9362	0.88	[[39 5] [4 66]]
CART	0.9697	0.9412	0.9474	0.9552	0.9143	[[44 4] [2 64]]
ANN	1.0	0.9706	0.9824	0.9706	0.9706	[[46 2] [0 66]]
CNN	1.0	0.9565	0.9737	0.9778	0.9565	[[45 3] [0 66]]

3.2.2. Bank Marketing :

Classifieurs	Rappel	Précision	Justesse	F1-score	Mesure de Jaccard	Matrice de confusion
KNN	0.2689	0.5829	0.8887	O.3681	0.2255	[[7054 191] [726 267]]
CART	0.4864	0.6607	0.9079	0.5603	0.3892	[[6997 248] [511 483]]
ID3	0.1581	0.8093	0.894	0.2645	0.1524	[[7208 37] [836 157]]
C4.5	0.2274	0.6415	0.902	0.3358	0.2018	[[7227 114] [693 204]]
ANN	0.4663	0.6633	0.9071	0.5476	0.377	[[7010 235] [530 463]]
CNN	0.4491	0.6788	0.9079	0.5406	0.3704	[[7034 211] [547 446]]

Tableau5:Résultas des classifieurs pour le jeu de données Bank Marketing

3.2.3. Iris :

Classifieurs	Rappel	Précision	Justesse	F1_score	Mesure de Jaccard	Matrice de Confusion
KNN	0.9333	0.9436	0.9333	0.9328	0.8769	[[8 0 0] [0 9 2] [0 0 11]]
ID3	0.8667	0.8667	0.8667	0.8667	0.7743	[[8 0 0] [0 9 2] [0 2 9]]
C4.5	0.6	0.5791	0.6	0.5472	0.4721	[[10 0 0] [0 1 10] [0 2 7]]
CART	0.9333	0.9436	0.9333	0.9328	0.8769	[[8 0 0] [0 9 2] [0 0 11]]
ANN	0.9333	0.9333	0.9333	0.9333	0.8778	[[8 0 0] [0 10 1] [0 1 10]]
CNN	0.9333	0.9436	0.9333	0.9328	0.8769	[[8 0 0] [0 11 0] [0 2 9]]

Tableau6:Résultas des classifieurs pour le jeu de données Iris

3.2.4. Mnist :

	Rappel	Précision	Justesse	F1-score	Indice de Jaccard
ANN	0.9736	0.9737	0.9736	0.9736	0.9487
CNN	0.9843	0.9844	0.9843	0.9843	0.969

Tableau7:Résultats des classifieurs pour le jeu de données Mnist

3.2.4.1. Matrices de confusion de ANN et CNN du jeu de données Mnist :

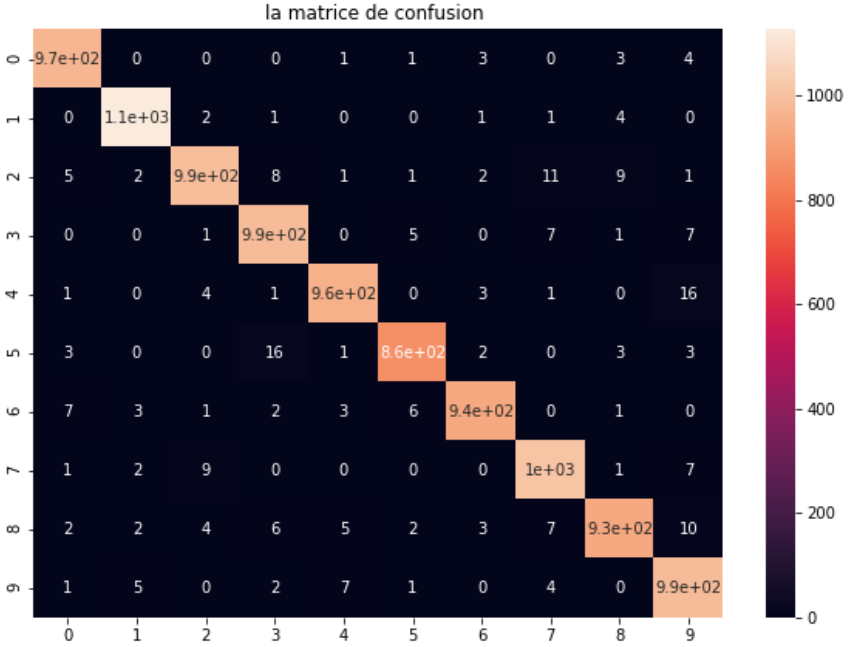

Figure 15:matrice de confusion ANN Mnist

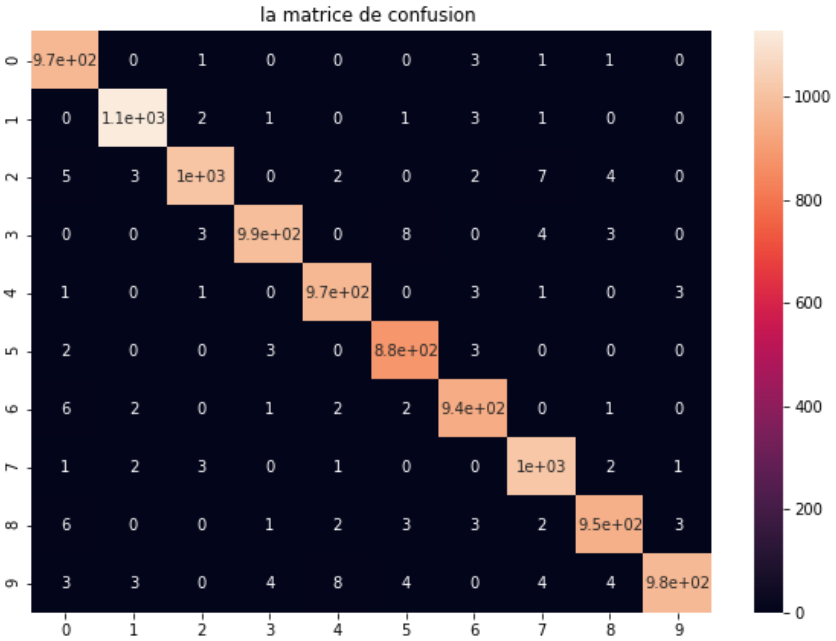

Figure 11:Matrice de confusion CNN Mnist

4. Discussion des résultats :

- le **tableau 4**, montre que tous les algorithmes ont donné de bons resultats pour le jeu de données Breast Cancer, où nous remarquons que les valeurs des métriques varient aux alentours de 0.90.Pour ce jeu de données, les réseaux de neurones ont eu les meilleurs résultas, vient ensuite l'algorithme KNN et enfin les algorithmes d'arbres de décision.
- Le **tableau 5** résume les résultats des métriques pour le jeu de données Bank Marketing, nous synthétisons d'apres ces resultats que la justesse et la precision sont assez elevés, contrairement aux metriques restantes les valeurs varient dans la plage [0.2,0.5], les algorithmes qui ont montré leurs performances pour ce jeu de données sont : les reseaux de neurones et l'algoruthme CART.
- D'apres les resultats du jeu de données Iris montré dans le **tableau 6**, le classifieur C4.5 n'est pas adapté à ce jeu de données, contrairement aux classifieurs, ANN, CNN, CART, KNN, ID3 avec des performances élevées pour des valeurs dépassant 0.86.
- Dans l'ensemble, les reseaux de neurones ANN et CNN implémentés pour le jeu de données Mnist,ont prouvé leurs performances avec un taux depassant 95 % d'apres le **tableau 7**
- Les deux figures **15,16** illustrent la matrice de confusion pour les réseaux de neurones ANN et CNN.

Dans chacune des matrices de confusion, la colonne correspond à la classe $\mathbf{c_i}$ prédite par le réseau de neurone, la ligne correspond à la classe réelles $\mathbf{c_y}$ de l'ensemble du teste du jeu de données MNIST. Les valeurs de chaque cellule représentent le nombre d'enregistrements de la classe réelle $\mathbf{c_y}$ qui ont été estimés comme appartenant à la classe $\mathbf{c_{i\ (i\ :0à9)}}$ à la différence de la diagonal qui représente le nombre d'enregistrements estimés comme appartenant à la même classe réelle $\mathbf{c_{y(y\ :0à9)}}$.

Nous remarquons d'après les deux illustrations que les valeurs de la diagonal pour le CNN sont plus grandes que celles de ANN, par exemple pour la classe **2,** CNN a réussi à prédire correctement **1000** images quant au ANN **990** images qui ont été prédit correctement. Pour la classe **9,** CNN n'a prédit aucune image comme appartenant à la classe **9** tandis que ANN a prédit **4** images comme appartenant classe **9.**

D'après les résultats retournés par les matrices de confusion nous pouvons conclure que CNN est plus performant qu'ANN pour le jeu de données de MNIST.

Conclusion :

Nous avons réussi à travers la série des tests effectués sur les algorithmes de créer le classifieur le plus adapté à chaque jeu de données en déterminant les meilleurs paramètres pour chacun.

D'abord, le choix de la valeur k et la métrique de distance influe sur la qualité de la prédiction du classifieur KNN, la distance euclidienne et **k=9** s'adaptaient mieux à l'objet de notre étude. Nous avons également conclu l'impact du nombre d'entraînements sur un réseau de neurones, qu'à partir d'un certain nombre d'entrainements, la performance du réseau n'est plus influencée. Quant aux arbres de décision le paramètre à prendre en compte est la profondeur de l'arbre, car il est strictement inutile de laisser l'arbre accroitre, puisque qu'il devient difficilement interprétable sans pour autant augmenter les performances du classifieur, les tests effectués ont montré que la profondeur **7** était suffisante pour l'ensemble des jeux de données.

L'évaluation faite à l'aide des métriques nous permis de déduire que la technique des réseaux de neurones présente les meilleurs résultats pour tous les ensembles de données.

Et nous pouvons conclure que la performance d'un algorithme est dépendante du jeu de données, le choix des paramètres.

Conclusion générale :

Ce mémoire s'inscrit dans l'objectif d'application des algorithmes de classification supervisée de data mining. Dans les lignes qui suivent nous procédons à un récapitulatif du travail effectué.

Nous nous sommes attelés, dans un premier temps**,** à définir les différents concepts liés au processus ECD. Nous avons montré que le data mining n'est qu'un maillon de traitement au sein du processus ECD permettant d'extraire des connaissances, ensuite nous avons procédé à un état de l'art au data mining dont nous avons présenté les différentes tâches que peut effectuer un système de data mining et les différentes techniques qui peuvent être appliquées pour accomplir ces tâches.

Dans un second temps, nous nous sommes basés sur les techniques de classification supervisée où nous avons détaillé quelques-unes, nous avons vu qu'il en existe plusieurs algorithmes, chaque algorithme est adapté à un contexte particulier, il peut donc réussir dans un contexte et échouer dans un autre. Nous avons conclu que le choix du bon algorithme se fait en fonction des données et des besoins.

La troisième partie de ce mémoire était consacrée à l'architecture générale de notre application, la structure et le fonctionnement des différents modules où nous avons présenté les algorithmes implémentés ajouter à cela les outils et l'environnement de développement qui nous ont facilité développement de l'application.

La dernière partie de ce mémoire portait sur les tests et les résultats des métriques d'évaluation des algorithmes implémentés sur les quatre jeux de données Breast cancer, Iris, Bank marketing et Mnist. Les résultats nous ont permis de constater qu'un algorithme peut être performant pour un jeu de données précis mais pas pour un autre.

Au final on ne peut pas affirmer qu'un tel algorithme est meilleur par rapport à un autre, le choix dépend du jeu de données, de sa taille et de ses caractéristiques mais aussi de l'algorithme utilisé et de ses paramétres.

Ce travail n'a pas été exempt d'obstacles. En effet, nous avons été confrontés à différentes difficultés, par exemple au niveau de l'implémentation des algorithmes, cependant il était une véritable expérience et une chance pour la découverte du domaine du data science.

Comme perspectives par rapport à ce travail, il serait intéressant d'implémenter d'autres techniques de classification supervisée ainsi les algorithmes de clustering.

Une autre perspective consiste à étendre le champ d'application pour faire en sorte une application qui serait capable à la fois d'analyser et de traiter n'importe quel jeu de données dans n'importe quel domaine, appliquer et évaluer diverses techniques du data mining, que ce soit dans le but de la classification ou la régression.

ANNEXE A :

Algorithme A-priori :

L'algorithme A-priori est un algorithme d'exploration de données conçu en 1994, par Rakesh Agrawal et Ramakrishnan Sikrant, dans le domaine de l'apprentissage des règles d'association. Il sert à reconnaître des propriétés qui reviennent fréquemment dans un ensemble de données (motifs fréquents) et d'en déduire une catégorisation.

b. Concepts de base :

Une transaction **T** (un achat) : Est identifiée par un **TID** (**T**ransaction **I**Dentifier) constituée d'un sous-ensemble d'items **J ⊆ m** tels qu'un item représente tout article **i**,

- **T= {t1, t2,…tn}** un ensemble de **n** transactions
- Un **itemset** est un ensemble d'items, il est aussi appelé **motif**.
- Une transaction **ti** contient un itemset ou motif **m** si et seulement si **m⊆ ti**
- **Support d'un motif :**

Le support d'un motif **m** est le rapport entre le nombre de transactions contenant le motif **m** sur le nombre total de transactions. Ou encore :

- Le support d'un motif m noté **sup(m)** est le pourcentage de transactions de **T** qui contiennent **m.**

$$\textbf{Sup } (m) = \frac{\text{Nombre de transactions contenant le motif } \mathbf{m}}{\text{Nombre total de transactions}} \quad \text{avec } 0 \leq \text{sup(m)} \leq 1$$

- **Motif fréquent**

Un motif est dit fréquent dans une base de transactions, si son support dépasse un seuil **minsup** appelé support minimum défini par l'utilisateur.

- **Indice de fiabilité de la règle :**

La règle **m1→m2** est vérifiée dans l'ensemble **T** avec un support **S**, où **S** est le pourcentage d'objets dans **T** contenant **m1∪m2**. **S** est l'indicateur de fiabilité de la règle**m1→m2** tels qu

$$\textbf{S} = \textbf{Sup } (m1 \rightarrow m2) = \frac{\text{Nombre de transactions contenant le motif } \mathbf{m1} \cup \mathbf{m2}}{\text{Nombre total transactions}}$$

Remarque :

Les règles qui dépassent un minimum de support et un minimum de confiance sont appelées règles solides.

b. Structure de l'algorithme :

Algorithme Apriori

Données: Base **T** de transactions, ensemble m d'items, seuil

Variables: **F** : ensemble des motifs fréquents

C : ensemble des motifs fréquents candidats

Sorties : F (ensemble des motifs fréquents)

i←1

C1 ← ensemble des motifs de taille 1 (un seul item)

Tant que **Ci**≠ ∅ faire

Calculer le Support de chaque motif m∈Ci dans la base

Fi← {**m** ∈ **Ci** /**sup**(**m**) ≥ **minsup**}

Ci+1← toutes les combinaisons possibles des motifs de Fi de taille i + 1

(en utilisant la jointure)

i ← i + 1

Fin Tant que

retourner (∪i ≥1)Fi

Bibliographie :

[1] MENOUER T, DERMOUCHE M, 2010.Mémoire fin d'étude.Systémes Informatiques. Application de techniques de data mining pour la classification.ESI, p : 4.

[2] Fahmi Ben Rejab, présentation en ligne, Introduction Data Mining, 2017/2018 disponible sur : https://slideplayer.fr/slide/13003096/

[3] Agrawal, T. Imielinski A. 1993."Mining Association Rules Between Sets of Items in Large Databases". IBM Almaden Research Center 650 Harry Road, San Jose, CA 95120, p: 207-216 disponible sur : https://rakesh.agrawal-family.com/papers/sigmod93assoc.pdf

[4] Agrawal R, Ramakrishnan S.1994."Fast Algorithms for Mining Association Rules". IBM Almaden Research Center 650 Harry Road, San Jose, CA 95120, p:13 disponible sur : http://rakesh.agrawal-family.com/papers/vldb94apriori.pdf

[5] Jiawei H, Jian P, Yiwen Y,Runying M ."Mining frequent patterns without candidate generation". School of Computing Science Simon Fraser University p : 12 disponible sur : https://www.cs.sfu.ca/~jpei/publications/sigmod00.pdf

[6] Dongkuan, Yingjie T, Article en ligne, "A Comprehensive Survey of Clustering Algorithms", publié : le 12 aout 2015 disponible sur : https://link.springer.com/article/10.1007/s40745-015-0040-1

[7] Oumiloud Horiya, Mokeddem Asma , Mémoire fin d'études ,"Classification non supervisée : Application de k-means",Université Abou Bakr Belkaid– Tlemcen, p: 12,2013-2014,disponible sur :

http://dspace.univ-tlemcen.dz/bitstream/112/6391/1/Application-de-k-means.pdf

[8] Hierarchical Clustering in Data Mining .Article en ligne, publié le 05-02-2020, disponible sur : https://www.geeksforgeeks.org/hierarchical-clustering-in-data-mining/

[9] Neeraj,"Regroupement hiérarchique utilisant DIANA et AGNES", publié le : 25 juillet 2019 à 7h30, disponible sur :https://www.datasciencecentral.com/profiles/blogs/usarrests-hierarchical-clustering-using-diana-and-agnes

[10] J. MACQUEEN.1967." Some methods for classification and analysis of multivariate observations".University of California, Los Angeles. p:281–297 disponible sur : http://www.cs.cmu.edu/~bhiksha/courses/mlsp.fall2010/class14/macqueen.pdf

[11] Abdelmadjid Boukra , Thèse, " Résolution du problème de tournées de véhicules avec collecte et livraison simultanées avec une approche coopérative de métaheuristiques",p:41,

Juin 2014,disponible sur : https://www.researchgate.net/figure/Exemple-de-deroulement-de-lalgorithme-K-means_fig18_337840545

[12] Leanrad KAUFMAN, Peter J. ROUSSEUW, Article en ligne; "Clustering by means of medoids ", 1987, disponible sur : https://www.researchgate.net/profile/Peter_Rousseeuw/publication/243777819_Clustering_by_Means_of_Medoids/links/00b7d531493fad342c000000.pdf

[13] Leanrad KAUFMAN, Peter J. ROUSSEUW, livre en ligne , "Finding Groups in Data: An Introduction to Cluster Analysis",p:126,1990,disponible sur : **https://books.google.dz/books?hl=fr&lr=&id=YeFQHiikNo0C&oi=fnd&pg=PR11&dq=Finding+Groups+in+Data&ots=5Bnby8QCrB&sig=ojhfRujwTNbXYA11UBS2WgEn344&redir_esc=y#v=onepage&q=Finding%20Groups%20in%20Data&f=false**

[14] Raymond T, Jiawei H . 1994." Efficient and Effective Clustering Methods for Spatial Data Mining",Department of Computer Science University of British Columbia Vancouver, B.C., V6T 124, Canada , Jiawei Han School of Computing Sciences Simon Fraser University Burnaby, B.C., V5A lS6, Canada .p 144-155.disponible sur : http://www.vldb.org/conf/1994/P144.PDF

[15] Dempster A. P,Laird N. M, Rubin D. B. 1977." Maximum Likelihood from Incomplete Data via the EM Algorithm ". Journal of the Royal Statistical Society. Series B (Methodological), Vol. 39, No. 1. (1977).p: 1-38.disponible sur : https://www.ece.iastate.edu/~namrata/EE527_Spring08/Dempster77.pdf

[16] Ester M, Kriegel H-P, Sander J, Xu X. 1996. " Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise".Institute for Computer Science, University of Munich Oettingenstr. 67, D-80538 Miinchen, German. p: 226–231, disponible sur : https://www.aaai.org/Papers/KDD/1996/KDD96-037.pdf

[17] Mihael Ankerst, Markus M. Breunig, Hans-Peter Kriegel, Jörg Sander," Ordering Points To Identify the Clustering Structure",University of Munich, Germany, disponible sur :

https://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=8947465ED8D218DDB011CC64B76E48D1?doi=10.1.1.129.6542&rep=rep1&type=pdf

[18] Chaouche Y, Azencott C-A, cours en ligne, "Explorez avec des algorithmes non supervisé". Disponible sur : https://openclassrooms.com/fr/courses/4379436-explorez-vos-donnees-avec-des-algorithmes-non-supervises/4379571-partitionnez-vos-donnees-avec-dbscan

[19] " Informatique et Techniques Numériques en Economie",disponible sur : http://www.up2.fr/M1/td/TD10_2.html

[20] G. V. Kass, 1980, An Exploratory Technique for Investigating Large Quantities of Categorical Data, Journal of Applied Statistics, Vol. 29, No. 2, p:119-127

[21] J. Ross Quinlan, 1986, « Induction of decision trees », 6 Kluwer Academic Publishers, Boston - Manufactured in The Netherland, p. 81-106, disponible sur : https://link.springer.com/content/pdf/10.1007/BF00116251.pdf

[22] Quinlan, J. R. C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, 1993.disponible sur : https://link.springer.com/article/10.1023/A:1022645310020

[23] Breiman, L., Friedman, J.H., Olshen, R.A., and Stone, C.J. (1984) Classification and Regression Trees, Brooks/Cole Publishing, Monterey, 1984, 358 pages

[24] Institut des Sciences de l'Information et de recherches scientifiques en Informatique, Automatique, Robotique, «Réseaux bayésiens » publié le 29 août 2016.disponible sur : https://twitter.com/ins2i_cnrs/status/770159804748881920

[25] Corina Cortes, Vladmir Vapnik.1995. "Support-Vector Networks".luwer Academic Publishers, Boston. Manufactured in The Netherlands. P : 25. Disponible sur : http://image.diku.dk/imagecanon/material/cortes_vapnik95.pdf

[26] Support Vector Machine Classification,colab reasearch, disponible sur :

https://colab.research.google.com/github/akshayrb22/playing-with-data/blob/master/supervised_learning/support_vector_machine/svm.ipynb#scrollTo=qan9BUe7octB

[27] " SVM a marge dure ", Article en ligne , disponible sur : https://reussirlem2info.files.wordpress.com/2013/02/bekhelifi-okba-svm-resume.pdf

[28] "Math Behind SVM(Kernel Trick) " ,Figure, 16 février 2020, disponible sur : https://mc.ai/math-behind-svmkernel-trick/

[29] Ben Aisen, Article en ligne,"Une comparaison des méthodes SVM multiclasses",15 décembre 2006 disponible sur : https://courses.media.mit.edu/2006fall/mas622j/Projects/aisen-project/

[30] T. M. COVER, P. E. HART, January 1967, "Nearest Neighbor Pattern Classification" . IEEE Transaction on information theory, VOL. IT-IS, NO. 1. P : 22-27.Disponible sur : http://ssg.mit.edu/cal/abs/2000_spring/np_dens/classification/cover67.pdf

[31] WASSIM.L, Présentation en ligne , Algorithme knn,2012/2013, disponible sur : https://fr.slideshare.net/wassimlahbibi/algorithme-knn

[32] Warren S. McCulloch and Walter Pitts. "A logical calculus of the ideas immanent in nervous activity. Bulletin of mathematical biophysics", vol. 5 (1943), pp. 115–133.;1943,disponible sur : https://www.cs.cmu.edu/~./epxing/Class/10715/reading/McCulloch.and.Pitts.pdf

[33] "Réseaux de neurones/Qu'est-ce qu'un neurone ? ", mis à jour le 14/01/2019,disponible sur : https://fr.wikiversity.org/wiki/R%C3%A9seaux_de_neurones/Qu%27est-ce_qu%27un_neurone_%3F

[34] SADJIDA.B, Présentaion en ligne , "Réseaux de neurons", publier le 1 déc. 2015,disponible sur : https://fr.slideshare.net/bechicalla/rseaux-neurons

[35] Réseau de neurones artificiels, disponible sur : https://fr.wikipedia.org/wiki/R%C3%A9seau_de_neurones_artificiels

[36] Nicotupe, " Deep Learning ", Article en ligne publié le : 01/03/2016, disponible sur : https://www.podcastscience.fm/dossiers/2016/03/01/deep-learning/ 01

[37] FAVIO.V A, Article en ligne , "Conversation about Deep Learning",15 août 2018, disponible sur : https://towardsdatascience.com/a-conversation-about-deep-learning-9a915983107

[38] "Tutoriel sur le réseau de neurones convolutifs: du niveau basique au niveau avancé", Article en ligne, disponible sur : https://missinglink.ai/guides/convolutional-neural-networks/convolutional-neural-network-tutorial-basic-advanced/

[39] Shuai Zhang, "Introduction to CNN", présentation en ligne, publié le : 12 février 2017

disponible sur : https://www.slideshare.net/ShuaiZhang33/lg-cnn20170213

[40] SHUBHAM. K, "Build Your First Image Classification Model with The MNIST Dataset", publier le 25/10/2019 disponible sur : https://medium.com/subex-ai-labs/build-your-1st-deep-learning-classification-model-with-mnist-dataset-1eb27227746b