REPUPLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHESCIENTIFIQUE
UNIVERSITE MOULOUD MAMMERI TIZI-OUZOU (UMMTO)
FACULTE DE GENIE ELECTRIQUE ET INFORMATIQUE
DEPARTEMENT D'INFORMATIQUE

Mémoire

de fin d'études

En vue d'obtention du diplôme de Master 2 Informatique
Option : Conduite de projet informatique

Thème

Etude et recherche bibliographique sur les méthodes de classification

Proposé et encadré par :

Mr AÏT EL HADJ Ali

Présenté par:

M^elle^ Lydia BATACHE
M^elle^ Samia DRICI

Devant le jury d'examen suivant :

Mme AÏT EL HADJ Fatiha	**Présidente**
Mr SAIDANI Faycal Rédha	**Examinateur**

Promotion 2018 / 2019

REMERCIEMENTS

"Aucun de nous ne s'est élevé à la seule force de son poignet. Nous sommes arrivés parce que quelqu'un s'est baissé pour nous aider. "

Thurgood Marshall

Au terme de ce travail,

Nos remercions avant tout le bon Dieu tout puissant de nous avoir donné patience, courage et volonté pour réussir notre mémoire,

Nous tenons à exprimer notre profonde gratitude à Mr ***AÏTELHADJ*** **Ali**, Maître de conférences à l'UMMTO, pour avoir encadré et dirigé notre travail. Nous le remercions pour ses conseils et ses remarques constructives qui nous ont permis d'améliorer la qualité de nos travaux,

Nos vifs remerciements vont aux membres du jury pour nous avoir fait l'honneur d'examiner et d'évaluer notre travail avec le poids de leurs compétences,

A tous les enseignants qui ont assuré notre formation durant notre parcours universitaire, nous souhaitons qu'ils trouvent dans ce travail l'expression de notre infinie reconnaissance,

Nous souhaitons également exprimer notre gratitude à tous ceux qui de près ou de loin ont participé à l'élaboration du présent travail.

Je dédie ce travail,

C’est avec un immense plaisir que je dédie ce travail,

A ma fierté, ma chère et mon adorable mère que j’adore, pour son sacrifice, son soutien tout au long de mes études et pour toutes les valeurs magnifiques qu’elle m’a donné durant toute ma vie,

A la mémoire de mon cher père que Dieu le garde dans son vaste paradis,

A mon chère Mourad et toute sa famille, A mes chers

sœurs et frères ainsi que leurs familles,

A ma très chère binôme Samia et toute sa famille,

A tous mes amis sans exception.

Lydia

Je dédie ce travail,

A mes chers parents, qui ont toujours été à mes côtés et ont toujours cru en moi. Pour leur amour, et leur soutien permanent tout au long de mes années d'études. Que Dieu vous protège,

A mes frères et sœurs, qui de par leur amour et leur soutien au quotidien ont contribués à la réalisation de ce travail,

, A tous mes amis, qui m'ont toujours soutenu,

Ainsi qu'à tous mes camarades du département informatique.

Samia

Liste des abréviations

RI : recherche d'information

CURE : Clustering Using REpresentatives

BIRCH :Balanced Iterative Reducing and Clustering using Hierarchies

ROCK :RObust Clustering using linKs

TSVQ :Tree Structured Vector Quantization

PAM :Partitioning Around Medoids

CLARA : Clustering LARge Applications

CLARANS : Clustering Large Applications based upon RANdomised Search

DBSCAN :Density Based Spatial Clustering of Applications with Noise

OPTICS : Ordering Points To Identify Clustering Structure

DBCLASD : distribution-based clustering algorithm of large spatial databases

STING : STatistical INformation Grid

CLIQUE : Clustering In QUEst

KNN : k-Nearest Neighbors

SVM : Support Vector Machine

RN : Réseaux de Neurones

AD : Arbre deDdécision

ID3 : Inductive Decision-tree

CAR T : Classification And Regression Trees

CHAID : Chi-squared Automatic Interaction Detection

VN : Vrai négatifs

FN : Faux négatifs

FP : Faux positifs

VP : Vrai positifs

WEKA :Waikato Environment for Knowledge Analysis

[1]. Trésor de la Langue Française Informatisé

Liste des figures

Liste des tableaux

Sommaire

Chapitre II : Classification Supervisée

Chapitre III : Classification non supervisé

Chapitre IV : Présentation du logiciel Weka

Chapitre V : Implémentation et représentation des résultats

Tout ce qui nous entoure, qu'il s'agisse de choses physiques ou abstraites, nous apparait de façon organisée. Lorsque nous voyons un animal, nous le désignons systématiquement par l'espèce à laquelle il appartient. Un sentiment sera également catégorisé, on parlera par exemple d'un sentiment de tristesse, de joie ou de peur. Pourtant, chacun éprouve la peur différemment. Il s'agit, en fait, d'un phénomène, conscient ou non, mais naturel et indispensable, de simplification pour mieux comprendre et mieux communiquer. L'exemple que nous venons d'évoquer n'est pas anodin. En effet on attribue les premières recherches théoriques sur la classification, aux besoins exprimés par les biologistes, de spécifier des classes ou espèces animales. Cette classification était censée permettre, étant donnée une description (simple) d'un animal, de "l'étiqueter" par tel ou tel nom d'espèce. Le problème s'est avéré plus complexe qu'il n'y paraissait...

Les applications se sont multipliées, chacune apportant son lot de problèmes, plus complexes les uns que les autres. Après les biologistes, ce sont les historiens, les médecins, les sociologues ou les didacticiens qui ont manifesté le besoin de classer leurs reliques, leurs patients ou leurs élèves. Aujourd'hui, en nous intéressant aux problèmes issus des technologies de l'information et de la communication, nous sommes amenés à traiter des données très complexes, par leur nature, leur taille (grands volumes de données appelés base de données).

Le Data Mining est particulièrement adapté aux traitements de ces bases de données afin d'être exploiter. L'augmentation constante de ces données à analyser dans diverses disciplines telles que la médecine, la biologie et l'économie devient de plus en plus difficile d'extraire l'information utile. Pour y parvenir, on fait souvent appel à la classification**.**

Se présent travail traite comme problématique :
Qu'est-ce qu'une classification ? Qu'elles sont ces approches ? Et comment les appliquer pour fouiller les données ?

L'objectif de notre travail est de donner la possibilité aux professionnels d'étudier et d'appliquer les différentes méthodes d'apprentissage supervisé et non supervisé à travers l'outil de classification Weka et d'analyser les résultats afin d'évaluer les performances de chaque méthode .

Ce présent mémoire s'articule autour de cinq chapitres. Après une introduction générale, nous commençons par présenter dans le premier chapitre les notions de bases sur la classification, ses différentes approches, ses différentes étapes, ainsi que ses divers domaines d'application et les critères qu'il faut pour une bonne classification. Le deuxième chapitre est consacré sur les méthodes de classification supervisées. Le troisième chapitre est réservé pour les méthodes de classification non supervisées. Le quatrième chapitre traitera du logiciel Weka, de ses différentes fonctionnalités et interfaces en se focalisant particulièrement sur son interface principale, son mode d'utilisation, spécialement dans le domaine de la classification supervisée et non supervisée. Enfin, le dernier chapitre est consacré aux étapes d'implémentation de notre travail et l'interprétation des résultats obtenus.

1. Introduction

«Le seul moyen de faire une méthode instructive et naturelle, est de mettre ensemble les choses qui se ressemblent et de séparer celles qui différent les unes des autres.»

M. Georges Buffon, Histoire naturelle, 1749.

Il est clair que le processus général de la classification dans le domaine informatique essaie de l'appliquer sur des données (points, tableaux, images, sons, . . . etc.), n'échappe pas à la règle imposée par ce célèbre naturaliste et écrivain Georges Buffon, et quel travail général des méthodes de classification, depuis 1749, consiste à imiter et automatiser ce principe en utilisant et inventant des moyens adéquats (matériaux-calculateurs-, et des théories classificatoires...etc.)

Allons de ce principe, nous présenterons dans ce chapitre tout d'abord ce que c'est la classification, ses méthodes, techniques, ses grandes approches, domaines d'applications, . . . etc.

2. Historique

La classification est l'une des tâches les plus anciennes de la Recherche d'information (RI). Ses débuts remontent aux années 1960, environ une quinzaine d'années après l'apparition du terme RI dans le mémoire de maitrise de Calvin N.Mooers [1]. Depuis, elle est considéré comme une étape cruciale dans la conception de Système de Recherche d'Information et a connu un développement considérable durant les 15 dernières années. Nous citons à titre d'exemple les travaux de Larry Page et Sergey Brin qui ont effectué dans le cadre de projet Google en 1998, une classification des documents en effectuant un poids aux termes suivant leurs origines (le PageRink décrit dans [2]).

3. Notion de classe

La notion de classe pour un système de classification a été habituellement synonyme de « thème ». Dans ce contexte, classer les documents revient à les organiser par différentes thématiques. Cependant, la problématique de classification a évolué en même temps que les besoins et elle s'intéresse aujourd'hui à différentes tâches pour lesquelles les catégories ne sont pas interprétables comme des thèmes : ainsi, par exemple, les tâches consistant à classer les documents par auteur, par genre, par style, par langue, ou encore selon que le document exprime un jugement positif ou négatif, etc.…

Ainsi la classe va correspondre à un besoin d'information d'un utilisateur ou d'une société et n'est donc pas obligatoirement un thème unique. Nous considérerons dans la suite qu'une classe est simplement une étiquette à associer à des documents **[3]**.

Dans la figure 1.1, un système de classification d'emails est représenté où les classes peuvent être de différentes natures (thèmes, messages provenant de certaines personnes, messages d'un certain type, etc.…)

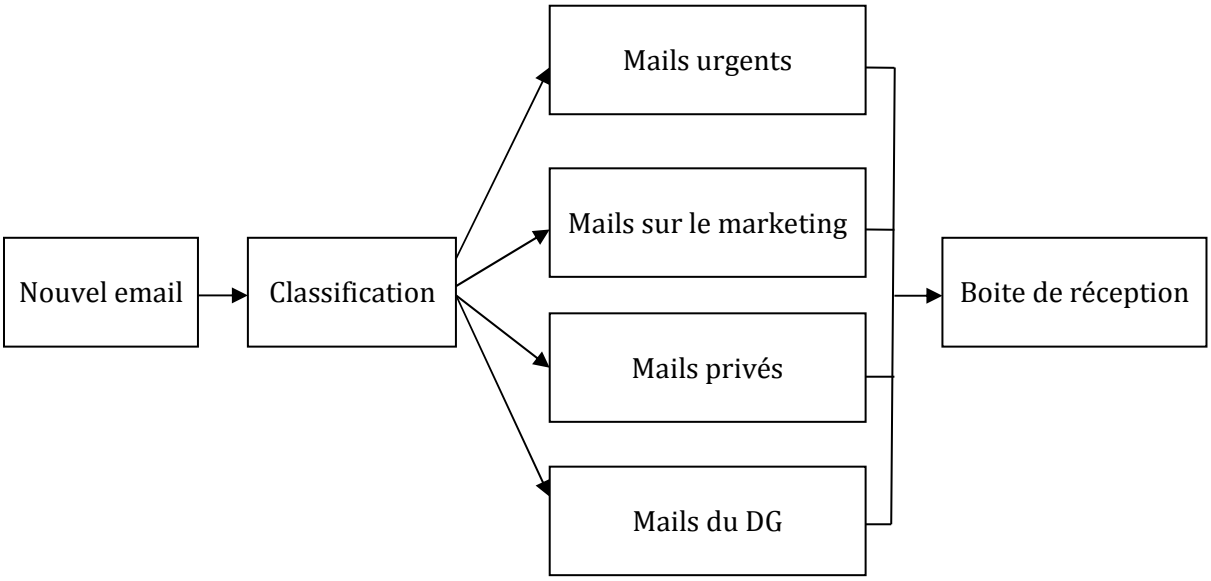


Figure I.1 : Exemple de système de classification d'emails

Ce système organise des emails dans des boîtes aux lettres qui correspondent chacune à une classe qui sont de différentes natures (« mails urgents », « Mails du Directeur général », etc.…).

4. Définitions

La classification est une discipline reliée de près ou de loin à plusieurs domaines, elle est connue aussi sous noms variés (classification, clustering, segmentation, . . .) selon les objets qu'elle traite et les objectifs qu'elle vise à atteindre.

Pour attribuer une définition au terme « classification », il faudrait d'abord définir ses racines, ça vient du verbe « classer » qui désigne plus une action qu'un domaine, ou plutôt une série de méthodes qu'une théorie unifiée.

En mathématique, On appelle classification, la catégorisation algorithmique d'objets. Elle consiste à attribuer une classe ou catégorie à chaque objet (ou individu) à classer, en se basant

sur des données statistiques. Elle fait couramment appel aux méthodes d'apprentissage et est largement utilisée en reconnaissance de formes.

D'une manière générale, la classification se définit alors comme une méthode mathématique d'analyse de données, pour faciliter l'étude d'une population d'effectif important, généralement des bases d'observations caractérisent un domaine particulier (animaux, plantes, malades, gènes, . . . etc.), où on les regroupe en plusieurs classes.

5. Définition formelle de la classification

Soient $C = \{c_1, c_2, \ldots, c_n\}$ un ensemble de classes et $D = \{d_1, d_2, \ldots, d_m\}$ un ensemble de documents à classer. On dira qu'un système de classification attribue automatiquement à chaque document un ensemble de classes : $0, 1$ ou plusieurs . Lorsqu'un document d_i appartient à une classe quelconque c_j, on dira que le document d_i est pertinent pour la classe c_j .Plusieurs formalisations du problème de classification ont été proposées dans la littérature par différents auteurs. On cite à titre indicatif la formalisation de Sebastiani **[4]** et reprise par Yang **[5]** qui semble très pratique. Pour cette formalisation on définit deux fonctions **[6]** :

- Une *fonction de décision* qui consiste à attribuer à chaque document un ensemble de classes.
- Une *fonction cible* indiquant la véritable appartenance d'un document à un ensemble de classes.

La fonction de décision permet d'estimer la *fonction cible*. Plus cette estimation est pertinente (correcte), meilleur est le système de classification. Ces deux fonctions associent à chaque couple $(d_i, c_j) \in D \times C$ une valeur booléenne indiquant si le document $d_i \in$ ou $\notin c_j$.

- La fonction de décision : $\varphi \rightarrow D \times C$ est telle que :

$$\varphi(di, cj) = \begin{cases} \text{vrai si di est attribué à la classe cj} \\ \text{faux sinon} \end{cases}$$

- La fonction cible : $\phi \rightarrow D \times C$ est telle que :

$$\phi(di, cj) = \begin{cases} \text{vrai si di appartient à la classe cj} \\ \text{faux sinon} \end{cases}$$

Habituellement avec les systèmes de classification basés sur les méthodes d'apprentissage, on évalue la fonction de décision en utilisant un corpus d'entraînement. Un corpus d'entrainement est une collection d'objets répartis en classes dont on connait à priori les noms (étiquettes) et le nombre. Un modèle d'apprentissage est caractérisé par trois phases principales dont :

- L'apprentissage (phase d'induction) qui consiste en l'élaboration du modèle sur un

corpus d'apprentissage dont on connaît la classification. Cela correspond à la fonction cible (training data).

- Le test et validation du modèle sur un nouvel échantillon test dont on connaît aussi la classification pour choisir le meilleur modèle (test data).
- L'application du classifieur (phase de déduction) aux documents à classer. Application de la fonction de décision par estimation de la fonction cible (application data).

6. Domaine d'application des techniques de classification

Les techniques de classification sont utilisées dans des nombreux domaines, comme :

- **Le domaine médical :** le chef d'un service hospitalier cherchera à regrouper des patients en sous-ensembles distincts en vue de définir la conduite thérapeutique là tenir devant un type particulier de malade.
- **Le domaine du marketing :** pour le lancement d'un nouveau produit, le responsable du service du marketing d'une entreprise pourra chercher à constituer des groupes de ville semblables vis-à-vis d'un certain nombre de critères, et choisir dans chaque groupe une ville type utilisée comme marché-test.
- **Le domaine politique :** un candidat aux élections municipales fixera sa stratégie électorale en fonction de déférents types d'électeurs construits à partir de différentes caractéristiques.
- **Le domaine de la publicité** : il s'agira d'adapter une campagne publicitaire à un groupe homogène d'individus (lecteurs de magazines, par exemple).

De nombreuses application citées dans la littérature dans le domaine de data Mining (fouille de données), Nous citons brièvement :

- La recherche documentaire (texte mining) **[7,8].**
- Analyse de données spatiales **[9,10]**.
- Application aux données de web **[11,12].**
- Analyse d'ADN dans la bi-informatique **[13].**
- Segmentation et traitement d'image **[14,15]**.

Dans tous ces cas, il s'agit de simplifier une réalité complexe pour laquelle aucune classification a priori ne s'emblée, en révélant des formes cachées et en isolant des points atypiques.

7. Contextes de classification

Plusieurs contextes de classification se distinguent, ils influents directement sur les modèles utilisés. Ludovic DENOYER a bien résumé les différents contextes de classification dans (Denoyer, 2004) que nous avons reporté dans ce qui suit :

7.1. La classification bi-classe

La classification bi-classe correspond au filtrage. C'est une problématique pour laquelle le système de classification répond à la question : « Le document appartient-il à la catégorie (classe) C ou non (i.e. ou à sa catégorie complémentaire ℸC ? » (Par exemple, un document est-il autorisé aux enfants ou non).

Cependant quand il s'agit d'effectuer une classification multi-classe qui permet de transmettre le document vers le ou les catégories(s) le(s) plus approprié(s), on parle alors de routage. Cette classification multi-classes, selon le cas, peut être disjointes ou non.

7.2. La classification multi-classes disjointes

La classification multi-classe disjointe correspond au contexte où nous sommes en présence de plusieurs classes (>1) et pour lequel un document appartient exactement à une seule classe. La classification à classes disjointes est la problématique pour laquelle le système de classification répond à la question .A quelle classe (au singulier) appartient tel document? **[16]**.

7.3. La Classification multi-classe

C'est le cas le plus général de la classification. Dans cette problématique le système de classification associe zéro (0), une (1) ou plusieurs (>1) classes à un document. Autrement dit, le système doit répondre à la question. Quelles sont les classes (au pluriel) auxquelles appartient le document ? On est dans un cas de classification non-exclusive.

8. Les méthodes de classification

Il existe deux grandes approches pour les méthodes de classification, l'approche ou classification « supervisée » et l'approche ou classification « non-supervisée ». Dans les deux cas, nous avons besoin généralement d'une mesure ou d'une fonctionnalité entre données pour prédire leur appartenance à une classe particulière. Ces deux approches comprennent une phase d'apprentissage ou de classification, cette phase correspond à la création de classes à partir d'exemples ayant des attributs connus; l'ensemble de ces classes constitue la base d'apprentissage. La classification supervisée dispose en plus d'une phase de décision ou de

généralisation, durant cette phase en présence d'une nouvelle donnée, le classifieur devra prédire l'appartenance de cette donnée à une classe de la base d'apprentissage.

La Figure 1.2 présente les différents types de méthodes, regroupés sous forme d'une hiérarchie par Jain et Dubes dans **[17]**.

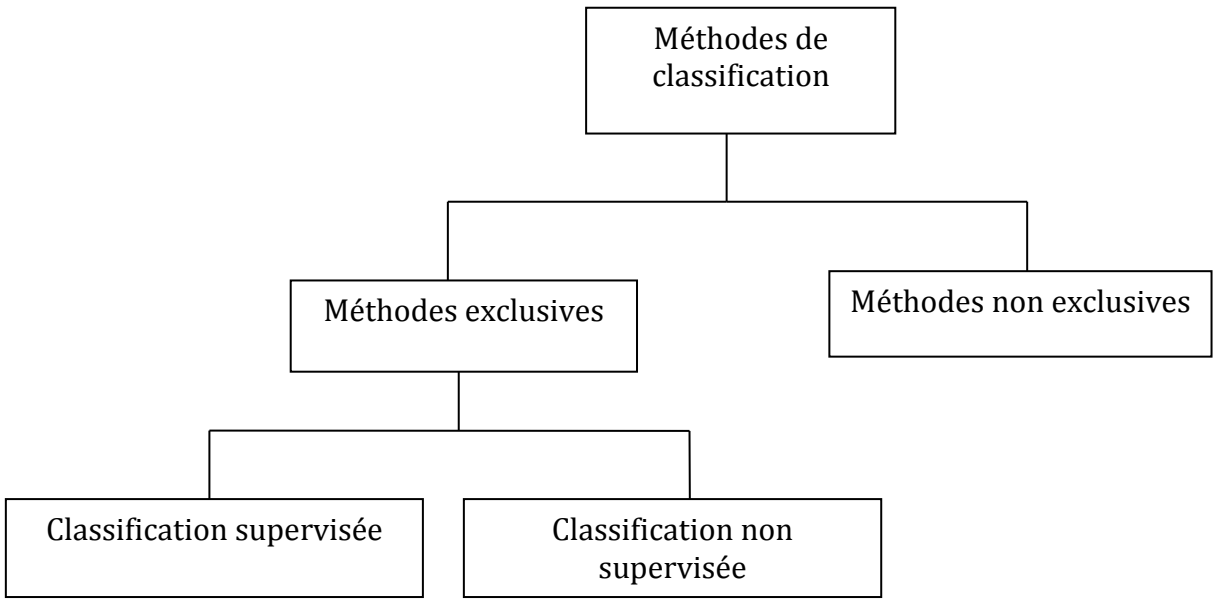


Figure I.2 : Différents types de méthodes de classification

8.1. Classification supervisée

Dans la classification supervisée, les classes sont connues a priori. Chaque classe dispose d'une sémantique bien particulière qui la différencie des autres classes et regroupe un ensemble de données étiquetées avec les attributs de la classe.

Dans ce type d'apprentissage, on cherche à estimer une fonction *f'(x)* qui est la relation entre les objets et leurs catégories donc un classificateur sera déterminé par la fonction *f(x)* qu'il implémente. Les objets utilisés comme données d'apprentissages sont accompagnés par la catégorie à laquelle ils appartiennent.

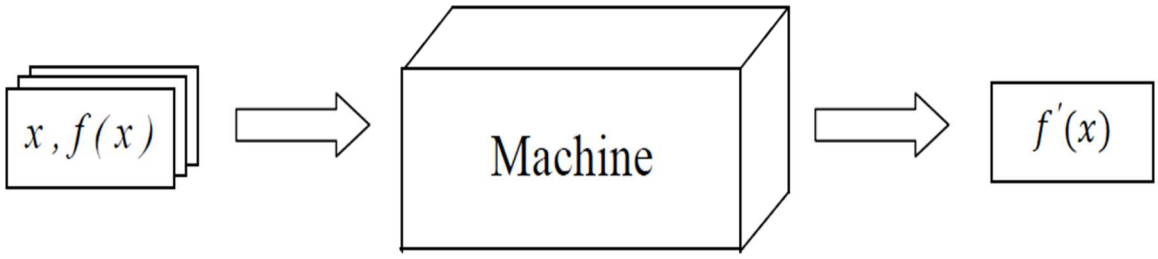


Figure I.3 : Schéma d'une machine Learning supervisé, *f'(x)* est une estimation de *f(x)*

Il existe de nombreuses techniques de classification supervisée **[18],** nous pouvons citer :

- Les Séparateurs à Vaste Marge (Support Vector Machine en Anglais).
- Les k plus proches voisins.
- Les réseaux bayésiens naïves.
- Les réseaux de neurons.
- Les arbres de decisions.

➢ Classification non-supervisée

Dans la classification non-supervisée, aussi appelée segmentation (clustering en anglais), les classes ne sont pas connues a priori. Elles sont construites à partir de certaines règles ou critères de regroupement qui dépendent des données disponibles à un moment donné. Les classes sont généralement fondées sur la structure des données, la sémantique associée à chaque classe est donc plus difficile à déterminer. On ne cherche pas cette fois à estimer une fonction mais on cherche à regrouper les objets ayant des caractéristique commune, les objets utilisés comme données d'apprentissage sont présentés sans leur catégories.

Il existe aussi de nombreuses techniques pour la classification non-supervisée, nous pouvons citer **[11]** :

- Les méthodes hiérarchiques : CURE, BIRCH, ROCK, Williames
- Les méthodes de partitionnement : K-means, PAM, CLARA
- Les méthodes basées sur la densité : DBSANS, OPTICS, DBCLASD
- Les méthodes basées sur les grilles : STING, CLIQUE, WAVECLUSTER

9. Etapes d'une classification

Le processus de classification passe par plusieurs étapes qui sont:

9.1 Choix des données

Il faut sélectionner les individus à classer et les variables qui serviront pour critère de classification. Si par exemple on veut faire une classification selon le milieu de résidence, il faut stratifier le fichier initial en deux sous-fichiers.

On sélectionne les individus qui résident en milieu urbain et ceux en milieu rural puis lancer la classification sur chacun de ces deux sous-fichiers.

9.2 Calcul des similarités entre les n individus à partir des données initiales

On choisit une distance ou un indice d'écart entre paires d'individus. La distance généralement utilisée dans les algorithmes de classification hiérarchique est la distance euclidienne.

9.3 Choix d'un algorithme de classification et exécution

Choisir l'algorithme suivant les exigences (performances, représentation des résultants, types de données) et fixer les éventuels paramètres.

9.4 L'interprétation des résultats

- Évaluation de la qualité de la classification,
- Description des classes obtenues.

10. Mesures de similarité et formules pour calcul de distance

Les données soumises à une technique de classification se présentent le plus souvent sous la forme d'un tableau rectangulaire avec en lignes les individus (objets, instances, etc.), et en colonnes les variables (attributs, caractère).

Ce tableau peut être un tableau de données numériques (quantitatives ou continus), un tableau de contingence (croisant deux partitions d'une même population ou, plus généralement un tableau de données mixtes (binaires, nominales, ordinales et continus).

Pratiquement, il s'agit de décrire les liaisons entre variables ou les similarités entre individus. Mais en général quand on parle de classification, il s'agit le plus souvent de recherche d'une partition d'un ensemble d'individus, lignes d'un tableau (n, p), en groupes homogènes et distincts.

Les mesures de ressemblance entre objets à classer dépendant de la nature des variables mesurées qui peuvent être binaires (symétrique ou asymétriques si les deux modalités n'ont pas le même poids), nominales (catégorielles à k modalités), ordinales ou numériques (mesurées dans une échelle linéaire ou non).

10.1. Distances et indices de similarité

10.1.1 Distance définie sur un ensemble E

C'est une application de du produit cartésien E×E dans R^+ satisfaisant aux axiomes suivants **[12]** :

Symétrie : $d(x,y) = d(y,x), \forall x \in E, \forall y \in E$ (1.1)

Positivité stricte : $d(x,y) > 0\ six \neq yetd(x,y) = 0 \leftrightarrow six = y, \forall x \in E, \forall y \in E$ (1.2)

Inégalité triangulaire : $d(x,y) \leq d(x,y) + d(x,z) + d(z,y), \forall x \in E, \forall y \in Eet\ \forall z \in E$ (1.3)

10.1.2 Similarité définie sur un ensemble E

C'est une application de du produit cartésien E×E dans R+ satisfaisant aux axiomes suivants [12] :

$s(x,y) = s(y,x) \geq 0 \qquad \forall x \in E, \forall y \in E$ (1.4)

$s(x,y) = s(y,y) = s_{max} > s(x,y) \qquad \forall x \in E, \forall y \in E$ (1.5)

Où $\boldsymbol{s_{max}}$ est La plus grande similarité possible.

10.1.3 Dissimilarité définie sur un ensemble E

Elle est définie à partir un indice de similarité : $dis(x,y) = 1 - s(x,y)$ (1.6)

Remarque : Une dissimilarité est une distance qui ne vérifie pas l'irrégularité triangulaire.

10.2 Mesures de ressemblance entre individus

Les deux grands types de méthodes de classification utilisent différentes distances et indices de similarité [Jain et Dubes, 1988].

- **Données numérique**: dans ce type d'attributs trois fonctions de distances plus courantes peuvent être calculées entre deux objets (x et y) : la distance Euclidienne, la distance de Chebychev et la distance de Manhatta. Ces distances sont un cas particulier d'une fonction de distance plus générale appelée **la distance de Minkowski** dénotée par d(x, y) :

$$d(x,y) = \left[\sum_{i=1}^{p} |x_i - y_i|^q\right]^{\frac{1}{q}} \qquad (1.7)$$

Où q est un entier positif qui peut être :

q=2, la distance calculée entre deux objets est la distance Euclidienne

$$d(x,y) = \sqrt{\sum_{i=1}^{p}(x_i \quad - y_i)^2} \qquad (1.8)$$

q=1, la distance calculée entre deux objets est la distance de Manhattan :

$$d(x,y) = \sum_{i=1}^{p}|x_i \quad - y_i| \qquad (1.9)$$

q= n, la distance calculée entre deux objets est la distance de Chebychev :

$$d(x,y) = max\{|x_i \quad - y_i|\} \qquad i = 1, \ldots\ldots, p \qquad (1.10)$$

❖ **Attributs binaires** : les fonctions de distance existantes pour les attributs binaires sont basées sur la proportion de correspondance dans les deux objets. On distingue deux types d'attributs :

- Les attributs binaires symétriques qui représentent deux états qui ont une importance égale et portent le même poids. La fonction de distances la plus utilisée est la distance de correspondance simple qui est une proportion de non correspondances de leurs valeurs définies comme :

$$d(x,y) = \frac{\alpha+\delta}{n} \qquad (1.11)$$

- Les attributs asymétriques dans lesquels un des états est plus important ou utile que les autres. La mesure de distance la plus utilisée pour les attributs asymétriques est la distance de Jaccard définie comme :

$$d(x,y) = \frac{\beta+\gamma}{\alpha+\beta+\gamma} \qquad (1.12)$$

Où :

- α: représente le nombre de valeurs d'attributs égales à 1 dans les deux objets *x* et *y*.
- β: représente le nombre de valeurs d'attributs égales à 1 dans l'objet x mais 0 dans l'objet *y*.
- γ: représente le nombre de valeurs d'attributs égales à 0 dans l'objet *x* mais 1 dans l'objet *y*.
- δ : représente le nombre de valeurs d'attributs égales à 0 dans les deux objets *x* et *y*.
- η : représente le nombre d'attributs. Chaque paire d'attributs doit nécessairement appartenir à l'une des quatre catégories de telle sorte que :

$\alpha + \beta + \gamma + \delta = \eta$

- **Des attributes nominaux:**

$$d(x,y) = \frac{p-m}{p} \quad ; \quad m = a + d \qquad (1.13)$$

Tels que m est le nombre d'attributs dont la valeur est pareil dans tous les deux objets, p est le nombre total d'attributs.

11. Critères pour une bonne classification

L'objectif principal des techniques de classification est de trouver une partition ou les objets d'une classe devraient être semblables (entre eux), les objets de différentes classes devraient être différents. Une bonne classification devrait accomplir différents critères :

- **Validité :** Elle peut se définir par :

Chaque classe d'une partition doit être homogène : Les objets qui appartiennent à la même classe doivent être semblables. Les classes doivent être isolées entre elles : Les objets de différentes classes doivent être différents. La classification doit s'adapter aux données : La classification doit pouvoir expliquer la variation des données **[19]**.

- **Interopérabilité :** Les classes doivent avoir une interprétation substantive c'est-à-dire qu'il est possible de donner des noms aux classes, dans le meilleur des cas les noms doivent correspondre aux types déduits d'une certaine théorie **[19]**.

- **Stabilité :** Les classes doivent être stable ça veut dire que les petites modifications dans les données et dans les méthodes ne doivent pas changer les résultats **[19].**

- **D'autres critères :** Parfois la taille et le nombre de classes sont employés en tant que critères additionnels: le nombre de classes doit être aussi petit que possible, et la taille des classes ne doit pas être trop petite [**19]**.

12. Conclusion

La classification a conquis aujourd'hui le centre d'intérêt des chercheurs. Plusieurs travaux ont été faits, dans lesquels ils visent à inventer et améliorer des méthodes de classification de plus en plus performantes ; Toujours pour améliorer la classification et diminuer la complexité temporelle de ces méthodes. Avec le progrès rapide sur les matériels d'informatique, le deuxième avantage semble beaucoup moins important maintenant et la qualité et la justesse de la classification devient le but ultime.

Dans ce chapitre le principe de la classification a été présenté, ainsi que les méthodes utilisées pour évaluer la qualité de la classification. Dans les prochains chapitres (II et III) nous allons présenter les deux types de classifications (respectivement la classification supervisée et non-supervisée).

1. Introduction

La classification supervisée est une tâche largement appliquée dans la vie courante. En effet, il existe une multitude de problèmes qui entrent dans ce cadre, parmi lesquels on trouve la reconnaissance des caractères manuscrits, la reconnaissance des paroles, la catégorisation des textes, la détection des spam, l'aide au diagnostic médical, la bio-informatique…etc.

A la différence de la classification non supervisée où les groupes d'objets sont découverts à posteriori, les techniques de classification supervisée s'appliquent lorsqu'on veut rattacher un nouvel objet (observation) à une classe qui est choisie parmi un ensemble de classes connues. Cette étape suit, généralement, l'étape de clustering.

La plupart des algorithmes de classification tentent de trouver un modèle (une fonction mathématique) qui explique le lien entre les données d'entrée et les classes de sortie. Un ensemble d'apprentissage (ensemble de classes) est donc utilisé par l'algorithme. Cette méthode de raisonnement est appelée inductive car on induit la connaissance (le modèle) à partir des données d'entrée (objets à classifier) et des sorties (leurs classes). Grâce à ce modèle, on peut alors prédire les classes de nouvelles données. Le modèle est bon s'il permet de bien prédire. Un exemple de cette catégorie d'algorithmes est le réseau de neurones.

Une autre approche qui n'est pas moins intéressante, c'est le raisonnement à partir des cas. Ces algorithmes ne cherchent pas à calculer le modèle mais à trouver, pour l'objet à classifier, un ou plusieurs cas similaires déjà résolus pour en déduire la classe. Les arbres de décision et l'algorithme k-NN adoptent ce principe.

Dans ce qui suit, nous allons décrire et détailler les principaux algorithmes de classification automatique supervisée.

2. Problématique

On dispose d'un ensemble *X*, comportant *N* données étiquetées (dont la classe est connue). Chaque donnée *x* est caractérisée, par *P* attributs et par sa classe $Ci \in C$, *C* étant l'ensemble des classes.

Le problème consiste alors, en s'appuyant sur l'ensemble $X = \{(xi; Ci); i \in \{1..N\}\}$, à prédire la classe de toute nouvelle donnée *x*.

3. Définition

La classification supervisée (catégorisation**)** vise à classer des objets selon des catégories bien définies au préalable.

Dans ce type de classification, les classes sont prédéfinies avec une description des données. Lorsqu'une nouvelle donnée arrive, on la compare avec la description de chaque classe et on la met dans celle qui lui ressemble le plus.

4. Algorithmes de classification supervisée

Parmi les algorithmes de classification supervisée les plus populaires [20], on trouve :

- L'algorithme des K plus proches voisins (ou K-NN).
- Machines à support de vecteurs (ou SVM).
- Les arbres de décision.
- Les réseaux de neurones (RNA).
- L'algorithme de Naïve Bayes.

4.1. Méthode de k plus proches voisins

La méthode des plus proches voisins (note parfois **k-PPV** ou **k-NN** pour **Nearest-Neighbor**) est une méthode supervisée et non-paramétrique. Son fonctionnement peut être assimilé à l'analogie suivante *"dis-moi qui sont tes voisins, je te dirais qui tu es"*.

Le principe général de la méthode des k-PPV consiste à déterminer pour chaque nouvel individu que l'on veut classer, la liste des plus proches voisins parmi les individus déjà classés. Autrement dit, l'objectif de l'algorithme est de classer les exemples non étiquetés sur la base de leurs similarités avec les exemples de la base d'apprentissage.

Cette méthode nécessite de choisir une distance, la plus classique est la distance euclidienne, et le nombre de voisins à prendre en compte.

Le principe de cet algorithme est très simple. On lui fournit :

- Un ensemble de données d'apprentissage **D** ;
- Une fonction de distance **d** ;
- Et un entier **k.**

Pour tout nouveau point de test **x,** pour lequel il doit prendre une décision, l'algorithme recherche dans **D** les **k** points les plus proches de **x** au sens de la distance **d**, et attribue **x** à la classe qui est la plus fréquente parmi ces **k** voisins.

➢ Comment choisir la valeur de k ?

Le choix de 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, 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 du overfitting et par conséquent un modèle qui se généralise mal sur des observations qu'il n'a pas encore vu.

➢ Algorithme k plus proches voisins

On peut schématiser le fonctionnement de K-NN par l'algorithme suivant :

Algorithme k plus proches voisins

Début Algorithme
Données en entrée :

- Un ensemble de données D.
- Une fonction de définition distance d.
- Un nombre entier k

Pour une nouvelle observation x dont on veut prédire sa variable de sortie y Faire :

1. Calculer toutes les distances de cette observation x avec les autres observations du jeu de données D
2. Retenir les k observations du jeu de données D les proches de x en utilisation le fonction de calcul de distance
3. Prendre les valeurs de y des k observations retenues :
   Calculer le mode de y retenues
4. Retourner la valeur calculée dans l'étape 3 comme étant la valeur qui a été prédite par K-NN pour l'observation .

Fin Algorithme

Exemple

Soit l'exemple de la **figure II.1** avec deux dimensions correspondant aux attributs *e1* et *e2*, et avec *k=3.*

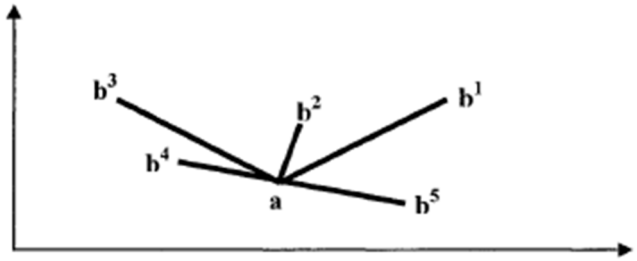


Figure II.1 : Méthode des 3-pvv

Dans cet exemple les trois plus proches voisins de *a* sont *b4, b2* et *b5,* donc *a* sera affecté à la classe majoritaire parmi ces trois points.

- **Critiques de la méthode:**
  - **Avantage de la méthodes k-PPV**

- Apprentissage rapide.
- Méthode facile à comprendre.
- Adapté aux domaines où chaque classe est représentée par plusieurs prototypes et où les frontières sont irrégulières (Exemple : reconnaissance de chiffre manuscrit ou d'images satellites).

  - **Inconvénients de la méthodes k-PPV**

- Prédiction lente car il faut revoir tous les exemples à chaque fois.
- Méthode gourmande en place mémoire.
- Sensible aux attributs non pertinents et corrélés.

4.2. Séparateurs à Vaste Marge (SVM)

Les machines à support de vecteurs *(SVM)* sont à l'origine des nouvelles méthodes de catégorisations, bien que les premières publications sur le sujet datent des années 60.
Avant d'aborder le principe de fonctionnement général des *SVM* voici quelques notions de base :

- ***Hyperplan* :** est un séparateur d'objets des classes. De cette notion, nous pouvons dire qu'il est évident de trouver une mainte d'hyperplans mais la propriété délicate des SVM est d'avoir l'hyperplan dont la *distance minimale* aux exemples d'apprentissage est maximale, cet hyperplan est appelé L'*hyperplan optimal*, et la distance appelée *marge*.

- ***Vecteurs Support* :** ce sont les points qui déterminent l'hyperplan tels qu'ils soient les plus proches de ce dernier.

Voici un schéma représentatif de ces notions:

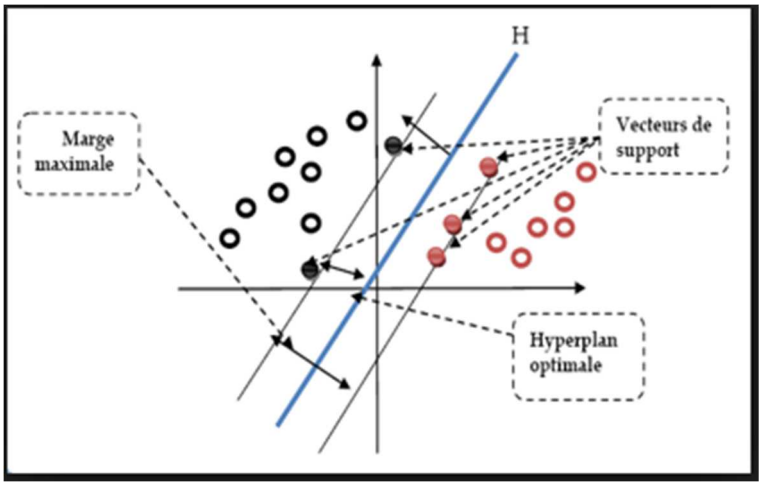


Figure II.2 : Les vecteurs à support

➢ Principe de la technique SVM

Cette technique est une méthode de classification à deux classes qui tente de séparer les exemples positifs des exemples négatifs dans l'ensemble des exemples. La méthode cherche alors l'hyperplan qui sépare les exemples positifs des exemples négatifs, en garantissant que la marge entre le plus proche des positifs et des négatifs soit maximale. Cela garantit une généralisation du principe car de nouveaux exemples pourront ne pas être trop similaires à ceux utilisés pour trouver l'hyperplan mais être situés d'un côté ou l'autre de la frontière.

L'intérêt de cette méthode est la sélection de vecteurs supports qui représentent les vecteurs discriminant grâce auxquels est déterminé l'hyperplan. Les exemples utilisés lors de la recherche de l'hyperplan ne sont alors plus utiles et seuls ces vecteurs supports sont utilisés pour classer un nouveau cas, ce qui peut être considéré comme un avantage pour cette méthode **[21].**

➢ Exemple d'application [21]

- Classification des données biologiques/ physiques.
- Classification des documents numériques.
- Classification d'expression faciale.

➢ Critiques de la méthode [21]

➢ Avantages des SVM

Les avantages théoriques et pratiques des SVM en ont fait un outil très prisé dans nombreux problèmes de classification. On cite parmi ces avantages:

- Minimisation de l'erreur empirique et structurelle.
- Algorithms optimisés.

- Simple, peu de paramètres à régler.
- Les SVM possèdent des fondements mathématiques solides.

➢ Inconvenients des SVM

- Les données dans des espaces de grande dimension sont souvent non linéairement séparable.
- Classification binaire, d'où la nécessité d'utiliser l'approche un-contre-un.
- Grande quantité d'exemples en entrées implique un calcul matriciel important.

4.3. Les arbres de décision :

Un arbre de décision est, comme son nom l'indique, un outil d'aide à la décision qui permet de classifier une population d'individus selon les valeurs de leurs attributs. C'est une représentation graphique de la procédure de classification où :

- Une feuille indique une classe.
- Un nœud spécifie un test que doit subir un certain attribut.
- Chaque branche sortant de ce nœud correspond à une valeur possible de l'attribut en question (Figure II.3).

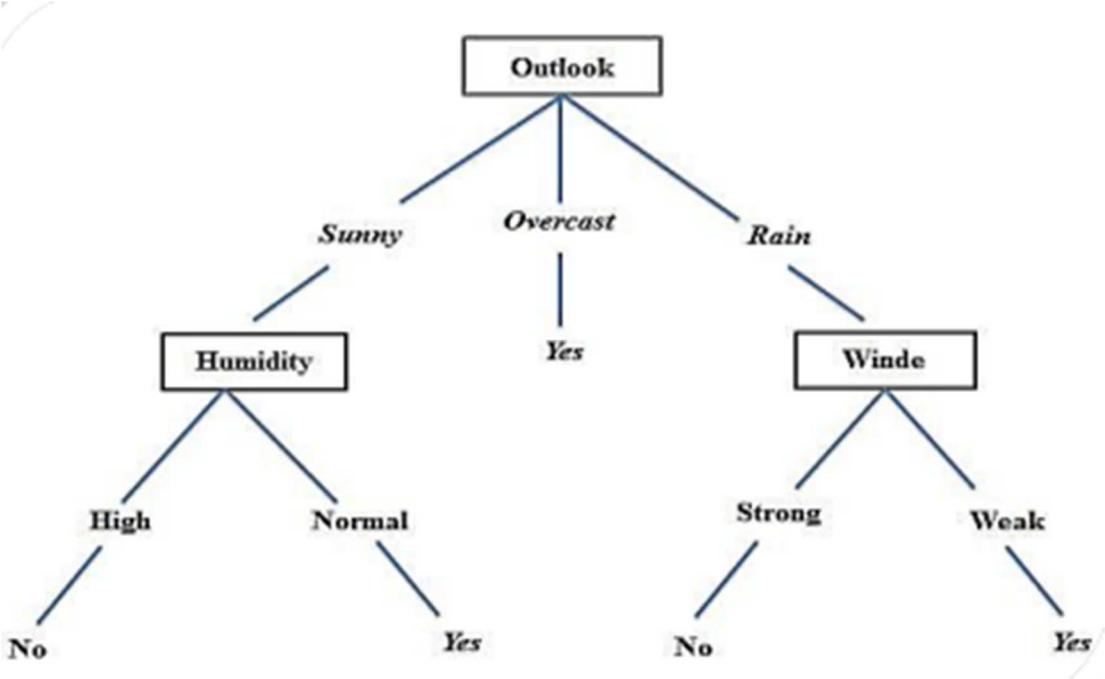


Figure II.3: l'arbre de décision

Pour classifier un nouvel objet, on suit le chemin partant de la racine (noeud initial) à une feuille en effectuant les différents tests d'attributs à chaque noeud. L'arbre permet d'émettre des prédictions sur les données par réduction, niveau par niveau, du domaine de solutions.

La démarche générale de construction de l'arbre de décision consiste en deux étapes:

- Construction de l'arbre à partir des données (apprentissage).
- Elagage de l'arbre dans le but d'alléger l'arbre résultant souvent volumineux.

➢ *Construction de l'arbre*

Il existe une grande variété d'algorithmes pour construire des arbres de décision ; quelques-uns des plus répandus portent les noms de ID3 (*Inductive Decision-tree*) introduit par Quinlan et amélioré pour devenir C4.5 [22], CART (*Classification And Regression Trees),* introduit par Breiman et al [23] et CHAID (*Chi-squared Automatic Interaction Detection*) [24].

Le principe général démarre d'un arbre vide et procède à la construction de l'arbre de manière inductive (à partir des données) et récursive en commençant par l'ensemble des objets tout entier. Si tous les objets sont de même classes, une feuille est créée avec le nom de la classe. Sinon, l'ensemble d'objets est partagé en sous-ensembles selon la valeur d'un certain attribut qui subissent le même traitement.

Afin d'avoir un arbre de décision concis et suffisant, il ne suffit pas de traiter les attributs séquentiellement. Au contraire, toute la richesse des arbres de décision consiste à choisir judicieusement les attributs d'éclatement pour aboutir, par le chemin le plus court et nécessaire, au plus grand nombre d'objets de la même classe.

Le choix des attributs peut se faire par plusieurs techniques, entre autres :

- Entropie (ID3, C4.5) [22].
- Indice de Gini (CART) [22].
- Table de Khi-2 (CHAID) [20].

➢ *Elagage de l'arbre*

L'opération d'élagage de l'arbre se fait en deux phases : Le pré-élagage et le post-élagage.

Le pré-élagage consiste à fixer un critère d'arrêt qui permet de stopper la construction de l'arbre lors de la phase de construction.

Le post-élagage est un traitement qui intervient après la construction entière de l'arbre. Il consiste à supprimer les sous-arbres qui n'améliorent pas l'erreur de classification.

➢ Les domains d'application

Cette méthode peut être utilisée dans plusieurs domains tels que:

Les études (pour comprendre les critères prépondérants dans l'achat d'un produit, l'impact des dépenses publicitaires), les ventes (pour analyser les performances par région, par enseigne, par vendeur), l'analyse de risques (pour détecter les facteurs prédictifs d'un comportement de non-paiement), Le domaine médical (pour étudier les rapports existant entre certaines maladies et des particularités physiologiques ou sociologiques) **[25].**

➢ Critiques de la méthode

➢ Les avantages

Les arbres de décision constituent un moyen très efficace de classification, et ce pour les avantages qu'elles présentent. Parmi ses avantages, on peut citer **[23] [26]** :

- Facilité à manipuler des données catégoriques.
- Traitement facile des variables d'amplitudes très différentes.
- La classe associée à chaque individu peut être justifiée.
- Les attributs apparaissant dans l'arbre sont des attributs pertinents.
- Pour le problème de classification considéré.

➢ Les Inconvénients

Ces méthodes présentent tout de même des inconvénients dont les plus importants sont :

- La sensibilité au bruit et aux points aberrants.
- La sensibilité au nombre de classes (plus le nombre de classes est grand plus les performances diminuent).
- Le besoin de refaire l'apprentissage si les données évoluent dans le temps.

4.4. Réseaux de neurones

Les réseaux de neurones sont à l'origine d'une tentative de modélisation mathématique du cerveau humain. Le principe général consiste à définir des unités simples appelées neurones, chacune étant capable de réaliser quelques calculs élémentaires sur des données numériques.

On relie ensuite un nombre important de ces unités formant ainsi un outil de calcul puissant.

Un réseau de neurones est un système composé de plusieurs unités de calcul simples (nœuds) fonctionnant en parallèle, dont la fonction est déterminée par la structure du réseau et l'opération effectuée par les nœuds.

➢ Principe de la méthode

Le principe de fonctionnement est le suivant : On dispose initialement d'une base de connaissances constituée de couples de données (entrées / sorties) et on souhaite utiliser cette base de données pour entrainer un algorithme à reproduire les associations constatées entre les entrées et les sorties de l'échantillon.

L'exemple le plus simple de réseau de neurones est souvent donné par le “perceptron multicouches’’ qui est un cas particulier de réseau de neurones (**Figure II.4**).

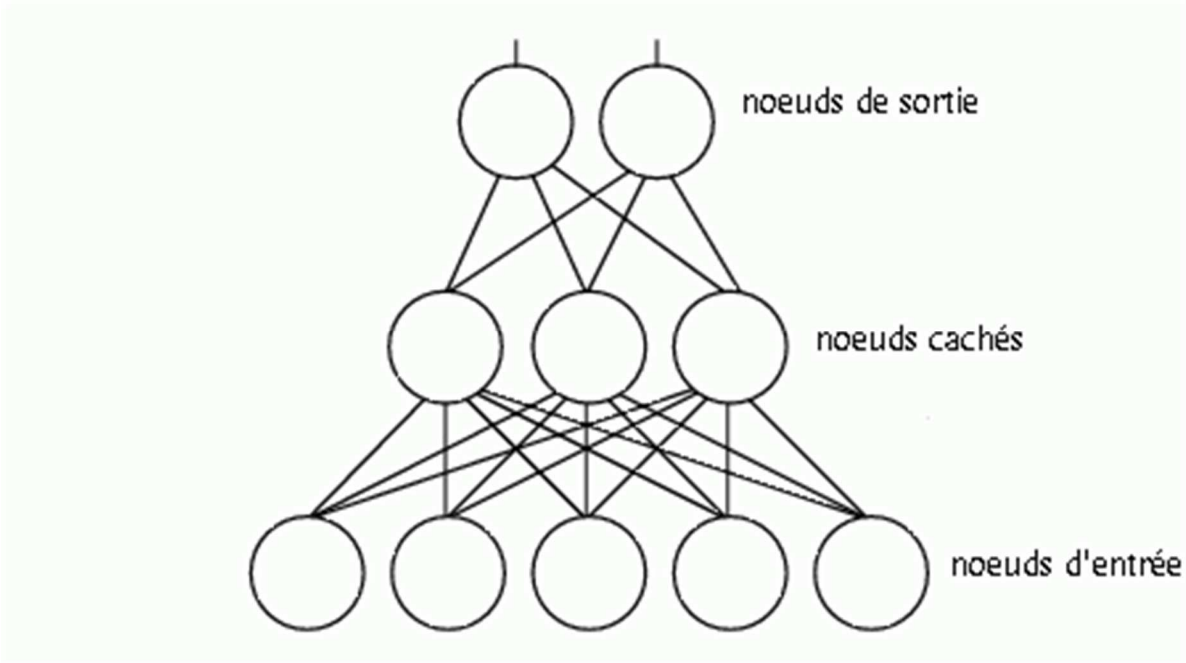


Figure II.4: Perceptron à trois couches

Pour un réseau de neurones avec N nœuds d’entrée, notés C(1), ..., C(N), et N poids affectées aux liaisons notés w(1), ...,w(N) l'entrée d’un nœud de la couche suivante sera généralement une somme pondérée des valeurs de sortie des neurones précédents :

$X = w(1)*C(1) + w(2)*C(2) + w(3)*C(3) + ... + w(N)*C(N)$

Les poids sont des paramètres adaptatifs, dont la valeur est à déterminer en fonction du problème via un algorithme d’apprentissage (propagation, rétro-propagation…) **[27].**

Les réseaux de neurones peuvent être utilisés pour effectuer une classification supervisée floue de la manière suivante : chaque nœud d'entrée correspond à un attribut de l'objet (autant de nœuds d'entrée que d'attributs).

On peut prendre un neurone de sortie par classe; la valeur de sortie est la valeur de la fonction d'appartenance (probabilité que l'objet appartienne à cette classe) **[28]**.

- **Critiques de la méthode**
  - **Avantages**

- Classification très bien précise (si bien paramétré).
- Résistance aux pannes (si un neurone ne fonctionne plus, le réseau ne se perturbe pas).

  - **Inconvénients**

- La détermination de l'architecture du réseau est complexe.
- Paramètres difficiles à interpréter (boite noire).
- Difficulté de paramétrage surtout pour le nombre de neurone dans la couche cachée.

4.5. Classification naïve bayésienne

La classification naïve bayésienne est un type de classification Bayésienne probabiliste simple basée sur le théorème de Bayes avec une forte indépendance (dite naïve) des hypothèses. Elle met en œuvre un classifieur bayésienne naïf, ou classifieur naïf de Bayes, appartenant à la famille des classifieurs Linéaires.

Un terme plus approprié pour le modèle probabiliste sous-jacent pourrait être « modèle à Caractéristiques statistiquement indépendantes ».

En termes simples, un classifieur bayésien naïf suppose que l'existence d'une caractéristique pour une classe, est indépendante de l'existence d'autres caractéristiques. Un fruit peut être considéré comme une pomme s'il est rouge, arrondi, et fait une dizaine de centimètres. Même si ces caractéristiques sont liées dans la réalité, un classifieur bayésien naïf déterminera que le fruit est une pomme en considérant indépendamment ces caractéristiques de couleur, de forme et de taille.

Selon la nature de chaque modèle probabiliste, les classifieurs bayésiens naïfs peuvent être entraînés efficacement dans un contexte d'apprentissage supervisé.

Ce classificateur se base sur le théorème de Bayes permettant de calculer les probabilités conditionnelles. Dans un contexte général, ce théorème fournit une façon de calculer la probabilité conditionnelle d'une cause sachant la présence d'un effet, à partir de la probabilité

conditionnelle de l'effet sachant la présence de la cause ainsi que des probabilités a priori de la cause et de l'effet.

- **Critiques de la méthode**
  - **Avantages**
    - La facilité et la simplicité de leur implémentation.
    - Leur rapidité.
    - Les méthodes Naïve Bayes donnent de bons résultats.
  - **Inconvénients**
    - Performances limitées quand il s'agit d'une grande quantité à traiter.
    - Modèle qualifié de naïve ou simple à cause de l'hypothèse d'indépendance.

5. Performance des méthodes de classification

La plupart des méthodes de classification mentionnées dans les chapitres précédents ont été largement appliquées dans plusieurs domaines. La question qu'on peut se poser est la suivante: comment évaluer les performances d'une méthode de classification?

En général, on divise l'ensemble de données disponibles en deux sous-ensembles: l'un servira pour l'apprentissage et l'autre pour le test.

L'ensemble d'apprentissage est utilisé pour déterminer les paramètres du modèle de classification, par exemple les poids dans le cas d'un réseau de neurones ou les prototypes des catégories dans le cas des méthodes de classification multicritère.

L'ensemble de test sert pour tester les performances de la méthode en calculant le taux de classification correcte de l'ensemble des cas. Ce taux est déterminé en divisant le nombre de cas bien classés sur le nombre des cas testés. Parfois on est confronté à des problèmes où l'ensemble de données est restreint et on veut exploiter ces données disponibles pour construire le classificateur d'une part et tester les performances de la méthode d'autre part. Pour cela on fait appel aux techniques de rééchantillonage (resampling techniques) ; parmi lesquelles la technique de validation croisée (cross-validation) est la plus utilisée.

Le principe de cette technique de validation croisée (cross-validation) consiste à diviser aléatoirement l'ensemble des données en *m* partitions mutuellement exclusives (m-fold cross validation). En suite la méthode est construite à partir de l'ensemble des partitions moins une qui servira de test. Après on réitère le processus en introduisant la partition testée dans

l'ensemble d'apprentissage et en prenant une autre partition d'apprentissage pour tester la méthode et ainsi de suite jusqu'à ce que toutes les données seront utilisées tantôt pour l'apprentissage et tantôt pour le test. La moyenne des taux de classification correcte sur toutes les partitions de test correspond au taux de prédiction **[29]**.

Plusieurs critères pour évaluer la performance d'un classifieur, ont été proposés :

5.1. Critères d'évaluation du classificateur [30]

Diverses façons existantes aujourd'hui ont pour objectif de comparer les décisions prises par le classificateur automatique à celles des experts humains et de calculer un score de performance :

Pour mieux illustrer ces différentes mesures on prend pour point de départ la table de contingence illustrée par le (**Tableau 2.3)**.

5.1.1. Matrice de confusion

Soit une tâche de classification supervisée où la variable à prédire a un nombre quelconque de modalités, $y \in \{1,2,\dots,q\}$. L'évaluation d'un classifieur se base sur la matrice de confusion, que l'on construit à partir des prédictions qu'il réalise sur un ensemble d'individus pour lesquels on connaît la classe d'appartenance réelle. Ainsi, la matrice de confusion, décrit la distribution conjointe des variables y et y' :

	$y = 1$	$y = 2$	….	$y = q$
$y' = 1$	$m_{1,1}$	$m_{1,2}$	…	$m_{1,q}$
$y' = 2$	$m_{2.1}$	$m_{2,2}$	…	$m_{2,q}$
…	…	…	…	…
$y' = q$	$m_{q,1}$	$m_{q,2}$	…	$m_{q,q}$

Tableau II.1 : Matrice de confusion

Si le classifieur ne commet aucune erreur, c'est-à-dire qu'il prédit toujours la classe réelle, alors *M* est une matrice diagonale (ce qui signifie que les coefficients en dehors de la diagonale sont tous nuls). Lorsque le classifieur se trompe, les effectifs des couples $(y = a, y' = b)$ tels que $a \neq b$ indiquent comment les erreurs comises se répartissent.

Dans le cas d'une tâche de classification supervisée binaire, $\in \{0,1\}$, où la modalité 1 de la variable à prédire correpond à la classe «postive» et l'autre à la classe «négative», on nomme les coefficients de la matrice de confusion de la manière suivante :

- VN : Nombre de *vrais négatifs.*
- FN : Nombre de *faux négatifs.*
- FP : Nombre de *faux positifs.*
- VP : Nombre de *vrais positifs.*

	$y = 0$	$y = 1$
$y' = 0$	VN	FN
$y' = 1$	FP	VP

Tableau II.2 Matrice de confusion classification supervisée binaire

5.1.2. Mesures d'évaluation [30]

On définit plusieurs mesures basées sur la matrice de confusion, afin de quantifier la performance d'un classifieur selon différents points de vue :

- Taux d'erreur global.
- Précision par classe, précision moyenne.
- Rappel par classe, rappel moyenne.
- F-mesure par classe, f-mesure moyenne.

➢ Taux d'erreur global

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

$$E = 1 - \frac{T_r(M)}{\sum_{i,j} M_{i,j}}$$

➢ Précision par rapport à une classe

La précision d'un classifieur par rapport à une certaine classe (autrement dit, par rapport à une certaine modalité de la variable à prédire), se mesure comme la proportion d'individus, parmi tous ceux pour lesquels le classifieur a prédit cette classe, qui appartiennent réellement à celle-ci. Pour la classe *c*, on calcule la précision comme suit (le nombre d'individus pour lesquels le classifieur a prédit la classe *c* correspond à l'effectif marginal pour *y'=c*) :

$$P_c = \frac{m_{c,c}}{m_{c,\blacksquare}}$$

➢ Rappel par rapport à une classe

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

$$R_c = \frac{m_{c,c}}{m_{\blacksquare,c}}$$

➢ F-mesure par rapport à une classe

On peut résumer les mesures de précision de rappel par rapport à une classe en un seul indicateur, en calculant la moyenne harmonique :

$$F_c = \frac{P_c \times R_c}{P_c + R_c}$$

➢ Précision moyenne, rappel moyen, f-mesure moyenne

On peut synthétiser les mesures de précision en calculant la moyenne pondérée par le nombre d'individus appartenant à chaque classe :

$$P_{moyenne} = \sum_c m_{\blacksquare,c} \; R_c$$

De même pour les mesures de rappel :

$$R_{moyenne} = \sum_c m_{\blacksquare,c} \; R_c$$

Et pour les f-mesures :

$$F_{moyenne} = \sum_c m_{\blacksquare,c} \; F_c$$

6. Conclusion

Nous avons vu une généralité sur les conceptions des méthodes de classification et un aperçu sur les principes de la première grande approche qui infère à partir d'un échantillon d'exemples classés une procédure (fonction de décision) de classification des nouveaux exemples non étiquetés. La Discrimination (ou les méthodes supervisées) peut être basée sur des hypothèses probabilistes (Classifieur naïf de Bayes, méthodes paramétriques) ou sur des notions de proximité (plus proches voisins) ou bien encore sur des recherché dans des espaces d'hypothèses (arbres de décision, réseaux de neurones).

Certes l'approche supervisée est très utilisée pour les raison et les avantages qu'on a mentionné pour chaque méthode , néanmoins il reste qu'il y a un manque de stratégies pour les exemples d'auto-apprentissage (c'est-à-dire , d'apprendre à partir d'une base sans aucune connaissance préalable) que les méthodes supervisée ne peuvent pas traiter , dans ce cadre vient la deuxième approche des méthodes de classification, qui est : l'approche non-supervisée (la classification automatique)

1. Introduction

Comme nous avons pu le voir dans le premier chapitre qu'il y a deux grandes approches en classification : la classification supervisée (classement) et la classification automatique (clustering), dans ce chapitre nous détaillerons les méthodes du deuxième type « clustering » qui est une des techniques statistiques largement utilisées dans la Fouille de Données. Il est dans un cadre d'apprentissage non supervisé, qui tente d'obtenir des informations sans aucune connaissance préalable, ce qui n'est pas le cas de l'apprentissage supervisé. La question principale autour de laquelle s'articulera le travail du Clustering est de savoir d'imiter le mécanisme humain d'apprentissage sans aucune information disponible auparavant, en établant des méthodes qui permettent d'apprendre à partir d'un certain nombre de données et de règles (d'exemples), selon certaines caractéristiques sans aucune expertise ou intervention requise. En effet, ce processus requit certains traitements ou combinaison avec d'autres méthodes, en pre- ou en post-processing, surtout pour une grande masse de données, pour bien réaliser entièrement sa tâche de classification, L'ensemble des techniques de traitement est souvent regroupé sous le terme de «fouille de données».

Dans ce chapitre, nous allons présenter les principaux algorithmes de clustering parmi les plus représentatifs tout en essayant de bien cerner les points forts et les faiblesses de chaque algorithme.

2. Définition

Le clustering est une approche de classification dont les classes existent a posteriori. Au départ, on dispose d'un ensemble d'objets non étiquetés (dont la classe est inconnue). A partir de ces objets, l'idée est de parvenir à détecter des objets ‘‘similaires’’ afin de les regrouper dans des classes **[30]**.

Selon **[31]**, le clustering consiste à diviser une population d'objets en sous-ensembles d'objets appelés classes pour que tous les objets dans une même classe soient similaires et les objets de classes distinctes soient dissimilaires.

Parmi les objectifs que le clustering doit permettre d'atteindre, citons la réduction des données et la prédiction basée sur les groupes.

3. Différentes approches de clustering

De nombreux algorithmes ont été proposés dans le domaine de la classification non supervisée .Leurs principes diffèrent selon plusieurs facteurs : le type d'attributs qu'ils traitent, la capacité de traiter un gros jeu de données, la capacité de traiter des données en

haute dimension, les mesures de distances utilisées, la complexité de l'algorithme (temps de calcul), la dépendance de l'ordre des données, la dépendance des paramètres prédéfinis par l'utilisateur où la connaissance acquise à priori sur les données, etc. **[32]**.

Plusieurs problèmes et limites sont liés à la technique de clustering qui met en avant la multiplicité et surtout la nature différente des paramètres à prendre en compte dans les différentes approches relatives à cette technique.

Ces paramètres qui influent de manière plus au moins importante sur les résultats obtenus, sont nécessaires pour que l'algorithme soit relativement générique et pour que celui-ci soit applicable dans plusieurs cas. Plus le nombre de paramètres augmente plus l'algorithme est adaptable et peut s'appliquer à une gamme de problèmes plus large. Cependant, l'augmentation des paramètres nécessite une connaissance importante de la part de l'expert sur ses données et sur le fonctionnement de l'algorithme **[33]**, **[34]**.

Selon ces critères et la manière dont les classes sont construites, les approches de classification sont divisées en : approche hiérarchique, approche de partitionnement, approche basée sur la grille et enfin l'approche basée sur la densité.

La **Figure III.1** présente les différentes approches de clustering **[35]**.

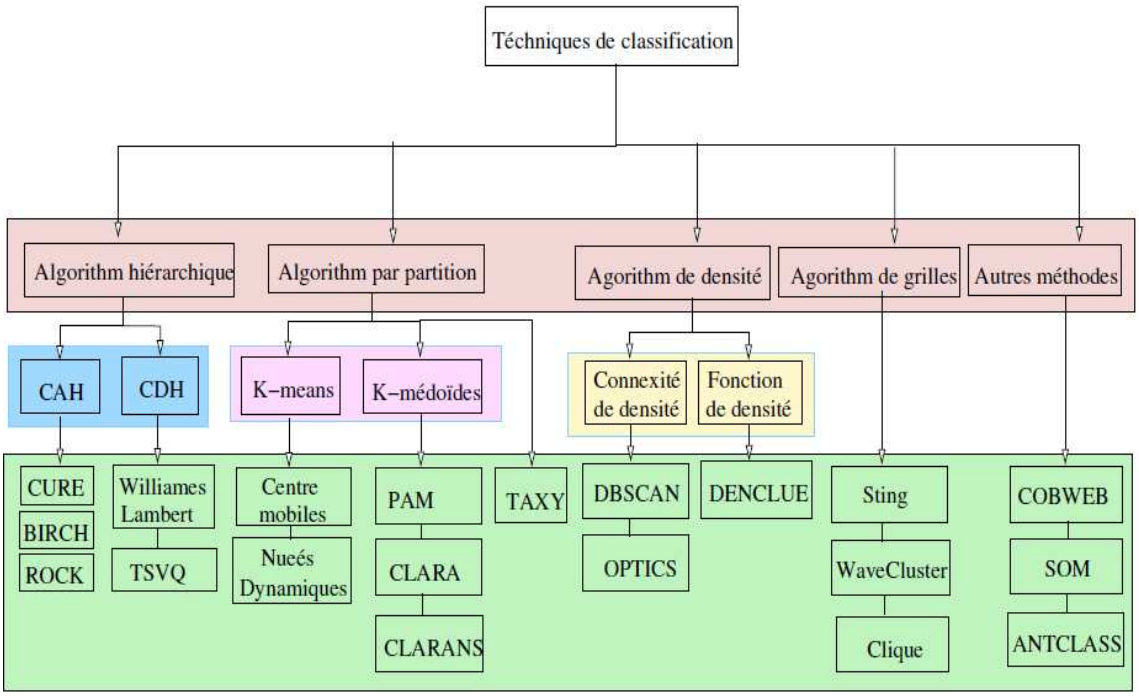


Figure III.1 : les différentes méthodes de clustering

Dans ce qui suit nous présenterons en détail les approches les plus utilisées dans les travaux de recherche de clustering.

3.1 Le clustering hiérarchique

Les méthodes hiérarchiques génèrent une succession de partitions emboîtées les unes dans les autres au lieu d'une seule partition de l'espace des données. Celles-ci sont souvent représentées sous la forme d'un dendrogramme (hiérarchie indicée).

Selon que l'on parcourt le dendrogramme « de haut en bas » ou « de bas en haut », la méthode sera dite divisive (descendante) ou agglomérative (ascendante).

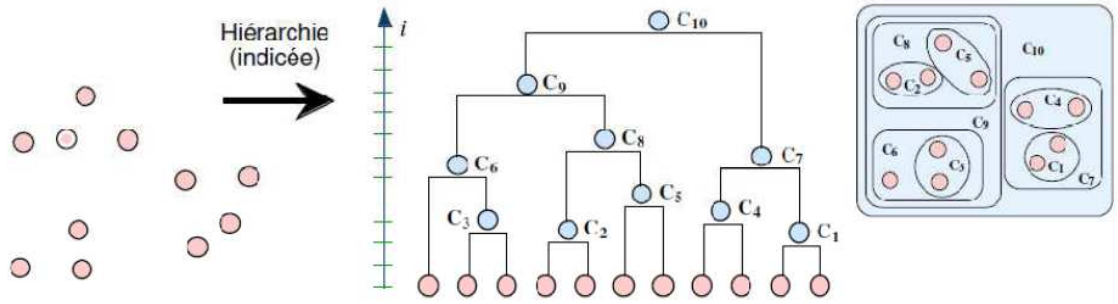


Figure III. 2: Exemple de dendrogramme et la détermination des clusters.

3.1.1 Les méthodes hiérarchiques ascendantes (agglomérative)

Permet de construire une hiérarchie entière objets sous la forme d'un "arbre" dans un ordre ascendant (du bas vers le haut). On commence en considérant chaque individu comme une classe et on essaye de fusionner deux ou plusieurs classes appropriées (selon une similarité) pour former une nouvelle classe. Le processus est itéré jusqu'à ce que tous les individus se trouvent dans une même classe. Cette classification génère un arbre que l'on peut couper à différents niveaux pour obtenir un nombre des classes plus ou moins grand.

Cette procédure est basée sur deux choix :

- La détermination d'un critère de ressemblance entre les individus. La méthode laisse à l'utilisateur le choix de la dissimilarité.
- La détermination d'une dissimilarité entre classes : procédée appelée un critère d'agrégation.

3.1.1.1. Les critères d'agrégation [35]

- **Le critère du saut minimal**

La distance entre 2 classes C1 et C2 est définie par la plus courte distance séparant un individu de C1 et un individu de C2.

$$D(C1, C2) = \min (d(x, y), x \in C1, y \in C2) \qquad (3.21)$$

➢ Le critère du saut maximal

La distance entre 2 classes C1 et C2 est définie par la plus grande distance séparant un individu de C1 et un individu de C2.

D (C1, C2) = max (d(x, y), x ∈C1, y∈ C2) (3.22)

➢ Le critère de la moyenne

Ce critère consiste à calculer la distance moyenne entre tous les éléments de C_1 et tous les éléments de C_2

$$D(C_1, C_2) = \frac{1}{n_{C_1} n_{C_2}} \sum_{x \in C_1} \sum_{y \in C_2} d(x, y) \qquad (3.23)$$

Avec :

n_{C_1} : Le cardinal de C_1

n_{C_2}: Le cardinal de C_2

➢ Le critère des centres de gravité

La distance entre 2 classes C_1 et C_2 est définie par la distance entre leurs centres de gravité.

$$D(C_1, C_2) = d\left(g_{C_1}, g_{C_2}\right) \qquad (3.25)$$

Avec :

g_{C_1} : Le centre de gravité de C_1

g_{C_2} : Le centre de gravité de C_2

➢ Le critère de Ward

Ce critère ne s'applique que si on est muni d'un espace euclidien. La dissimilarité ρ entre deux individus doit être égale à la moitié du carré de la distance euclidienne d. Le critère de Ward consiste à choisir à chaque étape le regroupement de classes tel que :
L'augmentation de l'inertie intra-classe soit minimale.

$$D(C_1, C_2) = \frac{n_{C_1} n_{C_2}}{n_{C_1} + n_{C_2}} d^2(gC_1, gC_2) \qquad (3.24)$$

Avec :

g_{C_1} : Le centre de gravité de C_1

g_{C_2} : Le centre de gravité de C_2

La difficulté du choix du critère d'agrégation réside dans le fait que ces critères peuvent déboucher sur des résultats différents. Selon les plus parts des références le critère le plus couramment utilisé est celui du Ward.

3.1.1.2 L'algorithme CURE (Clustering Using Representatives)

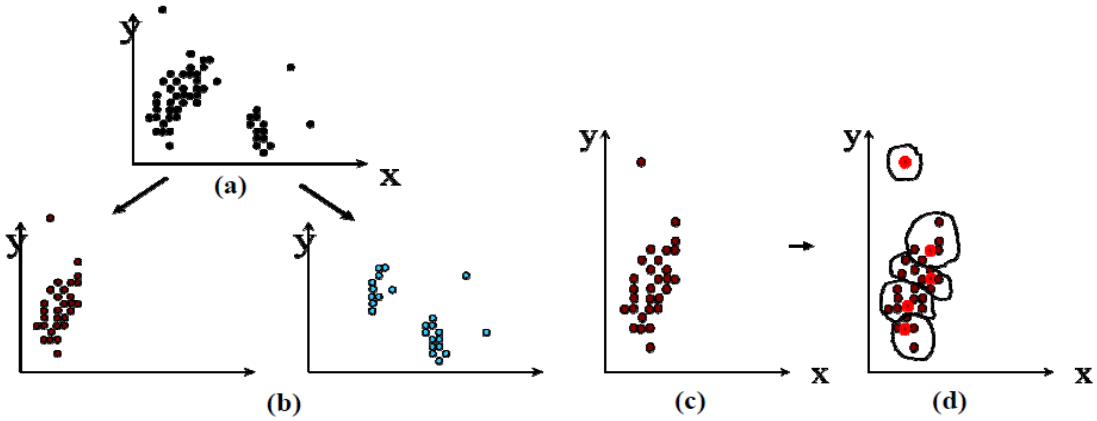
L'algorithme CURE proposé par Guha et al est plus robuste face au bruit (outliers) et permet d'identifier des groupes non sphériques et d'une grande variance de taille. Il réalise ceci en représentant chaque groupe par un nombre fixé de points qui sont générés en sélectionnant des points bien dispersés du groupe, et ensuite rapproché du point moyen au centre du groupe en le multipliant par un coefficient.

Le fait d'avoir plus d'un point représentatif permet à CURE de bien s'ajuster à la géométrie des clusters non sphériques et l'opération de rapprochement de ses points permet de diminuer les effets des outliers.

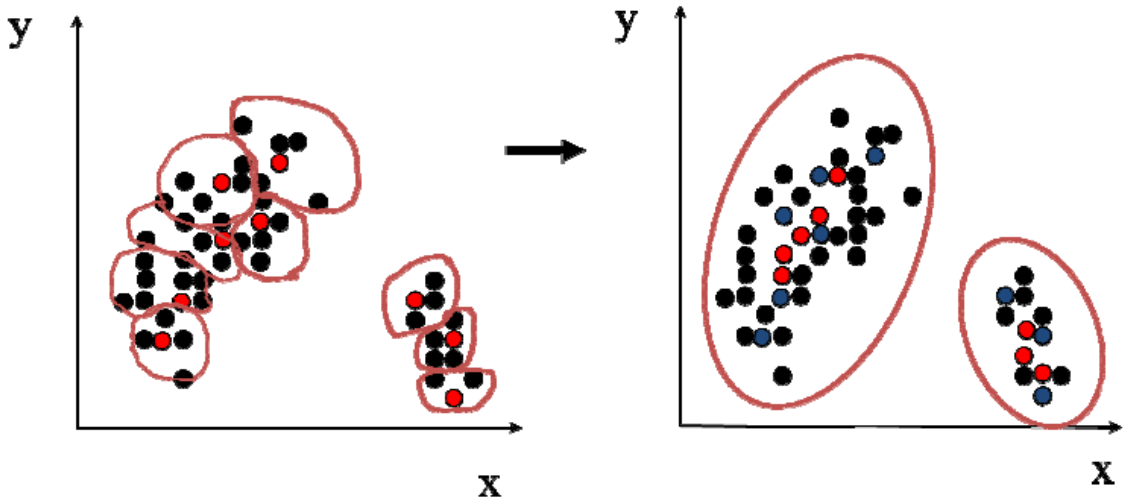
Le détail de cette méthode est décrit dans l'algorithme suivant **[36]*:***

L'algorithme Cure
• Prendre un sous-ensemble *s.*
•Partitionner s en *p* partitions.
•Dans chaque partition, créer *s/pq* clusters.
•Eliminer les exceptions (points aberrants)
•Regrouper les clusters partiels

Considérons l'exemple présenté dans la **figure III.3**, ce dernier est partitionné en deux partitions *p=2* partiellement mise en *s/pq=5* clusters **(d) [12].**



FigureIII.3 : Partitionement (b) Clustering (c).



FigureIII.4: Regroupement de clusters partiels.

Pour manipuler de grands volumes de données, CURE propose une combinaison d'échantillonnage aléatoire et de partitionnement. Un échantillon tiré de l'ensemble des données est tout d'abord partitionné et chaque partition est partiellement mise en cluster.
Chacun de ces groupes partiels sera à niveau regroupé lors d'une seconde passe de l'algorithme pour extraire les clusters désirés. Le principal problème de CURE est sa complexité en terme temps d'exécution.

3.1.2 Les méthodes hiérarchiques descendantes (divisve)

Contrairement, dans une approche descendante (divisive), on part d'un grand cluster que l'on divise en deux progressivement de façon à optimiser un critère donné pour obtenir au final un ensemble de singletons (des groupes qui contiennent un seul individu).parmi les algorithmes les plus connus dans cette méthodes on site : Williames et Lambert, TSVQ.

Dans les deux cas, un dendrogramme représente les différentes étapes successives de la recherche des clusters. Il présente à chaque niveau quels éléments ont été rassemblés dans une approche de « agglomerative » ou au contraire quels éléments ont été créés dans une approche de « divisive ».

Les avantages de la classification hiérarchique

- Facilité pour traiter différentes formes de similarité ou de distance entre objets.
- Applicable aux différents types d'attributs.
- Une flexibilité en ce qui concerne le niveau de granularité.

Les points faibles

- Choix du critère d'arrêt qui reste vague.
- Fait que la plupart des algorithmes hiérarchiques ne revoient pas les clusters intermédiaires qu'une fois qu'ils sont construits pour les améliorer.

3.2 Méthodes de partitionnement

Leur principe général est de démarrer à partir d'un certain nombre de classes qui sont partitionnés d'une manière aléatoire en effectuant une ‘‘redistribution’’ des objets ou en essayant d'identifier les classes comme étant des régions très peuplées jusqu'à la rencontre d'un critère d'arrêt.

Ces méthodes de partitionnement sont basées sur une distance ou un indice de similarité entre objets à classer.

Les algorithmes de partitionnement que nous allons développer sont k-Means, k-Medoïds, PAM et CLARA

3.2.1 Méthode k-Means

L'algorithme k-means (k-moyenne) a été introduit par J.MacQuenn. Il est sans aucun doute la méthode de partitionnement la plus connue et la plus utilisée dans divers domaines d'application scientifiques et industrielles. Ce succès est dû au fait que cet algorithme présent un rapport coût/efficacité avantageux.

Dans sa version classique, l'algorithme consiste à sélectionner aléatoirement k objets qui représente les centroïdes initiaux. Un objet est assigné au cluster pour lequel la distance entre l'objet et le centroïde est minimale. Les centroïdes sont alors recalculés et l'on passe à l'itération suivante.

La fonction objectif traditionnellement utilisée est :

$$F = \sum_{r=1}^{k} \sum_{x_i \in c_r} (x_i - g_r)^2 \qquad (3.26)$$

Où :

c_r est la classe numéro *r*

x_i est un objet dans une classe

g_r est le centre de classe c_r

La **figure III.5** montre un exemple de déroulement de l'algorithme k-means sur un nuage d'objets bidimensionnels, avec k=3 :

La figure **III.5.(a)** : choix aléatoire de trois centre initiaux (objets) marqués par (+) et affectation de chaque objet restant au centre le plus proche ;

La figure **III.5.(b)** : calcul du nouveau centre pour chaque classe (moyenne des objets de la classe) et redistribution des objets selon les nouveaux centres ;

La figure **III.5.(c)** : le processus est répété jusqu'à stabilité des classes.

.

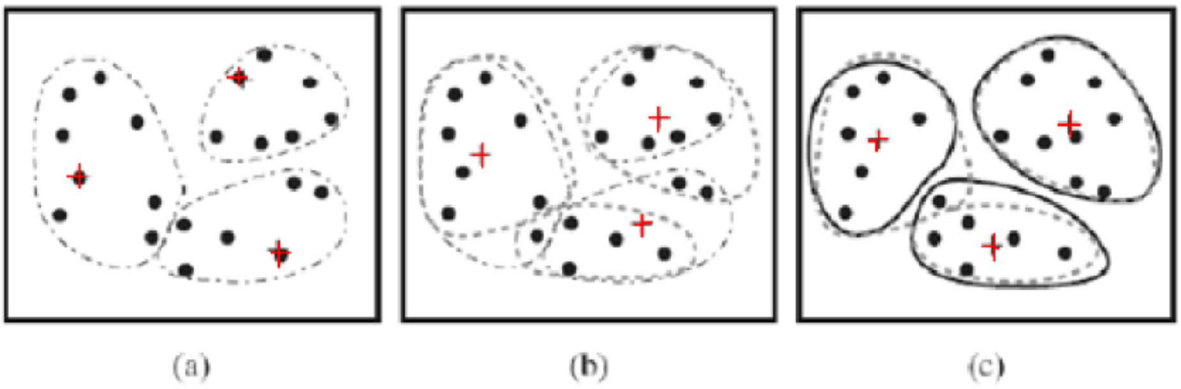


Figure III.5 : Partitionnement basé sur k-means

L'algorithme général de ces méthodes **[37]** :

Algorithme k-means :
Données : k le nombre maximum de classes désiré. **Début** (1) Choisir k objets au hasard (comme centre des classes initiales) (2) Affecter chaque objet au centre le plus proche (3) Recalculer le centre de chacune de ces classes (4) Répéter l'étape (2) et (3) jusqu'à stabilité des centres (5) Editer la partition obtenue. **FIN**

L'algorithme des k-means ne peut être utilisé que sur des données décrites par des attributs numériques permettant ainsi le calcul des centroïdes. Dans le cas d'attributs symboliques (ou catégoriels) plusieurs variantes ont été proposées : méthodes des centres mobiles et méthodes des nuées dynamiques.

3.2.1.1 Méthode des centres mobiles:

L'auteur de cette méthode est Forgy en 1965 [38], le principe de l'algorithme consiste à construire une partition en k classes en sélectionnant k objets au hasard de l'ensemble des objets et les considérer comme centres des classes. Après cette sélection, on affecte chaque objet au centre le plus proche en créant k classes, les centres des classes seront remplacés par des centres de gravité. Ainsi des nouvelles classes seront créés. Généralement la partition obtenue est localement optimale car elle dépend du choix initial des centres. Pour cette raison, les résultats entre deux exécutions de l'algorithme sont significativement variés.

Algorithme centre mobile
Données : k le nombre maximum de classes désiré. **Début** (1) Choisir k objets au hasard (comme centre des classes initiales) (2) Affecter chaque objet au centre le plus proche, ce qui donne une partition en k classes $P_1 = \{C_1,\ldots,C_k\}$ (3) On calcule les centres de gravité de chaque classe de P_1, ce qui donne k nouveaux centres de classes (4) Répéter l'étape 2 et 3 jusqu'à ce que deux itérations successives donnent la même partition (5) Editer la partition obtenue. **FIN**

Avantages de k-means

Comme avantages de cet algorithme, on cite :

-Grande simplicité.

- Compréhensible.

- Les classes sont facilement interprétables et représentées naturellement par les centroïdes.

-Il s'adapte bien pour des populations de grandes tailles.

-Complexité algorithmique est intéressante, puisqu'elle est en $o(nkt)$ où t est le nombre d'itérations, n le nombre d'objets et k le nombre de classes.

- L'algorithme converge, généralement, avec k et t suffisamment inférieur à n (k, t<<n).

Inconvénients de k-means

Parmi les inconvénients de cet algorithme, on cite :

- Le nombre de classes doit être fixé au départ.

-Le résultat dépend du tirage initial des centres des classes.

-Il ne détecte pas les données bruitées (isolées).

-Le nombre de classe est un paramètre de l'algorithme. Un bon choix du nombre k est nécessaire, car un mouvais choix de k produit de mauvais résultat.

3.2.2 Méthodes des k-médoïdes

Les méthodes des k-médoïdes se différencient de la méthode des k-means par l'utilisation de médoïdes plutôt que des centroïdes pour représenter les classes. L'esquisse de ces méthodes ressemble à celle des k-moyennes sauf que, contrairement à l'algorithme k-means où la classe est représentée par une valeur moyenne, le centroïde, dans l'algorithme k-medoïdes une classe est représentée par un de ses objets prédominants, le médoïde.

L'algorithme a été introduit initialement par Kaufman et Roussew, c'est un algorithme itératif combinant la réaffectation des objets dans des classes avec une intervention des médoïdes et des autres objets.

Comme avantages de cet algorithme par rapport à k-Means, on cite :

- Il s'adapte à n'importe quel type de données .
- Il est insensible aux objets isolés.

Pour améliorer la qualité de clustering de L'algorithme k-Medoids, de nombreuses variantes s'y sont succédées, entre autres les algorithmes PAM, CLARA et CLARANS.
Dans ce qui suit nous allons présenter les principes de l'algorithme PAM et CLARA

3.2.2.1 L'algorithme PAM

PAM (*Partitioning around Medoids*) a été développée par Kaufman et Rousseeuw [39]. L'idée de cet algorithme consiste à commencer avec un ensemble de *k* médoides puis échanger le rôle entre un objet médoide et un non-médoide si cela permet de réduire la distance globale.

L'avantage de cet algorithme est qu'il est plus robuste que les méthodes k-means en présence de bruit,et son inconvénient majeur est son coût total de calcul, il est d'une complexité quadratique de l'ordre de $o(k.(n-k)^2)$ avec *n* le nombre d'objets et *k* le nombre de classes pour chaque itération, ceci le rend non adaptable pour une population importante d'objets.

3.2.2.2 L'algorithme CLARA

L'algorithme CLARA (*Clustering LARge Application*) a été mise en oeuvre par Kaufman et Rousseeuw dans le but de réduire le coût de calcul de PAM.
Cet algorithme travaille sur des échantillons. On prend une petite partie d'objets (échantillon). Ensuite, les *k* médoides sont déterminés en appliquant PAM sur cet échantillon. Si cet échantillon est choisi d'une manière aléatoire, alors il représente bien tous les objets, donc les médoides sont similaires à ceux qui sont créés à partir de tous les objets.

L'algorithme est, généralement, exécuté sur plusieurs échantillons pour obtenir le meilleur résultat. Les auteurs ont indiqué, suite à leurs expérimentations, que la taille d'un échantillon de (40+2*k*), *k* étant le nombre de classes, donne un bon résultat [30].
Le principal inconvénient de l'algorithme CLARA, outre les difficultés d'échantillonnage, est que si un objet qui serait le meilleur médoide n'apparait dans aucun échantillon, alors la meilleure solution ne pourra jamais être atteinte.

3.3 Les méthodes basées sur la densité

Les algorithmes basés sur la densité sont capables de découvrir des clusters de formes arbitraires, ce qui assure l'isolement des bruits (outliers) et la prévention contre la formation de clusters non pertinents **[40]**.

Ces algorithmes groupent des objets selon des fonctions de densité spécifiques. La densité est habituellement définie comme nombre d'objets dans un voisinage particulier des éléments de données. Dans cette approche, un cluster donné continue à augmenter de taille tant que le nombre d'objets dans le voisinage dépasse un certain seuil.
Cette approche se subdivise en deux types :

3.3.1 Méthodes basée sur la densité connective (Density-Based Connectivity Clustering)

Dans cette technique de clustering [40], la densité et la connectivité sont mesurées en termes de distribution locale des voisins les plus proches.

La densité-connectivité ainsi définie est une relation symétrique et tous les points accessibles à partir des noyaux des objets peuvent être factorisés dans les composants connectés maximaux servant de clusters. Les points qui ne sont pas connectés à tout point du noyau sont considérés comme des bruits (outliers) et ils ne sont couverts par aucun cluster.

Les points non fondamentaux à l'intérieur d'un cluster représentent sa borne. Finalement, les objets du noyau sont les points internes. Le processus est indépendant de l'ordre de données et n'a pas de limitation sur les dimensions ou le type d'attributs.

3.3.2 Méthodes basée sur les fonctions densité (DENSITY FUNCTIONS CLUSTERING)

Dans cette méthode, une fonction de densité est utilisée pour le calcul de la densité. La densité globale est définie comme la somme des fonctions de densité de tous les objets.

Les clusters sont déterminés par les attracteurs de densité qui sont définis comme les maxima locaux de la fonction de densité globale.

3.3.3 DBSCAN (Density Based Spatial Clustering of Applications with Noise) [39]

Dans cette méthode la découverte d'un groupe se déroule en 2 étapes :

- Choisir aléatoirement un point dense.
- Tous les points qui sont atteignables à partir de ce point, selon le seuil de densité, forment un groupe.

Un point est considéré dense si le nombre de ses voisins est supérieur à un MinPts (Paramètre d'entrée). Deux points sont dit voisins si la distance entre eux ne dépasse pas un seuil donné (Eps).

DBSCAN commence avec un point de départ arbitraire qui n'a pas été visité. Si ce point n'est pas dense, alors c'est un bruit, sinon il serait assigné avec l'ensemble de ses voisins à un nouveau cluster. Ce procédé est répété d'abord sur l'ensemble des voisins puis sur le reste des points qui n'ont pas été classés ou marqués bruit. A la fin de l'algorithme deux types de point apparaissent : les points denses qui appartiennent à un groupe et le reste qui est considéré comme bruit.

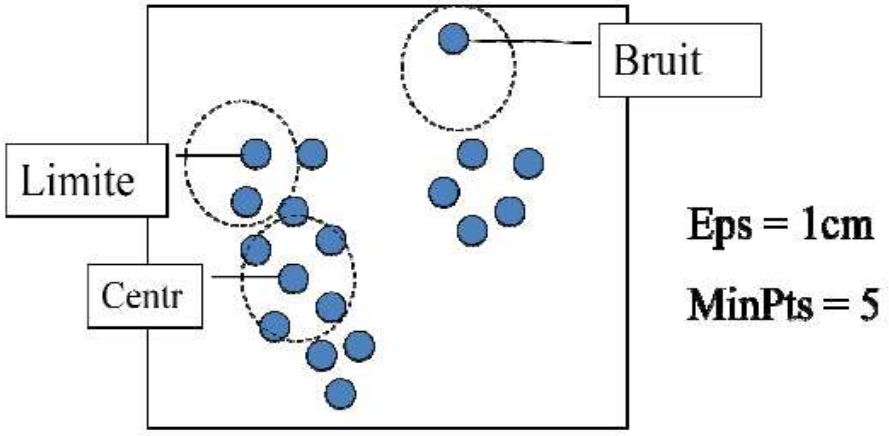


Figure III.6 : Les points denses et les points bruit.

3.4 Méthode basée sur les grilles

L'idée de ces méthodes est qu'on divise l'espace de données en un nombre fini de cellules formant une grille. Ce type d'algorithme est conçu pour des données spatiales. Une cellule peut être un cube, une région, un hyper rectangle.

En fait, avec une telle représentation des données, au lieu de faire la classification dans l'espace de données, on la fait dans l'espace spatial en utilisant des informations statistiques des points dans la cellule. Les méthodes de ce type sont hiérarchiques ou de partitionnement. Les algorithmes les plus connus sont STING, CLIQUE, WaveCluster.

4. Mesures d'évaluation de la qualité des clusters

La comparaison entre les différents algorithmes de clustering est une tâche très difficile car, contrairement à l'apprentissage supervisé, nul ne sait si l'ensemble des clusters obtenus est exact, en particulier dans les espaces de grande dimension. L'objectif principal de la validation des clusters est d'évaluer le résultat de clustering afin de trouver le meilleur partitionnement du jeu de données. Il existe plusieurs mesures, nous présentons dans ce qui suit quelques-unes **[41] [42] :**

4.1 . Entropie

La mesure d'Entropie représente le degré relatif d'aspect aléatoire du partitionnement peut être évalué en utilisant la notion d'entropie d'un cluster.

L'entropie est une mesure de qualité qui permet de mesurer la répartition des objets dans un cluster.

L'entropie d'un cluster C de taille n_r est calculée selon la formule suivante :

$$Entropie(C) = -\frac{1}{\log q}\sum_{n=1}^{q}\frac{n_r^i}{n_r} log \frac{n_r^i}{n_r} \qquad (3.27)$$

Où : q représente le nombre total de clusters et n_r^i représente le nombre de séquences qui font partie du i^{eme} cluster C.

L'entropie totale du clustering calculée en fonction des entropies générées par tous les clusters de la partition est donnée par la formule suivante :

$$Entropie = \sum_{r=1}^{q}\frac{n_r}{n}(C_{r)} \qquad (3.28)$$

Où : n représente le nombre total de séquences.

On considère qu'une petite valeur d'entropie totale traduit un bon clustering des données.

4.2 Inertie Intra clusters.

La mesure d'inertie-intra cluster d'un cluster k, mesure la concentration des points N_k du cluster autour de son centre de gravité μ_k qu'est calculée par les deux formules suivantes :

$$Intra(C) = \sum_{i \in C_K} d^2\left(x_i, \mu_k\right) \qquad (3.29)$$

$$\mu_k = \frac{1}{N_K} \sum_{i \in C_K} x_i$$

L'inertie intra totale du partitionnement est la sommation des inerties intra clusters Plus cette inertie est faible, plus petite est la dispersion des points autour du centre de gravité. Pour obtenir un bon partitionnement basé sur une fonction de distance, il convient de minimiser l'inertie intra-classe pour obtenir des clusters les plus homogènes possible.

4.3 Inertie Inter cluster

La mesure d'inertie inter cluster d'un cluster k, mesure l'éloignement des centres des clusters entre eux. Elle prend en compte la dispersion des clusters selon leurs centres de gravité μ_k par rapport au centre de gravité du nuage de points μ.

Elle est calculée par les deux formules suivantes :

$$Inter(C) = \sum_{k} N_k \; d^2\left(\mu_k, \mu\right) \qquad (3.30)$$

$$\mu_k = \frac{1}{N_k} \sum_{i \in C_K} x_i$$

L'inertie inter totale du partitionnement est la sommation des inerties inter clusters. Plus cette inertie est grande, plus les clusters sont bien séparés afin d'obtenir une bonne partition.

Pour obtenir un bon partitionnement basé sur une fonction de distance, il convient de maximiser l'inertie inter afin d'obtenir des sous-ensembles bien différenciés.

5. Conclusion

La technique de clustering présente l'une des techniques du Data Mining les plus utilisée, appliquée dans divers domaines dont les données arrivent sous forme d'un flux. Dans le cadre de cette technique, nous avons étudié dans ce chapitre les différentes approches relatives à cette technique ainsi que leurs algorithmes respectifs proposés.

Ces différentes approches diffèrent les unes des autres par leur principe de segmentation, les mesures de proximité (distances) qu'elles utilisent, la nature des données et attributs qu'elles traitent. Dans ce cadre, une analyse attentive des données aide à bien choisir la meilleure approche et de ce fait, le meilleur algorithme adéquat. Le choix de l'algorithme approprié dépend fortement de l'application, la nature des données et les ressources disponibles. Nous avons passé en revue quatre types d'approches basées sur la technique de clustering plus précisément les approches hiérarchiques et de partitionnement qui constituent les approches les plus utilisées dans ce contexte. Nous avons à cet effet, détaillé leurs principes de fonctionnement et présenté leurs points faibles et leurs points forts.

Dans ce qui suit nous allons présenter d'une manière simple et complète le fonctionnent de le logiciel weka (environnement Waikato pour l'analyse de connaissances) qui sera utilisé dans le dernier chapitre pour notre étude comparative des différentes méthodes de classification.

1. Introduction

Depuis plusieurs années, data mining a été un vaste domaine de recherche pour de nombreux chercheurs en raison de la quantité énorme de données et d'informations disponibles dans les bases de données. Avec une telle quantité de données, il existe un besoin de techniques et d'outils puissants qui peuvent gérer les données de meilleure façon et extraire la connaissance pertinente.

Dans ce chapitre nous présentons quelques outils de data mining aux différentes catégories, et nous exposons aussi en détail l'outil que nous avons utilisé dans notre étude comparative «Weka», ses outils, ses différentes interfaces, ses concepts de base, son mode d'utilisation en se basant particulièrement sur la partie classification.

2. Définition

Les outils de fouille de données sont des programmes spécialisés dans l'analyse et extraction de connaissance à partir de grande quantités des données informatisées, pour objectif aide l'analyste en exploration de données : extraction d'un savoir ou d'une connaissance par des méthodes automatique ou semi-automatique.

L'exploration des donnés (fouille des données) se propose d'utiliser un ensemble d'algorithmes au différents disciplines scientifiques telle que les statistiques, l'intelligence artificiel ou l'informatique pour construire des modèles à partir des données, afin de trouver des structures intéressantes ou des motifs selon des critères fixés au préalable, et d'en extraire un maximum de connaissances.

On peut classer les outils en deux catégories très distinctes :

A. . **Les logiciels commerciaux :** proposent une interface graphique conviviale, ils sont destinés à la mise en œuvre de traitements sur des données en vue du déploiement des résultats. Les méthodes disponibles sont souvent peu référencées, il est de toute manière impossible d'accéder à l'implémentation

B. Les outils libres : sont constitués d'un assemblage de bibliothèques de programmes. Un chercheur peut facilement accéder au code source pour vérifier les implémentations, ajouter ses propres variantes, et mener de nouvelles expérimentations comparatives. Ces plates-formes ne sont guère accessibles à des utilisateurs non informaticiens.

3. Présentation

Weka, autrement dit, environnement Waikato pour l'analyse de connaissances [43], est un package Open Source très populaire, en d'autres termes, un ensemble de classes et d'algorithmes développé sous Java, à l'Université de Waikato en Nouvelle-Zélande. Il propose différents algorithmes d'apprentissage automatique (supervisée ou non), à savoir, Naïve Bayes, Arbre de décision, SVM, réseau de neurones, etc. avec les fonctionnalités de prétraitement des données, analyse et évaluation des résultats. Il offre une interface GUI conviviale pour manipuler et inspecter les données et visualiser les résultats. Ce package peut fonctionner sur les plateformes Linux, Windows et Mac.

La figure ci-dessous présente la fenêtre d'invite au lancement de Weka

Figure IV.1 : Fenêtre d'invite Weka

Weka se compose principalement :

- De classes Java permettant de charger et de manipuler les données.
- De classes pour les principaux algorithmes de classification supervisée ou non supervisée.
- D'outils de sélection d'attributs, de statistiques sur ces attributs.
- Des classes permettant de visualiser les résultats. On peut l'utiliser à trois niveaux:
  - Via l'interface graphique, pour charger un fichier de données, lui appliquer un algorithme, vérifier son efficacité.
  - Invoquer un algorithme sur la ligne de commande.
  - Utiliser les classes définies dans ses propres programmes pour créer d'autres méthodes, implémenter d'autres algorithmes, comparer ou combiner plusieurs méthodes.

4. Historique

➢ En **1992**, **l'Université de Waikato en Nouvelle-Zélande** commença le développement de la version originale de Weka.

➢ En **1997**, la décision fut prise de développer une nouvelle fois Weka à partir de zéro en Java, y compris l'implémentation des algorithmes demodélisation.

➢ En **2005**, Weka reçoit le **SIGKDD** (Data Mining and Knowledge Discovery Service Award).

➢ En **2006**, **Pentaho** acquiert une licence exclusive pour utiliser Weka pour de l'informatique décisionnelle. Il forme le composant d'exploration de données analytique et prédictif de la suite de logiciels décisionnels Pentaho.

5. Concepts de base de Weka

5.1 Format de données

Weka comporte son propre format de fichier appelé ARFF (Attribute Relation File Format), mais peut aussi traiter des données issues de bases relationnelles (BDD SQL), binaire, des fichiers type CSV (Open File), ou encore, charger des fichiers sur le Web (OpenURL).

Le format ARFF précise essentiellement trois champs :

- ➢ Définition du nom de l'ensemble de données avec @relation ; Le nom doit être aussi compréhensible que possible.
- ➢ Définition des features avec @attribute.
  - Attributs nominaux suivis des valeurs entre accolades.
  - Attributs numériques avec real.
  - Attributs chaines avec string, les valeurs doivent être entre doubles guillemets.
  - Attributs dates avec date (yyyy-MM-dd-THH : mm:ss).
- ➢ @datasignale le début des instances.

Note : Les commentaires sont précédés de % (% Ceci est un commentaire)

La figure suivante illustre un exemple de fichier ARFF :

```
% Ensemble de donnees sur la meteo
@relation weather

% Definition des features
@attribute outlook {sunny, overcast, rainy}
@attribute temperature real
@attribute humidity real
@attribute windy {TRUE, FALSE}
@attribute play {yes, no}

% Debut des instances
@data
sunny,85,85,FALSE,no
sunny,80,90,TRUE,no
overcast,83,86,FALSE,yes
rainy,70,96,FALSE,yes
...
```

Figure IV.2 : Fichier ARFF

6. Interfaces Weka

Weka dispose de plusieurs interfaces graphiques qui sont :

6.1. Explorer

L'interface graphique du logiciel Weka présente six onglets correspondant soit à des étapes du processus d'apprentissage, soit des classes d'algorithmes de classification (supervisée ou non):

- **Preprocess :** La saisie des données, l'examen et la sélection des attributs, les Transformations d'attributs.
- **Classify :** Les méthodes de classification.
- **Cluster :** Les méthodes de segmentation (clustering).
- **Associate :** Les règles d'association.
- **Select attributes** : L'étude et la recherche de corrélations entre attributs.
- **Visualize :** représentations graphiques des données.

La **figure V.3** Présente l'interface principale Explorer de Weka.

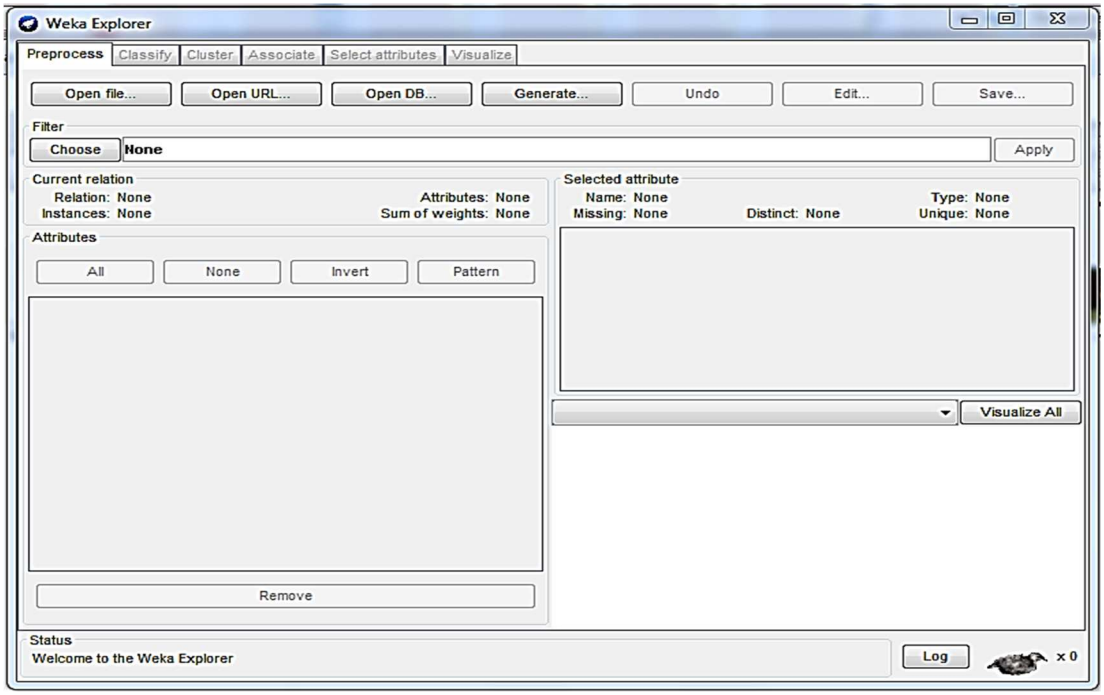


Figure IV.3 : Interface Explorer

Les différents onglets de l'interface principale Explorer sont :

6.1.1. L'onglet Preprocess

L'onglet **Preprocess**, ou « préprocesseur » a plusieurs fonctionnalités d'import de données (saisie des données, examen et sélection des attributs, transformations d'attributs), permet de charger un fichier au format spécifique de Weka, le format ARFF, ou, depuis des bases de données ou un fichier CSV et pour prétraiter ces données avec un algorithme appelé filtering. Ces filtres peuvent être utilisés pour transformer les données (par exemple, transformer des attributs numériques réels en attributs discrets) et rendre possible l'effacement d'instances et d'attributs selon des critères spécifiques.

La figure IV.4 illustre l'interface de l'onglet Preprocess

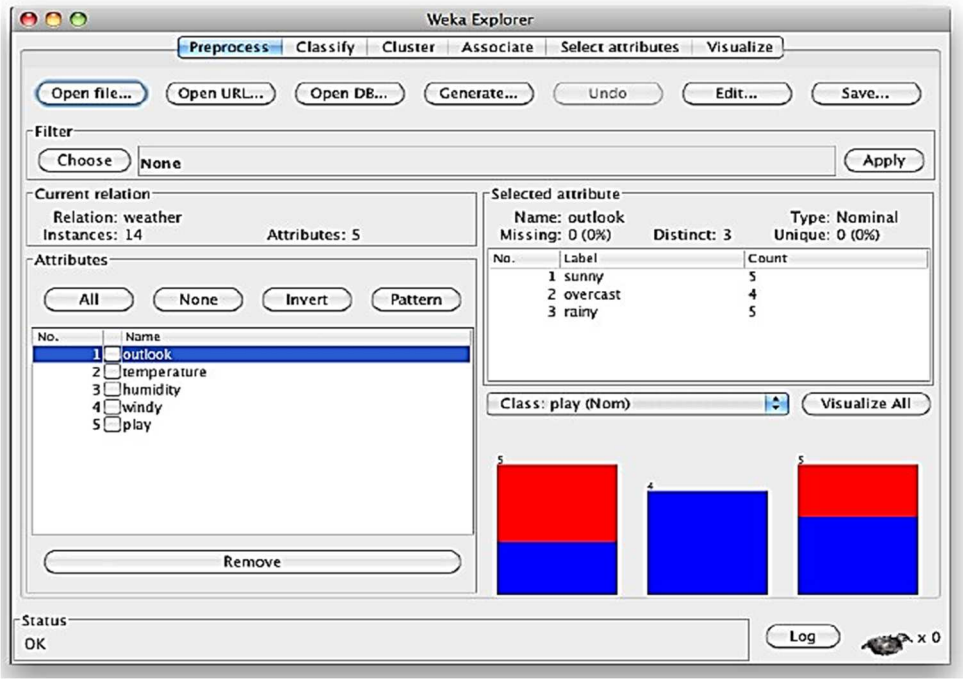


Figure IV.4 : Onglet Preprocess

Une fois le fichier ARFF chargé, une liste d'attributs apparaît à gauche de la fenêtre

6.1.2. L'onglet Classify

Une fois les données chargées, l'onglet Classify permet à l'utilisateur d'appliquer des méthodes de classification et des algorithmes de régression au jeu de données résultant, pour estimer la précision du modèle prédictif, et de visualiser les prédictions erronées ou le modèle lui- même (si le modèle est sujet à visualisation, comme un Arbre de décision).

La figure III.5 illustre l'interface de l'onglet Classify

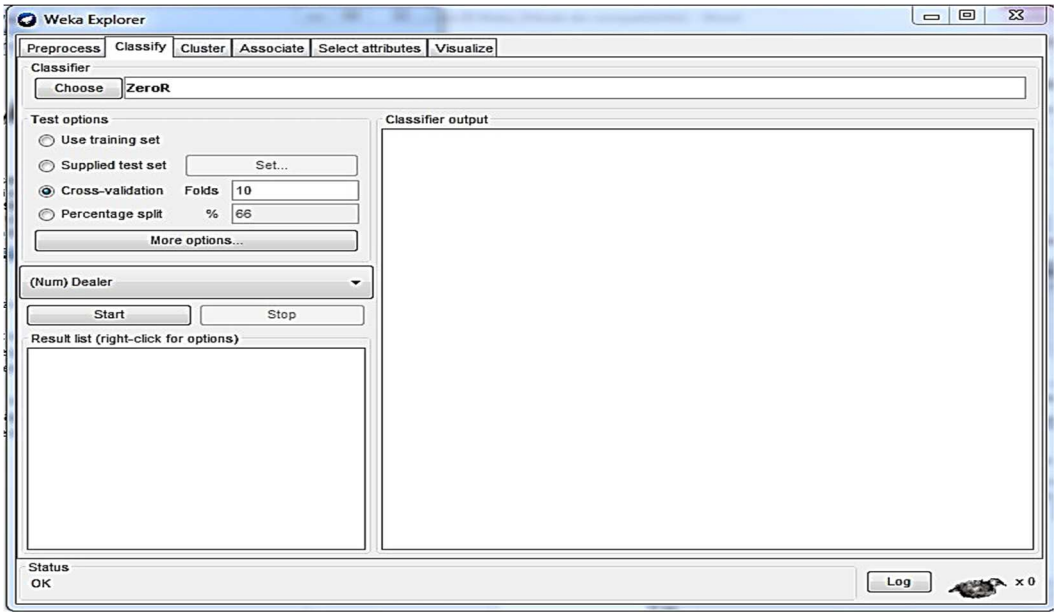


Figure IV.5 : Onglet Classify

6.1.3. L'onglet Cluster

L'onglet **Cluster** donne accès aux techniques de clustering (méthodes de segmentation) de Weka, comme l'algorithme K-means.

La figure IV.6 illustre l'interface de l'onglet Cluster

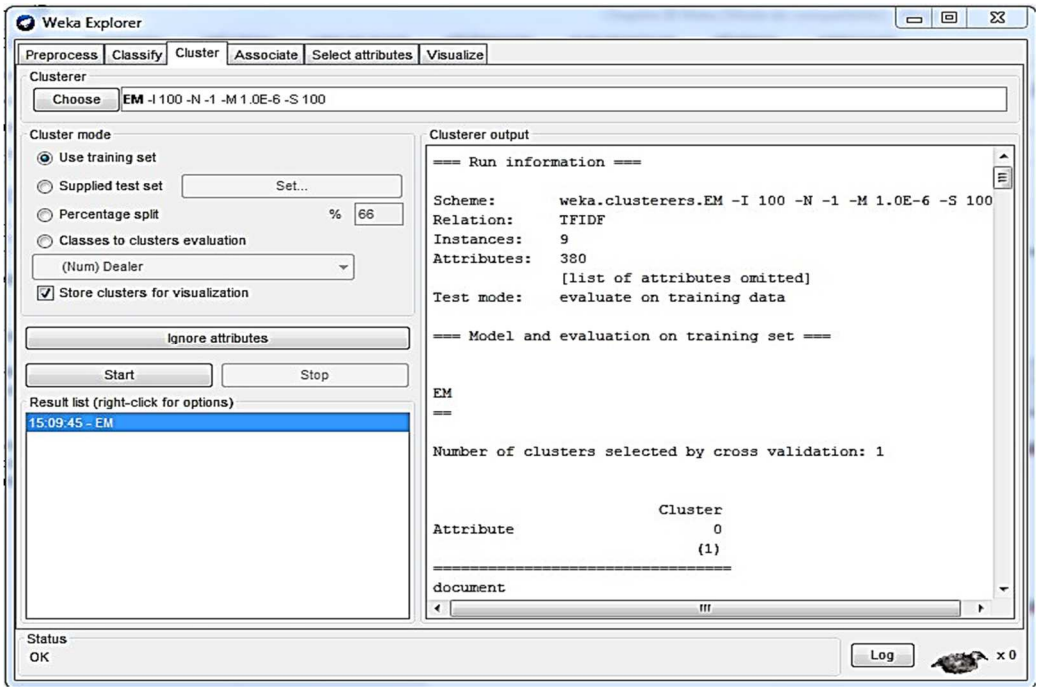


Figure IV.6 : Onglet Clust

6.1.4. L'onglet Associate : contient les règles d'association. La **figure III.7** illustre l'interface de l'onglet Associate

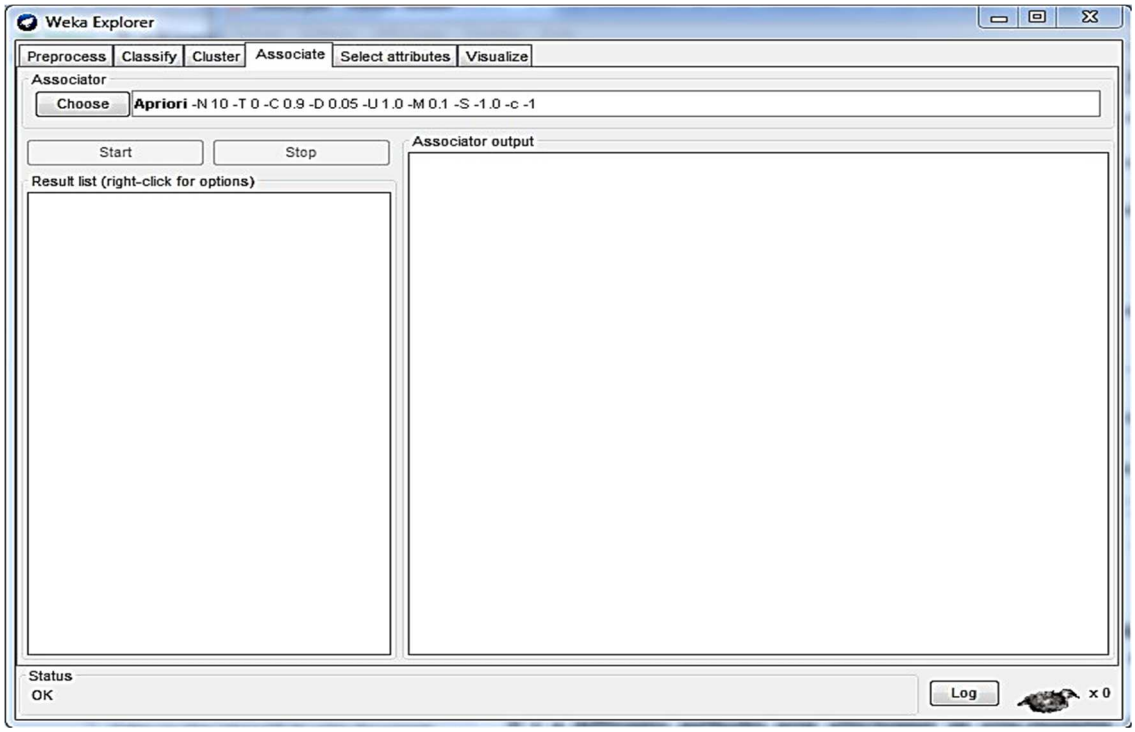


Figure IV.7 : Onglet Associate

6.1.5. L'onglet Select attributes : L'onglet « Select attributes » fournit des algorithmes pour l'identification des attributs les plus prédictifs dans un jeu de données et permet la sélection des attributs à utiliser pour la classification.

Il y a différentes méthodes pour sélectionner un sous-ensemble d'attributs à utiliser dans la classification. Ceci est très utile quand les données sont très bruitées, avec beaucoup d'attributs qui n'aident pas à la classification. Un nettoyage (sélection) est très bénéfique dans ce cas. Cette sélection aide aussi à accélérer les traitements. Pour cela, il faut choisir dans « Attribute Evaluator » la méthode InfoGainAttributeEval (la sélection basée sur le gain d'information), et dans « Search Method » la méthode Ranker – qui ordonne les attributs selon leur valeur. En cliquant sur Ranker, on peut préciser les critères de sélection, par exemple en fixant un seuil, ou en fixant un nombre d'attributs à garder.

La **figure IV.8** illustre l'interface de l'onglet Select Attribute

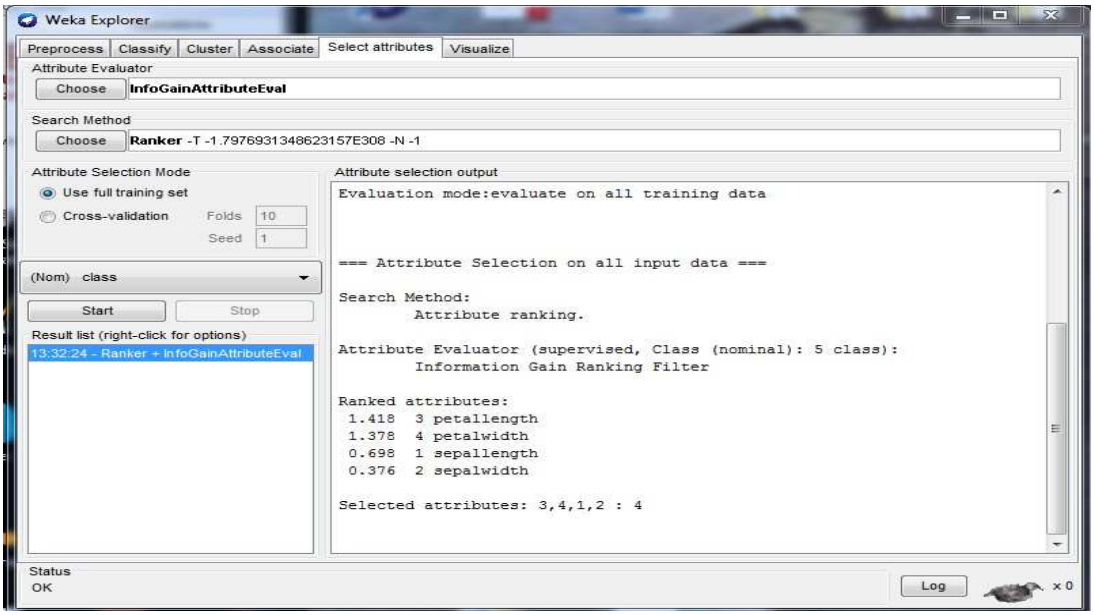


Figure IV.8 : Onglet Select attributes

6.1.6. L'onglet Visualize : Le dernier onglet « Visualize » montre une matrice de nuages de points, ou des nuages de points individuels peuvent être sélectionnés et élargis, et davantage analysés en utilisant divers opérateurs de sélection. La fenêtre Visualize dispose d'un ensemble de 25 graphiques, qui représentent chacun une vue sur l'ensemble d'exemples selon deux dimensions possibles, la couleur des points étant leur classe. Sur le graphique, chaque point représente un exemple : on peut obtenir le descriptif de cet exemple en cliquant dessus. La couleur d'un point correspond à sa classe. Au départ, le graphique n'est pas très utile, car les axes représentent le numéro de l'exemple.

La **figure IV.9** illustre l'interface de l'onglet Visualize

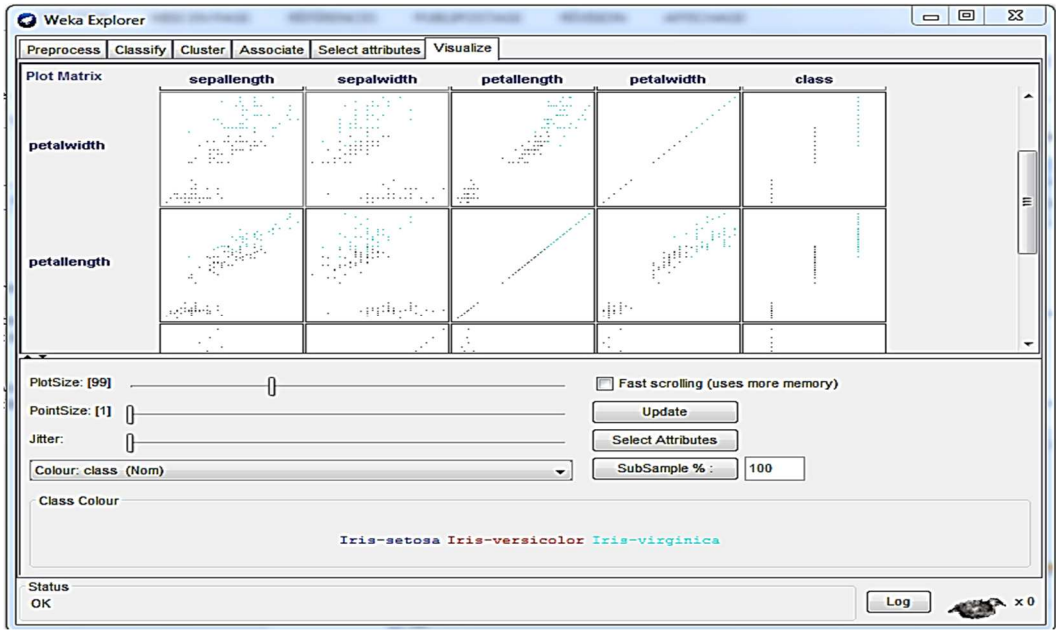


Figure IV.9 : Onglet Visualize

6.2. Experimenter

L'experimenteur permet la comparaison systématique (taxinomique) des performances prédictives des algorithmes d'apprentissage automatique de Weka sur une collection de jeux de données.

La **figure IV.10** Montre l'interface Experimenter de Weka

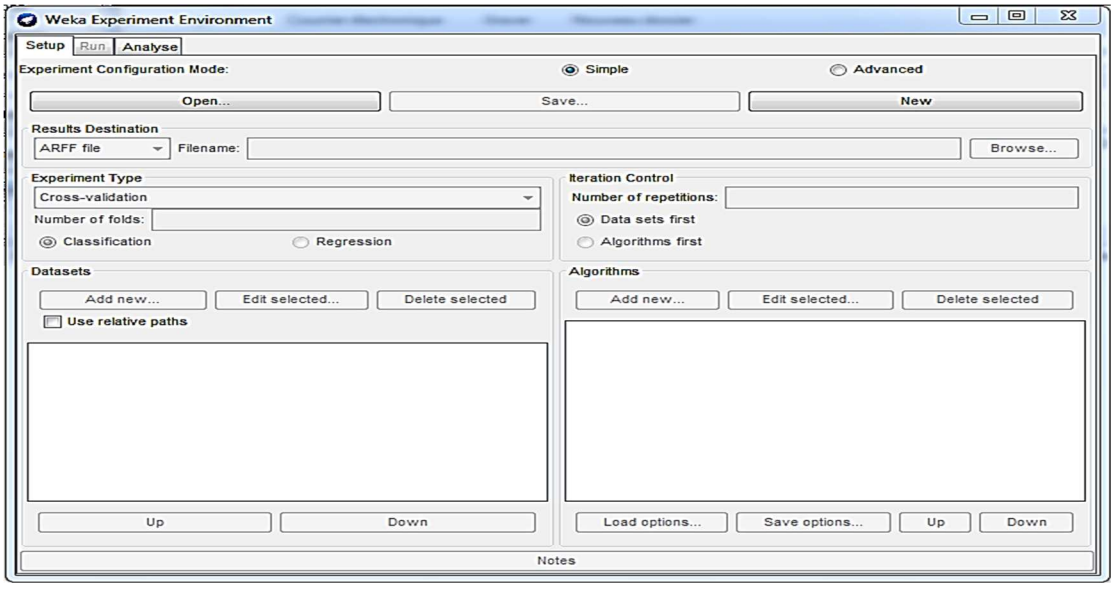


Figure IV.10 : Interface Expérimenter

7.3. Knowledge Flow

L'interface **Knowledge Flow**, ou « flux de connaissance » atteint à peu près les mêmes fonctionnalités que l'Explorer.

La figure suivante présente l'interface Knowledge Flow de Weka.

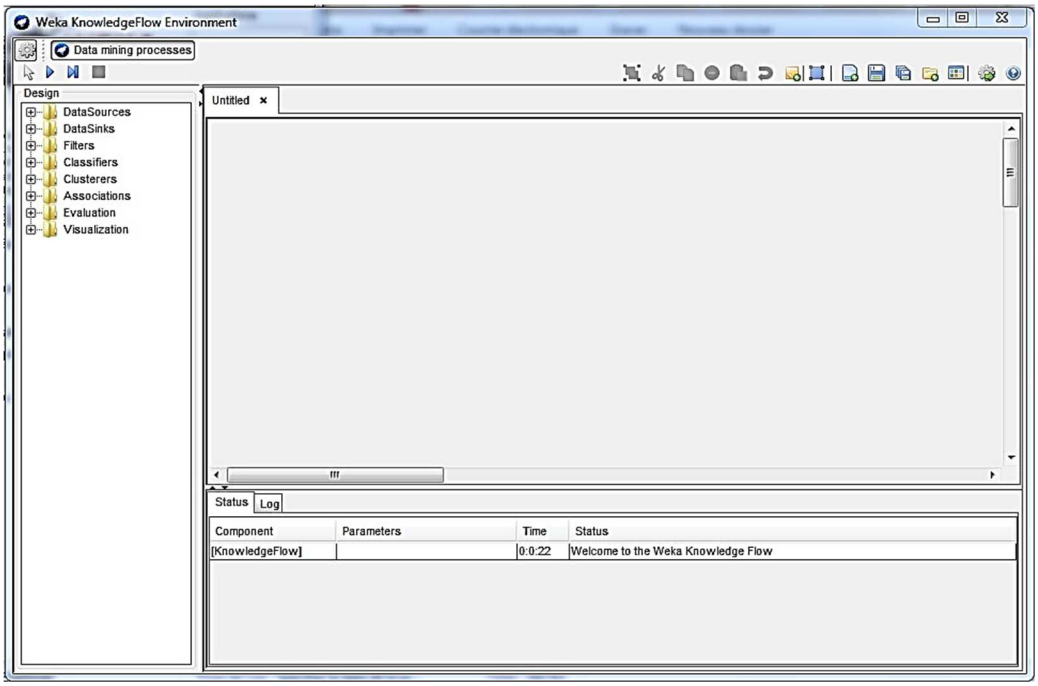


Figure IV.11: Interface Knowmedge FlowCLI

6.4. Simple CLI,

L'interface **Simple CLI**, qui est un interpréteur de ligne de commande, elle est recommandée pour une utilisation plus poussée, elle offre des fonctionnalités supplémentaires et utilise beaucoup moins de mémoire.

La figure ci-dessous illustre l'interface CLI de Weka

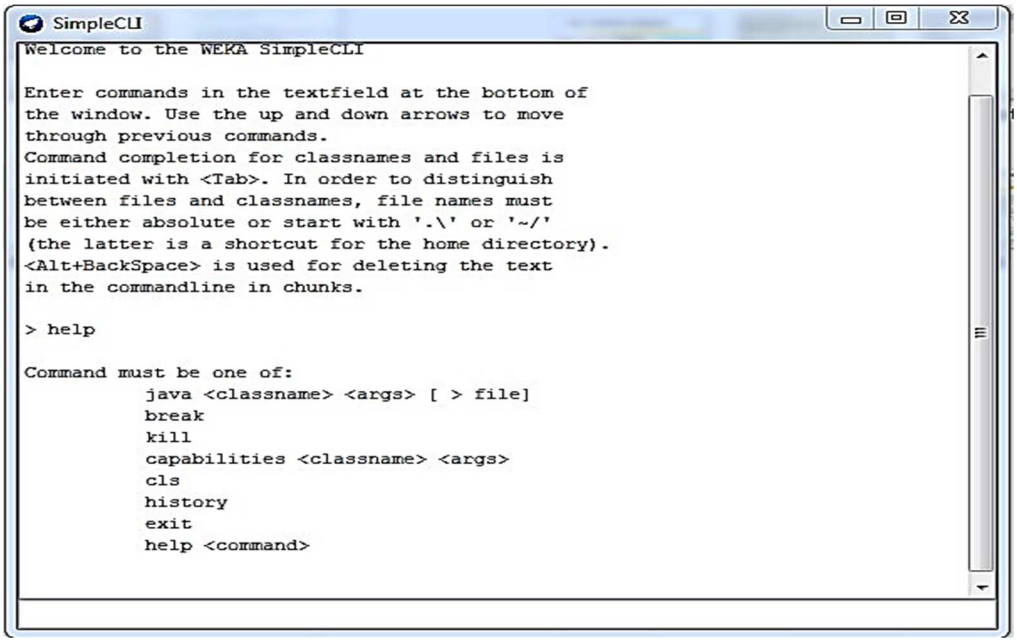


Figure IV.12 : Interface SimpleCLI

7. Classification sous weka

7.1. Algorithmes de classification

Weka implémente un très grand nombre de classificateurs (le séparateur a vaste marge, d'arbres, de réseaux neurone, etc.).

V.2. Validation de modèles

Il existe différentes méthodes de validation de modèles dans Weka, qui se trouvent sur la partie Test options de l'onglet Classify comme illustrés ci-dessous.

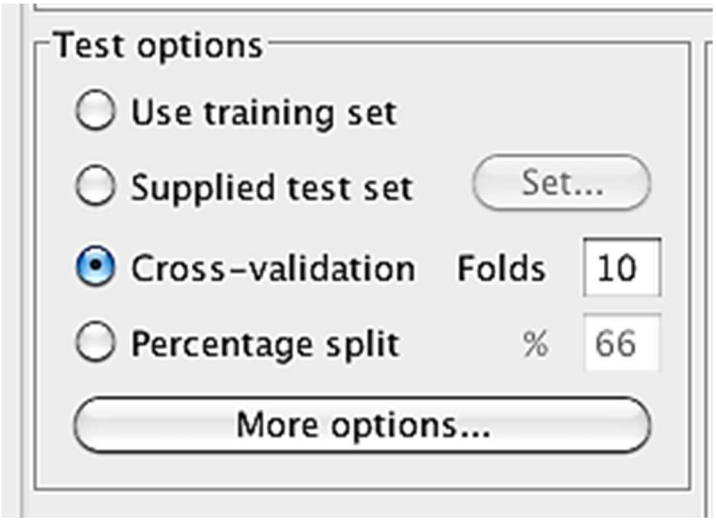


Figure IV.13 : Fenêtre Test options

- **Use training set :** qui tente d'entraîner un arbre en utilisant tous les exemples d'entraînement, comme résultat un arbre est obtenu, ainsi que le résultat de classification sur ce même ensemble de données.
- **Supplied test set** (avec le paramètre set–choix du jeu de données pour la validation) : consiste à évaluer le modèle sur un autre jeu de données (a priori différent de celui utilisé pour construire le modèle).
- **Cross-validation** (avec comme paramètre « folds ») **:** pour avoir l'effet de validation croisée, c'est-à-dire de découper les données en parties. Elle est utilisée lorsqu'on ne possèdepasdéjàunecollectionaveclessous-ensemblesd'entraînementetdetestdéjà séparés. Elle consiste à diviser les données en **n** groupes. Les modèles sont construits sur **n-1** groupes et testés sur le nième groupe. Puis le groupe de test est changé, et le même procédé est répété jusqu'à avoir réalisé toutes les combinaisons. La moyenne des validations est alors considérée comme la validation finale.
- **Percentage split** (avec comme paramètre « pourcentage ») : consiste à utiliser un certain pourcentage des données pour construire le modèle et l'autre partie pour le valider.

7.3. Processus de classification

Chaqueclassificateurasaproprelistedeparamètres.Cesclassificateurspeuventêtrelancés sur un échantillon de données préalablement chargé (et éventuellement filtré par sélection d'attributs, discrétisation ou autre).

L'onglet Classify de la fenêtre de l'exploreur Weka permet d'exécuter des processus d'apprentissage, et d'observer les résultats et les performances estimées. Pour ce faire, il faut indiquer le mode d'évaluation (cross validation, empirical evaluation, etc.), sélectionner le classificateur choisi et indiquer ses paramètres (en cliquant sur la commande), puis lancer la classification. En bas de la fenêtre, la progression de l'apprentissage apparaît.

La fenêtre de droite indique les performances estimées du type de classificateur appris. Il existe ainsi plusieurs méthodes d'estimation de ces performances (qu'il faut sélectionner puis éventuellement paramétrer), mais au final, les mêmes indicateurs sont fournis, parmi lesquels le taux de bonne classification, ou le temps moyen d'apprentissage. En réalisant un clic droit sur le processus terminé dans la liste des résultats (bas-gauche), nous pouvons accéder à la visualisation du modèle, lorsque cette dernière est faisable.

Le modèle appris peut être ensuite sauvegardé, pour être ultérieurement visualisé, et pour évidemment, être entraîné sur de nouvelles données.

La figure qui suit illustre un exemple de classification sous Weka et ses résultats.

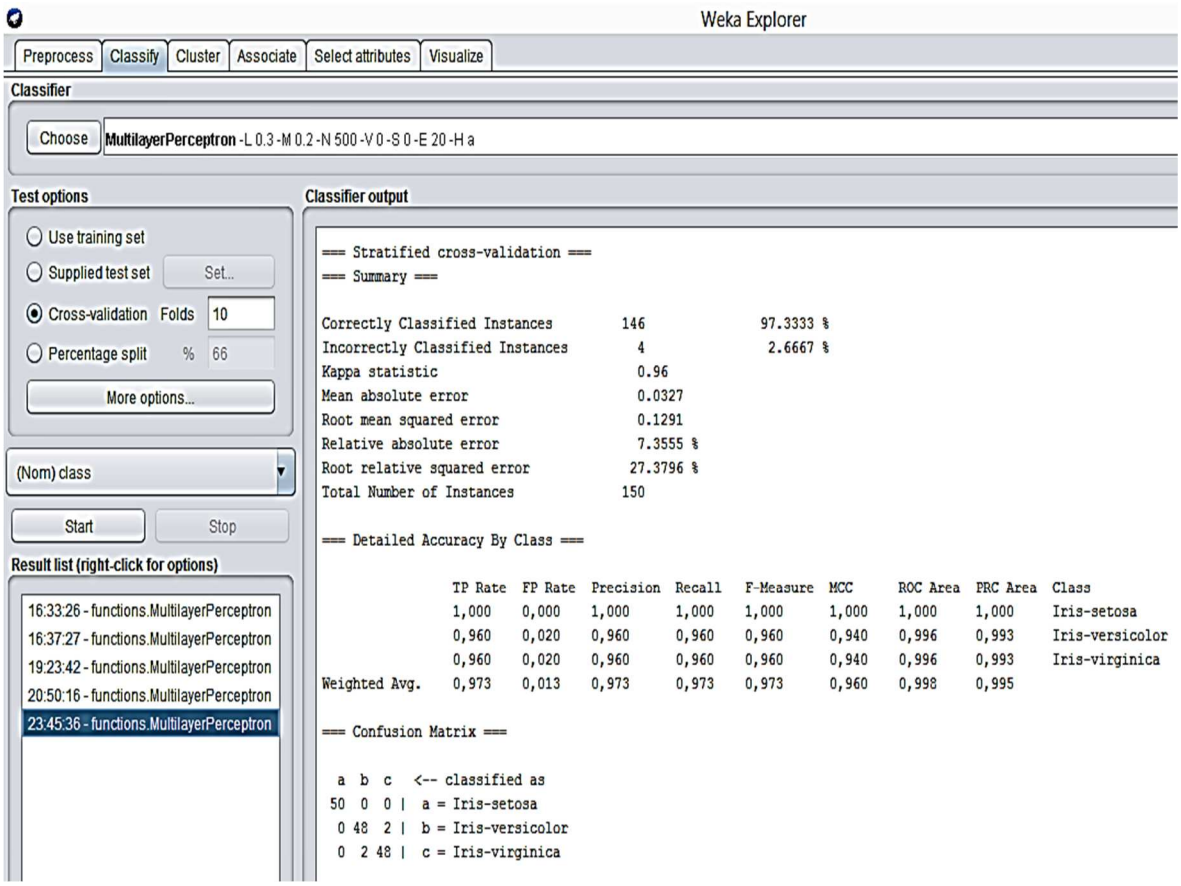


Figure IV.14 : Exemple de classification

8. Conclusion

Weka est un des très rares logiciels à proposer un dispositif assez facile d'accès grâce à son interface conviviale. Il propose beaucoup d'algorithmes classiques en apprentissage automatique (supervisé ou non), avec les fonctionnalités de prétraitement des données, analyse et évaluation des résultats.

Dans ce chapitre nous avons eu un aperçu des fonctionnalités de classification de Weka ainsi que ces méthodes de base. Le but était de nous familiariser avec ce logiciel, car bien qu'on puisse aisément comprendre certaines fonctionnalités de base de Weka, pour en faire une utilisation réellement utile, il s'avère absolument nécessaire d'avoir une compréhension des modèles mathématiques proposés, voire d'utiliser son propre modèle.

1. Introduction

Ce chapitre, consiste à étudier et appliquer les différentes méthodes d'apprentissage supervisé et non supervisé à travers l'outil de classification Weka et analyser les résultats afin d'évaluer les performances de chaque méthode.

Ce présent chapitre est subdivisé en deux parties. La première est réservée pour l'étude comparative des différents algorithmes d'apprentissage supervisé à savoir : le séparateur à vaste marge**(SVM),** le perceptron multicouche **(PMC)** du réseau de neurone et L'arbre de décision**(AD)** en utilisant Weka.

La deuxième partie a pour but de comparer les méthodes de classification non supervisée à savoir **K-means** et **clustering hiérarchique.**

La figure suivante montre la démarche qu'on a suivie durant l'expérimentation :

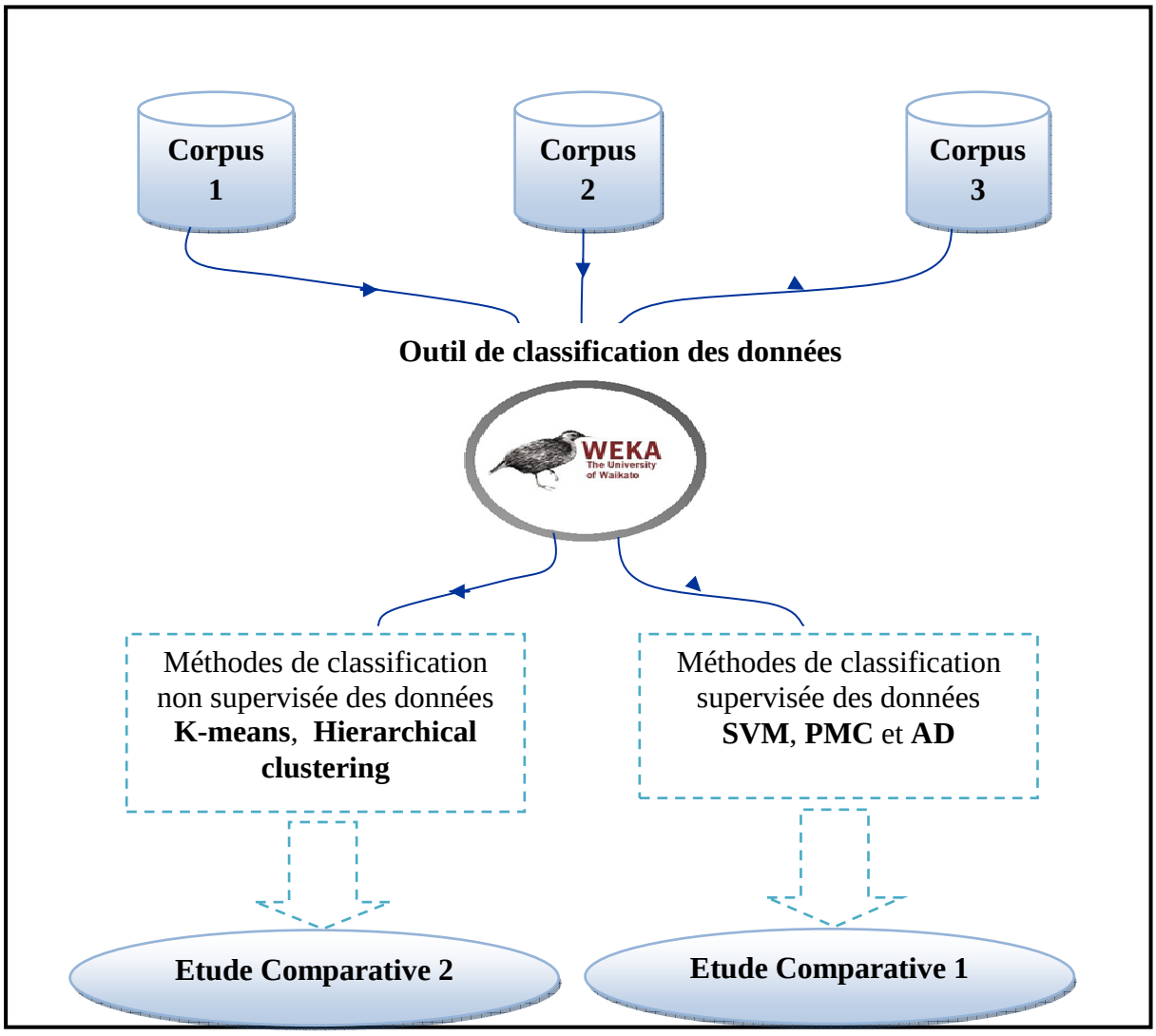


Figure V.1 : schéma synoptique de notre étude

2. Collection des données.

On a recueilli l'ensemble des jeux de données dans l'entrepôt de « UCI Machine Learning Repository » **[44]** qui fournit un ensemble destiné à l'apprentissage artificiel et à la fouille de données.

Le tableau ci-après montre les caractéristiques des bases choisies:

	Nombre d'instances	**Nombre d' attributs**	**Type d' attributs**	**Nombre de classes**	**Description**
Glass.arff	214	10	Réel	7	L'ensemble de données décrit les composants des verres de fabrications de certains éléments à base de vitre.
Mfeat_morphology.arff	2000	7	Réel	10	Cet ensemble de données se compose des caractéristiques de chiffres manuscrits (0-9) extrait d'une collection de cartes de services publics néerlandais.
page-blocks.arff	5473	11	Entier et réel	5	Cette base contient le classement de blocs de mise en page d'un document séparant la zone de textes de la zone graphique.

Tableau V.1 : caractéristiques des tableaux de données

La taille variable de la base va permettre de voir est-ce qu'ils donneront tous les mêmes résultats, peu importe les données traitées pour les comparer par la suite.

3. Représentation des données

Les données (classes, attributs) sont stockées dans des fichiers texte au format arff (Attribute-Relation File Format).

Dans ce qui suit, nous allons décrire les deux études effectuées sur la classification supervisée et non supervisée.

4. Descriptions des deux études

4.1. Etude 1 : La classification supervisée

Dans cette partie, on va s'intéresser aux différents algorithmes d'apprentissage supervisé à savoir : le séparateur à vaste marge **(SVM),** le perceptron multicouche **(PMC) du** réseau de neurone et L'arbre de décision**(AD)**.

1. Le séparateur à vaste marge (SVM)

Le principe est de trouver un classifieur, ou une fonction de discrimination, dont la capacité de généralisation (qualité de prévision) est la plus grande possible. Les exemples sont représentés par des points dans un espace et on cherche un (hyper) plan séparant au mieux les classes et que toutes les observations soient le plus éloigné de ce plan. On choisira la fonction **SMO** dans Weka pour cette tâche.

2. Le perceptron multicouche (PMC)

L'objectif est de prédire la classe des futures observations en mesurant le taux d'erreurs (ou le coût) entre les sorties désirées et les sorties observées par l'algorithme c'est-à-dire les risques empiriques et les risques réels. On va opter pour la fonction **Multilayer perceptron** dans Weka pour réaliser cette expérimentation**.**

3. L'arbre de décision(AD)

L'arbre de décision est une représentation graphique d'une procédure de classification. Dans Weka, cet algorithme est représenté par la fonction **J48**.

a. Expérimentations et résultats

Weka est un logiciel libre qui implémente un ensemble d'algorithmes d'apprentissage. Le choix de cet outil est dû au fait qu'il est très utilisé dans le domaine de l'apprentissage artificiel et de la fouille de données et il est aussi facile à manipuler. Il est compatible avec le format de données qu'on a choisi.

Comme on a déjà mentionné ci-dessus, on va s'intéresser à la classification supervisée. Dans Weka, on peut trouver tous les algorithmes correspondant dans l'onglet classify. Après avoir chargé les données dans l'outil, on applique les différentes fonctions destinées à cette étude. Weka offre quatre options pour faire l'évaluation de la performance du modèle appris, dans cette expérience, on ne va se pencher que sur l'une d'elle. À savoir, la validation croisée

(cross validation), l'ensemble d'apprentissage va être subdivisé en 10 parties (fold =10); l'algorithme va apprendre 10 fois sur 9 parties et la dernière partie sert à évaluer le modèle, puis les 10 évaluations sont combinées. La figure ci-après illustre l'onglet et les options de tests.

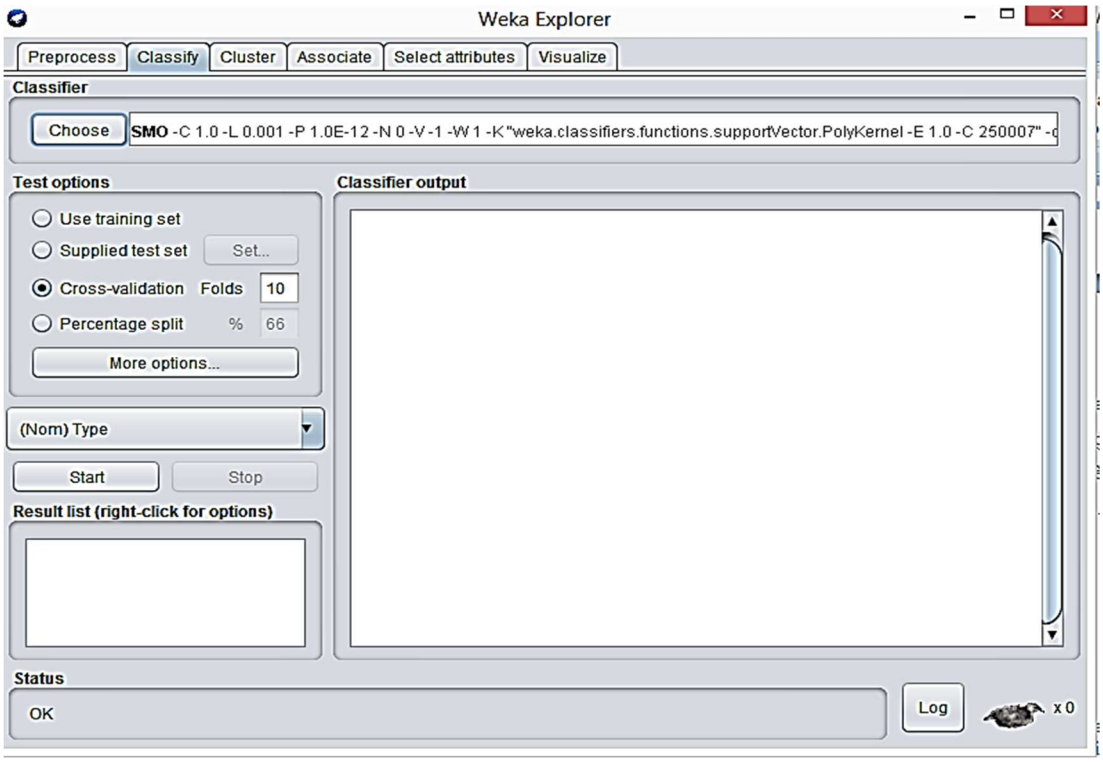


Figure V.2 : l'ongle Weka explorer

Les résultats de cette expérience de classification supervisée pour chaque algorithme et dans chaque collection de donnée sont représentés dans ce qui suit :

A- **La collection :** Glass.href

➢ Algorithme SVM (SMO sous Weka)

La **figure V.3** représente les résultats de la classification avec l'algorithme **SVM.**

```
Time taken to build model: 0.27 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         120               56.0748 %
Incorrectly Classified Instances        94               43.9252 %
Kappa statistic                          0.3563
Mean absolute error                      0.2137
Root mean squared error                  0.3166
Relative absolute error                100.938  %
Root relative squared error             97.5567 %
Total Number of Instances              214

=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               0,500    0,167    0,593      0,500    0,543      0,350    0,766     0,522     build wind float
               0,803    0,478    0,480      0,803    0,601      0,316    0,672     0,462     build wind non-float
               0,000    0,000    0,000      0,000    0,000      0,000    0,637     0,106     vehic wind float
               0,000    0,000    0,000      0,000    0,000      0,000    ?         ?         vehic wind non-float
               0,154    0,005    0,667      0,154    0,250      0,302    0,901     0,415     containers
               0,000    0,000    0,000      0,000    0,000      0,000    0,538     0,156     tableware
               0,759    0,016    0,880      0,759    0,815      0,791    0,903     0,763     headlamps
Weighted Avg.  0,561    0,227    0,524      0,561    0,517      0,352    0,739     0,479

=== Confusion Matrix ===

  a  b  c  d  e  f  g   <-- classified as
 35 35  0  0  0  0  0 |  a = build wind float
 14 61  0  0  1  0  0 |  b = build wind non-float
  8  9  0  0  0  0  0 |  c = vehic wind float
  0  0  0  0  0  0  0 |  d = vehic wind non-float
  0 10  0  0  2  0  1 |  e = containers
  0  7  0  0  0  0  2 |  f = tableware
  2  5  0  0  0  0 22 |  g = headlamps
```

Figure V.3 : classification avec l'algorithme SVM pour le corpus Glass

D'après cette figure, on voit que seulement 56.075% des instances ont été classées correctement. La matrice de confusion, indique que les classes ont toutes des erreurs, plus particulièrement les classes vehic wind float, vehic wind non-float, tableware qui n'ont aucune instance correctement classée. La classe build wind float a 35 instances sur 70 correctement classée. La classse build wind non- float a 60 instances sur 76 correctement classée. Et la classe headlamps a 22 instances sur 29 correctement classée.

Le tableau suivant présente les mesures d'exactitude par classe pour la méthode SVM, on les trouve dans la partie « Detailed Accuracy By Class ».

Classe	TP Rate	FP Rate	Precision	Recall	F-Measure
build wind float	0,500	0,167	0,593	0,500	0,543
build wind non-float	0.803	0.478	0.480	0,803	0,601
vehic wind float	0.000	0.000	0.000	0,000	0,000
vehic wind non-float	0.000	0,000	0,000	0,000	0,000
containers	0.154	0,005	0,667	0.154	0,250
tableware	0.000	0.000	0.000	0.000	0.000
headlamps	0.759	0.016	0.880	0.759	0.815

Tableau V.2 : Mesures d'exactitude par classe pour la méthode SVM pour le corpus Glass

Ci-dessus la définition de chaque mesure:

TP Rate : c'est le rapport des vrais positifs. Il correspond à :

Nombre de vrais positif / Nombre d'exemple de cette classe

C'est donc le rapport entre le nombre de bien classé et le nombre total d'éléments qui devraient être bien classé.

- FP Rate : *Nombre de faux positifs / Nombre d'exemples n'étant pas de cette classe*

Les données des taux TP Rate et FP Rate permettent de reconstruire la matrice de confusion pour une classe donnée. Symétriquement, la matrice de confusion permet de calculer TP Rate et FP Rate.

- Precision : c'est le rapport entre le nombre de vrais positifs et la somme des vrais positifs et des faux positifs. Une valeur de 1 exprime le fait que tous les exemples classés positifs l'étaient vraiment.

- Recall : un Recall de 1 signifie que tous les exemples positifs ont été trouvés.

- F-Measure : cette quantité permet de regrouper en un seul nombre les performances du classifieur (pour une classe donnée).

*F-Measure = (2 * Recall * Precision) / (Recall * Precision)*

- ## Algorithme PMC (Multilayer perceptron sous Weka)

La figure IV.8 représente les résultats de la classification avec l'algorithme **PMC**

```
Time taken to build model: 0.63 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         145               67.757  %
Incorrectly Classified Instances        69               32.243  %
Kappa statistic                          0.5528
Mean absolute error                      0.1114
Root mean squared error                  0.2627
Relative absolute error                 52.5915 %
Root relative squared error             80.958  %
Total Number of Instances              214

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
               0,743     0,181     0,667       0,743   0,703      0,548    0,862     0,643     build wind float
               0,684     0,217     0,634       0,684   0,658      0,460    0,815     0,741     build wind non-float
               0,059     0,005     0,500       0,059   0,105      0,151    0,654     0,161     vehic wind float
               0,000     0,000     0,000       0,000   0,000      0,000    ?         ?         vehic wind non-float
               0,692     0,030     0,600       0,692   0,643      0,620    0,957     0,604     containers
               0,778     0,020     0,636       0,778   0,700      0,689    0,926     0,514     tableware
               0,828     0,011     0,923       0,828   0,873      0,856    0,931     0,889     headlamps
Weighted Avg.  0,678     0,141     0,671       0,678   0,659      0,537    0,847     0,665

=== Confusion Matrix ===

  a  b  c  d  e  f  g   <-- classified as
 52 16  1  0  0  0  1 |  a = build wind float
 17 52  0  0  4  3  0 |  b = build wind non-float
  8  8  1  0  0  0  0 |  c = vehic wind float
  0  0  0  0  0  0  0 |  d = vehic wind non-float
  0  3  0  0  9  0  1 |  e = containers
  0  1  0  0  1  7  0 |  f = tableware
  1  2  0  0  1  1 24 |  g = headlamps
```

Figure V.4: Classification avec l'algorithme PMC pour le corpus Glass

Dans cette figure, on remarque que 67.757% des instances ont été classés correctement. La matrice de confusion indique que les erreurs ont concerné toutes les classes. Par exemple, on voit que la classe vehic wind float a eu seulement 1 (une) instance correctement sur 9.

Le tableau suivant présente les mesures d'exactitude par classe pour la méthode PMC, on les trouve dans la partie « Detailed Accuracy By Class ».

Classe	TP Rate	FP Rate	Precision	Recall	F-Measure
build wind float	0,743	0,181	0,667	0,743	0,703
build wind non-float	0,684	0,217	0.643	0,684	0,658
vehic wind float	0,059	0,005	0.500	0,059	0,105
vehic wind non-float	0,000	0,000	0.000	0.000	0.000
containers	0,692	0,030	0,600	0,692	0.643
tableware	0,778	0,020	0,636	0,778	0,700
Headlamps	0.828	0.011	0.671	0.828	0.873

Tableau V.3 : Mesures d'exactitude par classe pour la méthode PMC pour le corpus Glass

➢ Algorithme Arbre de décision (J48 sous Weka)

La figure IV.10 illustre les résultats de classification avec l'algorithme **Arbre de décision**

```
Time taken to build model: 0.44 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         143               66.8224 %
Incorrectly Classified Instances        71               33.1776 %
Kappa statistic                          0.55
Mean absolute error                      0.1026
Root mean squared error                  0.2897
Relative absolute error                 48.4507 %
Root relative squared error             89.2727 %
Total Number of Instances              214
=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
               0,714     0,174     0,667       0,714   0,690      0,532    0,806     0,667     build wind float
               0,618     0,181     0,653       0,618   0,635      0,443    0,768     0,606     build wind non-float
               0,353     0,046     0,400       0,353   0,375      0,325    0,766     0,251     vehic wind float
               0,000     0,000     0,000       0,000   0,000      0,000    ?         ?         vehic wind non-float
               0,769     0,010     0,833       0,769   0,800      0,788    0,872     0,575     containers
               0,778     0,029     0,538       0,778   0,636      0,629    0,930     0,527     tableware
               0,793     0,022     0,852       0,793   0,821      0,795    0,869     0,738     headlamps
Weighted Avg.  0,668     0,130     0,670       0,668   0,668      0,539    0,807     0,611

=== Confusion Matrix ===

  a  b  c  d  e  f  g   <-- classified as
 50 15  3  0  0  1  1 |  a = build wind float
 16 47  6  0  2  3  2 |  b = build wind non-float
  5  5  6  0  0  1  0 |  c = vehic wind float
  0  0  0  0  0  0  0 |  d = vehic wind non-float
  0  2  0  0 10  0  1 |  e = containers
  1  1  0  0  0  7  0 |  f = tableware
  3  2  0  0  0  1 23 |  g = headlamps
```

Figure V.5 : la classification avec l'algorithme J48

Cette figure nous indique que 66.8224 % des instances ont été classés correctement. La matrice de confusion en bas, indique que les erreurs ont concerné les sept classes .

Le tableau suivant présente les mesures d'exactitude par classe pour la méthode J48, on les trouve dans la partie « Detailed Accuracy By Class ».

Classe	TP Rate	FP Rate	Precision	Recall	F-Measure
build wind float	0,714	0,174	0,667	0,714	0,690
build wind non-float	0.618	0.181	0.653	0,618	0,635
vehic wind float	0.353	0.046	0.400	0,353	0,375
vehic wind non-float	0.000	0,000	0,000	0,000	0,000
containers	0.769	0,010	0,833	0.769	0,800
tableware	0.778	0.029	0.538	0.778	0.636
headlamps	0.793	0.022	0.852	0.793	0.821

Tableau V.4 : Mesures d'exactitude par classe pour la méthode J48

Dans le tableau suivant nous allons énumérer les trois algorithmes avec le nombre d'instances correctement classifiés et le nombre d'instances incorrectement classifiés pour

chacun d'eux.

	Nombre de d'instances correctement classifiés	**Nombre de d'instances incorrectement classifiés**
KNN	145	69
Arbre de décision	143	71
SVM	120	94

Tableau V.5 : Evaluation des performances de classification pour le corpus Glass

Le tableau qui suit illustre les pourcentages d'instances correctement classifiés pour les différents algorithmes de corpus glass.

Algorithmes de classification	**Pourcentage de classification correcte**
PMC	67.757%
Arbre de décision	66.8224%
SVM	56.0748%

Tableau V.6 : Pourcentage de bonne classification pour les différents algorithmes pour le jeu de donnée glass

D'après les résultats de classification de la première collection, on remarque que la méthode PMC enregistre le meilleur pourcentage de documents correctement classifiés (67.757%) puis viens l'arbre de décision et la SVM avec 66.8224% et 56.0748% respectivement.

B. Deuxième collection Collection : Mfeat_morphology

➢ Algorithme SVM (SMO sous Weka)

La figure V.6 représente les résultats de la classification avec l'algorithme **SVM**

```
Time taken to build model: 0.91 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        1400               70      %
Incorrectly Classified Instances       600               30      %
Kappa statistic                          0.6667
Mean absolute error                      0.162
Root mean squared error                  0.2758
Relative absolute error                 89.9988 %
Root relative squared error             91.9337 %
Total Number of Instances             2000

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0,985    0,000    1,000      0,985    0,992      0,992    0,990     0,987     1
                 0,925    0,008    0,925      0,925    0,925      0,917    0,985     0,902     2
                 0,830    0,059    0,608      0,830    0,702      0,673    0,936     0,556     3
                 0,510    0,034    0,622      0,510    0,560      0,520    0,891     0,460     4
                 0,585    0,039    0,626      0,585    0,605      0,563    0,927     0,500     5
                 0,485    0,017    0,764      0,485    0,593      0,576    0,939     0,571     6
                 0,565    0,072    0,467      0,565    0,511      0,454    0,901     0,389     7
                 0,805    0,053    0,629      0,805    0,706      0,675    0,947     0,574     8
                 0,960    0,000    1,000      0,960    0,980      0,978    0,994     0,985     9
                 0,350    0,051    0,432      0,350    0,387      0,329    0,900     0,386     10
Weighted Avg.    0,700    0,033    0,707      0,700    0,696      0,668    0,941     0,631
=== Confusion Matrix ===

   a   b   c   d   e   f   g   h   i   j   <-- classified as
 197   0   3   0   0   0   0   0   0   0 |   a = 1
   0 185   0   0   6   0   0   8   0   1 |   b = 2
   0   0 166  11   2   3   2  16   0   0 |   c = 3
   0   1  17 102  34  26   0  20   0   0 |   d = 4
   0  10   2  25 117   0   1  45   0   0 |   e = 5
   0   0  72  21   4  97   0   6   0   0 |   f = 6
   0   1   0   1   0   0 113   0   0  85 |   g = 7
   0   2  10   3  24   0   0 161   0   0 |   h = 8
   0   0   0   0   0   0   2   0 192   6 |   i = 9
   0   1   3   1   0   1 124   0   0  70 |   j = 10
```

Figure V.6 : la classification avec l'algorithme **SVM**

Dans cette figure, on remarque que 70% des instances ont été classés correctement. La matrice de confusion, indique que les erreurs ont concerné toutes les classes.

Le tableau suivant présente les mesures d'exactitude par classe pour la méthode SVM, on les trouve dans la partie « Detailed Accuracy By Class ».

- **Algorithme PMC (Multilayer perceptron sous Weka)**

La figure V.7 représente les résultats de la classification avec l'algorithme **PMC**

```
Time taken to build model: 4.88 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        1499               74.95   %
Incorrectly Classified Instances       501               25.05   %
Kappa statistic                          0.7217
Mean absolute error                      0.0619
Root mean squared error                  0.1846
Relative absolute error                 34.4065 %
Root relative squared error             61.5325 %
Total Number of Instances             2000
```

```
=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0,985    0,000    1,000      0,985    0,992      0,992    0,997     0,994     1
                 0,940    0,009    0,917      0,940    0,928      0,920    0,988     0,932     2
                 0,740    0,026    0,759      0,740    0,749      0,722    0,946     0,712     3
                 0,555    0,032    0,657      0,555    0,602      0,564    0,903     0,582     4
                 0,740    0,044    0,649      0,740    0,692      0,657    0,952     0,696     5
                 0,730    0,023    0,777      0,730    0,753      0,726    0,947     0,779     6
                 0,190    0,023    0,475      0,190    0,271      0,255    0,937     0,475     7
                 0,860    0,029    0,768      0,860    0,811      0,791    0,986     0,886     8
                 0,970    0,000    1,000      0,970    0,985      0,983    0,995     0,988     9
                 0,785    0,091    0,491      0,785    0,604      0,568    0,923     0,461     10
Weighted Avg.    0,750    0,028    0,749      0,750    0,739      0,718    0,957     0,751
```

```
=== Confusion Matrix ===
```

=== Confusion Matrix ===

```
   a   b   c   d   e   f   g   h   i   j   <-- classified as
 197   0   1   0   0   1   0   1   0   0 |   a = 1
   0 188   0   0   7   0   0   5   0   0 |   b = 2
   0   1 148  13   9  18   2   9   0   0 |   c = 3
   0   1  12 111  41  21   0  14   0   0 |   d = 4
   0  10   2  20 148   0   0  20   0   0 |   e = 5
   0   0  26  21   5 146   0   2   0   0 |   f = 6
   0   1   0   1   0   0  38   0   0 160 |   g = 7
   0   3   4   3  18   0   0 172   0   0 |   h = 8
   0   0   0   0   0   0   3   0 194   3 |   i = 9
   0   1   2   0   0   2  37   1   0 157 |   j = 10
```

Figure V.7 : classification avec l'algorithme PMC

Dans cette figure, on remarque que 74.95% des instances ont été classés correctement.

La matrice de confusion, indique que les erreurs ont concerné toutes les classes.

Le tableau suivant présente les mesures d'exactitude par classe pour la méthode PMC, on les trouve dans la partie « Detailed Accuracy By Class ».

- **Algorithme Arbre de décision (J48 sous Weka)**

La figure V.8 illustre les résultats de classification avec l'algorithme Arbre de décision

```
Time taken to build model: 0.17 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        1441               72.05    %
Incorrectly Classified Instances       559               27.95    %
Kappa statistic                          0.6894
Mean absolute error                      0.0647
Root mean squared error                  0.1999
Relative absolute error                 35.9588 %
Root relative squared error             66.6367 %
Total Number of Instances             2000

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
               0,985     0,001     0,995       0,985   0,990      0,989    0,992     0,986     1
               0,920     0,012     0,898       0,920   0,909      0,898    0,952     0,831     2
               0,715     0,048     0,622       0,715   0,665      0,627    0,855     0,538     3
               0,525     0,043     0,574       0,525   0,548      0,501    0,834     0,464     4
               0,700     0,042     0,651       0,700   0,675      0,638    0,868     0,561     5
               0,585     0,027     0,709       0,585   0,641      0,609    0,923     0,588     6
               0,965     0,109     0,496       0,965   0,655      0,649    0,935     0,480     7
               0,815     0,023     0,795       0,815   0,805      0,783    0,920     0,730     8
               0,985     0,001     0,995       0,985   0,990      0,989    0,997     0,995     9
               0,010     0,006     0,167       0,010   0,019      0,017    0,917     0,444     10
Weighted Avg.  0,721     0,031     0,690       0,721   0,690      0,670    0,919     0,662

=== Confusion Matrix ===

   a   b   c   d   e   f   g   h   i   j   <-- classified as
 197   0   2   0   0   1   0   0   0   0 |   a = 1
   0 184   2   1   7   0   0   5   0   1 |   b = 2
   1   3 143  21   8  14   2   8   0   0 |   c = 3
   0   2  17 105  34  27   0  13   0   2 |   d = 4
   0  11   7  24 140   2   1  15   0   0 |   e = 5
   0   0  48  24   9 117   0   1   0   1 |   f = 6
   0   1   0   1   0   0 193   0   0   5 |   g = 7
   0   3   8   7  17   2   0 163   0   0 |   h = 8
   0   0   0   0   0   0   2   0 197   1 |   i = 9
   0   1   3   0   0   2 191   0   1   2 |   j = 10
```

Figure V.8 : classification avec l'algorithme Arbre de décision

Dans cette figure, on remarque que 72.05% des instances ont été classés correctement.

La matrice de confusion, indique que les erreurs ont concerné toutes les classes.

C. Troisième Collection : page-blocks

- **Algorithme SVM (SMO sous Weka)**

La figure V.9 représente les résultats de la classification avec l'algorithme **SVM**

```
Time taken to build model: 0.19 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        5086               92.9289 %
Incorrectly Classified Instances       387                7.0711 %
Kappa statistic                          0.4677
Mean absolute error                      0.2437
Root mean squared error                  0.322
Relative absolute error                319.9351 %
Root relative squared error            165.2368 %
Total Number of Instances             5473
=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Cla
                 0,998    0,666    0,929      0,998    0,962      0,538    0,726     0,941     1
                 0,492    0,002    0,931      0,492    0,644      0,664    0,785     0,496     2
                 0,429    0,000    0,923      0,429    0,585      0,628    0,680     0,456     3
                 0,023    0,000    1,000      0,023    0,044      0,150    0,949     0,438     4
                 0,070    0,000    0,889      0,070    0,129      0,246    0,841     0,126     5
Weighted Avg.    0,929    0,598    0,930      0,929    0,909      0,534    0,736     0,887

=== Confusion Matrix ===

    a    b    c    d    e   <-- classified as
 4902   11    0    0    0 |    a = 1
  166  162    1    0    0 |    b = 2
   16    0   12    0    0 |    c = 3
   84    1    0    2    1 |    d = 4
  107    0    0    0    8 |    e = 5
```

Figure V.9 : la classification avec l'algorithme **SVM**

Dans cette figure, on remarque que 92.93% des instances ont été classés correctement.

La matrice de confusion, indique que les erreurs ont concerné toutes les classes.

- **Algorithme PMC (Multilayer perceptron sous Weka)**

La figure V.10 représente les résultats de la classification avec l'algorithme **PMC**

```
Time taken to build model: 7.33 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        5267               96.2361 %
Incorrectly Classified Instances       206                3.7639 %
Kappa statistic                          0.785
Mean absolute error                      0.02
Root mean squared error                  0.1103
Relative absolute error                 26.2328 %
Root relative squared error             56.5978 %
Total Number of Instances             5473

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0,990    0,257    0,971      0,990    0,980      0,795    0,968     0,995     1
                 0,830    0,006    0,904      0,830    0,865      0,858    0,981     0,904     2
                 0,429    0,001    0,750      0,429    0,545      0,565    0,773     0,503     3
                 0,705    0,001    0,939      0,705    0,805      0,811    0,980     0,846     4
                 0,504    0,005    0,699      0,504    0,586      0,586    0,933     0,605     5
Weighted Avg.    0,962    0,231    0,960      0,962    0,960      0,793    0,967     0,976

=== Confusion Matrix ===

    a    b    c    d    e   <-- classified as
 4862   27    3    3   18 |    a = 1
   53  273    0    1    2 |    b = 2
   13    0   12    0    3 |    c = 3
   23    1    0   62    2 |    d = 4
   55    1    1    0   58 |    e = 5
```

Figure V.10 : la classification avec l'algorithme PMC

- **Algorithme Arbre de décision (J48 sous Weka)**

La figure V.11 illustre les résultats de classification avec l'algorithme Arbre de décision

```
Time taken to build model: 0.09 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        5302               96.8756 %
Incorrectly Classified Instances       171                3.1244 %
Kappa statistic                          0.8319
Mean absolute error                      0.0162
Root mean squared error                  0.1068
Relative absolute error                 21.2092 %
Root relative squared error             54.8294 %
Total Number of Instances             5473
=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0,986    0,164    0,981      0,986    0,984      0,839    0,940     0,982     1
                 0,888    0,007    0,896      0,888    0,892      0,885    0,947     0,822     2
                 0,821    0,001    0,793      0,821    0,807      0,806    0,964     0,786     3
                 0,898    0,002    0,888      0,898    0,893      0,891    0,985     0,793     4
                 0,539    0,005    0,681      0,539    0,602      0,599    0,836     0,468     5
Weighted Avg.    0,969    0,148    0,967      0,969    0,968      0,837    0,939     0,958

=== Confusion Matrix ===

    a    b    c    d    e   <-- classified as
 4846   30    6    4   27 |    a = 1
   31  292    0    4    2 |    b = 2
    5    0   23    0    0 |    c = 3
    8    1    0   79    0 |    d = 4
   48    3    0    2   62 |    e = 5
```

Figure V.11 : la classification avec l'algorithme Arbre de décision

Le tableau suivant résume l'évaluation des performances de classification pour les trois corpus

corpus		Glass		Morphologie		Page blocks	
Nbr d'instances		214		2000		5473	
		Instance correctement classifié	Instance incorrectement classifié	Instance correctement classifié	Instance incorrectement classifié	Instance correctement classifié	Instance incorrectement classifié
PMC	Nbr	145	69	1449	501	5267	206
	%	**67.76**	32.24	**74.95**	25.05	**96.24**	3.76
AD	Nbr	143	71	1441	559	5302	171
	%	66.82	33.18	72.05	27.95	**96.88**	3.12
SVM	Nbr	120	94	1400	600	5086	387
	%	56.07	43.93	70	30	92.92	7.07

Tableau V.7 : Evaluation des performances de classification pour les trois corpus

b. **Expérimentations des résultats**

Les expériences ont été faites sur des bases de taille différente. Le taux d'erreur calculé sur l'échantillon d'apprentissage est optimisé pour les trois expérimentations, l'utilisation de la validation croisée permet d'estimer la vraie valeur de l'erreur.

Lorsqu'on l'a appliqué, au fur et à mesure que le nombre d'instances croit, les algorithmes ont mieux prédits.

Pour des données assez petites, les exemples apprennent mal puisque le taux des instances mal classées est beaucoup plus grand tandis que pour les données de grande taille, ce taux diminue.

Cette constatation conduit à conclure que plus l'exemple d'échantillon est riche, plus la précision est bonne. Ce propos est confirmé par l'utilisation de la base « page_blocks.arff » puisqu'on a 5473 instances qui est assez grand par rapport aux deux autres bases.

De ce fait, aucun des algorithmes ne permet de faire un apprentissage sur un ensemble de données petit puisque le taux d'erreur de classification est très significatif.

On peut constater que le perceptron multicouche a une bonne prédiction par rapport aux deux autres algorithmes.

L'arbre de décision n'est pas loin derrière car il peut même rivaliser avec le PMC, contrairement au PMC et SVM, sa performance est optimisé.

Dans le cas de SVM, bien qu'il prédit mieux avec un vaste nombre d'instances, les résultats qu'il offre reste faible par rapport au PMC et à l'arbre de décision, il est peut-être dû au fait que Weka ne fournit qu'un SVM linéaire ce qui peut limiter son efficacité, il y a aussi le type des données apprenantes, alors que dans la littérature, le SVM tire son avantage dans la classification des documents.

Conclusion

Plusieurs méthodes sont proposées pour le problème général de la classification. Ils différent par les mesures de proximité qu'ils utilisent, la nature des données qu'ils traitent et l'objectif final de la classification. Chacune de ces méthodes possède ses points forts et ses points faibles.

Au final, on ne peut pas affirmer qu'un tel algorithme est meilleur par rapport à un autre, le choix dépend du problème où il va être appliqué, de ses caractéristiques. Malgré cela, les méthodes PMC, l'arbre de décision et SVM, restent les meilleures dans notre cas.

Donc, le choix d'une méthode appropriée dépend fortement de l'application, la nature des données et les ressources disponibles. Une analyse attentive des données aide à bien choisir le meilleur algorithme. Il n'existe pas un algorithme qui peut répondre à toutes les demandes.

4.2. Etude 2: La classification non supervisée

Dans cette partie du chapitre on va comparais les deux méthodes de classification non supervisé à savoir **K-means**, **Hierarchic al clustering .**

1. K-means (SimpleKMeans) sous weka

K-moyennes (ou k-means en anglais) est une méthode de partitionnement de données et un problème d'optimisation combinatoire. Étant donnés des points et un entier k, le problème est de diviser les points en k clusters.

- K=2 (nombre de cluster).
- La distance utilisée est la distance euclidienne .

2. La classification hiérarchique (Hierarchical Clustering) sous weka

Construit une hiérarchie de clusters ou, en d'autres termes, une arborescence de cluster, également appelée dendrogramme.

- K=2 (nombre de cluster).
- La distance utilisée est la distance euclidienne .

a. Expérimentation et résultats

Comme on a déjà mentionné au début du chapitre on va utiliser l'outil Weka, on va s'intéresser cette fois à la classification non supervisé (clustering). Dans Weka, on peut trouver quelques algorithmes de clustering dans l'onglet cluster (simple K-Means et Hierarchical Clustering).

Après avoir chargé les données dans l'outil, on applique les différentes fonctions destinées pour cette étude. Weka offre une option pour faire l'évaluation de la performance du modèle appris, c'est l'option C**lasses to clusters evaluation**.

C**lasses to clusters evaluation**: Dans ce mode, Weka commence par ignorer l'attribut de classe et génère le clustering. Ensuite, au cours de la phase de test, il attribue des classes aux clusters, en fonction de la valeur majoritaire de l'attribut de classe dans chaque cluster. Ensuite, il calcule l'erreur de classification en fonction de cette affectation et affiche également la matrice de confusion correspondante. Un exemple de ceci pour k-means est présenté ci-dessous.

Figure ci-après illustre l'onglet et les options de tests.

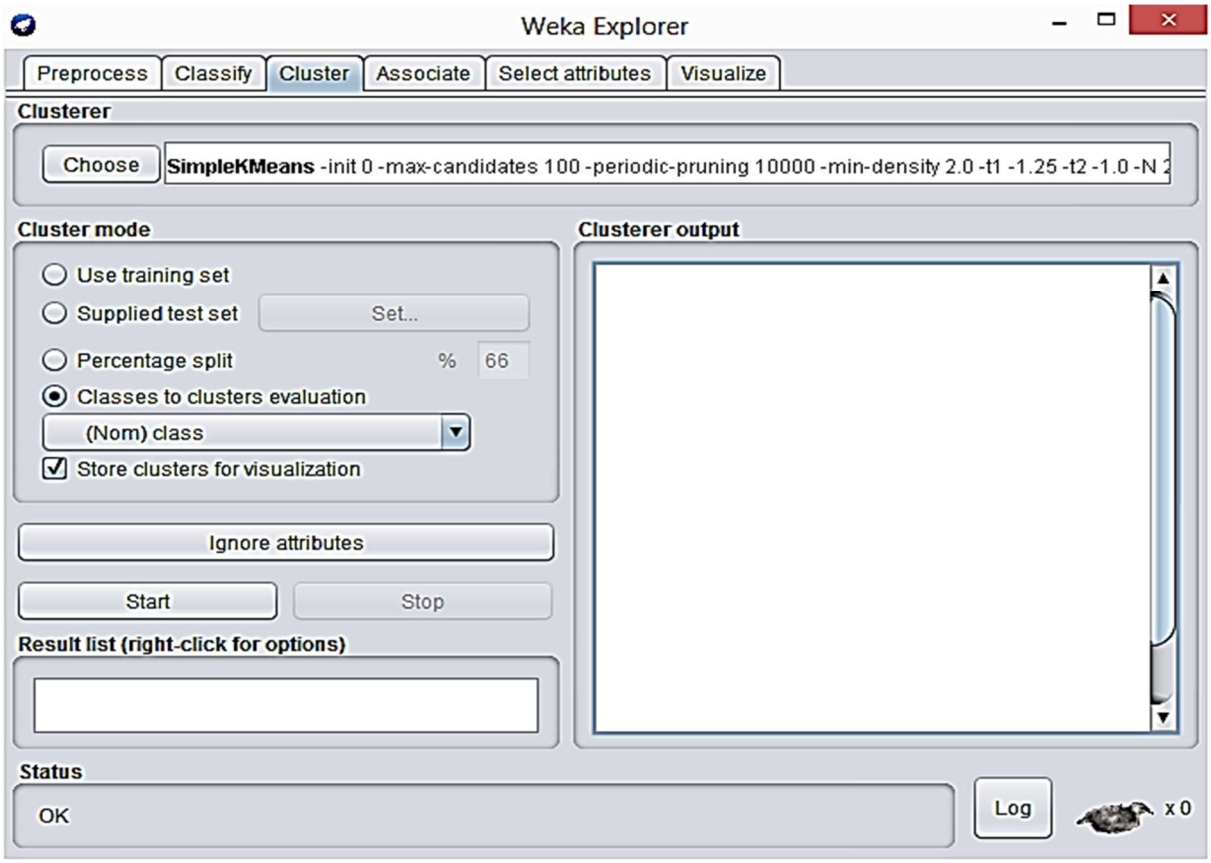


Figure V.12 : l'ongle cluster de Weka explorer

Les résultats de cette expérience de clustering pour chaque algorithme et dans chaque collection sont représentés dans ce qui suit :

A- **Premier Collection :** Glass.href

➢ Algorithme SimpleKMeans

La **figure V.13** représente les résultats de la classification avec l'algorithme **K-Means.**

```
kMeans
======

Number of iterations: 13
Within cluster sum of squared errors: 34.13433421599164

Initial starting points (random):

Cluster 0: 1.52152,13.05,3.65,0.87,72.32,0.19,9.85,0,0.17
Cluster 1: 1.51618,13.53,3.55,1.54,72.99,0.39,7.78,0,0

Missing values globally replaced with mean/mode

Final cluster centroids:
                        Cluster#
Attribute    Full Data          0          1
                (214.0)    (162.0)     (52.0)
=============================================
RI               1.5184     1.5181     1.5191
Na              13.4079    13.2811    13.8027
Mg               2.6845     3.4541     0.2871
Al               1.4449     1.3104      1.864
Si              72.6509    72.6122    72.7715
K                0.4971     0.4957     0.5013
Ca                8.957     8.6236     9.9954
Ba                0.175      0.028     0.6333
Fe                0.057     0.0605     0.0462

Time taken to build model (full training data) : 0.3 seconds

=== Model and evaluation on training set ===

Clustered Instances

0      162 ( 76%)
1       52 ( 24%)


Class attribute: Type
Classes to Clusters:

  0  1  <-- assigned to cluster
 70  0 | build wind float
 65 11 | build wind non-float
 17  0 | vehic wind float
  0  0 | vehic wind non-float
  1 12 | containers
  5  4 | tableware
  4 25 | headlamps

Cluster 0 <-- build wind float
Cluster 1 <-- headlamps

Incorrectly clustered instances :       119.0    55.6075 %
```

Figure V.13 résultat de la classification Simple K-means sur le corpus Glasse

En utilisant l'algorithme **Simple K-Means**, Il est possible de classer les instances du corpus « Glass ». Parmi le total de 214 instances, le nombre d'instances classés correctement est (95 éléments) indique une précision de 44.39%, les 119 instances restantes (ou 55,61%) étant classés de manière incorrecte. Les résultats obtenus en utilisant l'algorithme Simple K-means, fournissent une faible précision avec une vitesse de traitement élevée.

- **Algorithme Hierarchic al clustering**

La figure V.14 représente les résultats de la classification avec l'algorithme **Hierarchical clustering**

```
Time taken to build model (full training data) : 0.26 seconds

=== Model and evaluation on training set ===

Clustered Instances

0      212 ( 99%)
1        2 (  1%)


Class attribute: Type
Classes to Clusters:

  0  1  <-- assigned to cluster
 70  0 | build wind float
 76  0 | build wind non-float
 17  0 | vehic wind float
  0  0 | vehic wind non-float
 11  2 | containers
  9  0 | tableware
 29  0 | headlamps

Cluster 0 <-- build wind non-float
Cluster 1 <-- containers

Incorrectly clustered instances :       136.0    63.5514 %
```

Figure V.14 : résultat de la classification avec l'algorithme Hierarchical clustering pour le corpus Glass

Parmi le total de 214 instances, le nombre d'instances classés correctement est (78 instances) indique une précision de 36.45%, les 136 instances restantes (ou 63.55%) étant classés de manière incorrecte. Les résultats obtenus en utilisant l'algorithme Hierarchical clustering fournissent une faible précision avec une vitesse de traitement élevée.

Dans le tableau suivant nous allons énumérer les deux algorithmes avec le nombre d'instances correctement cluster et le nombre d'instances incorrectement cluster et le temps d'exécution pour chacun d'eux.

	Nombre de d'instances correctement cluster	**Nombre de d'instances incorrectement cluster**	**Temps d'exécution (en seconds)**
K-Means	95	119	0.30
Hierarchical clustering	78	136	0.72

Tableau V.8 : Evaluation des performances de clustering pour le corpus Glass

Le tableau qui suit illustre les pourcentages de documents correctement classifiés pour les différents algorithmes pour le corpus Glass

	Pourcentage des instances correctement classifiés	Pourcentage des instances incorrectement classifiés	Temps d'excution (s)
K-Means	44.39	55.61	0.30
Hierarchical clustering	36.43	63.57	0.72

Tableau V.9 : Pourcentage de bonne classification pour les différents algorithmes pour le corpus glass

B- Premier Collection : Mfeat_morphology.arff

- **Algorithme SimpleKMeans**

La figure IV.15 représente les résultats de la classification avec l'algorithme **K-Means**

```
kMeans
======

Number of iterations: 9
Within cluster sum of squared errors: 233.40205269441583

Initial starting points (random):

Cluster 0: 0,2,0,205.324861,2.06861,12950.392482
Cluster 1: 0,2,0,211.320861,2.09846,13706.803401

Missing values globally replaced with mean/mode

Final cluster centroids:
                        Cluster#
Attribute   Full Data         0         1
             (2000.0)   (789.0)  (1211.0)
=========================================
att1           0.4925    1.2433    0.0033
att2           1.7335    0.5209    2.5235
att3           0.7135    1.0051    0.5235
att4         156.4189   136.344  169.4982
att5           1.6396    1.3907    1.8017
att6        6155.2004 2919.4193 8263.4014

Time taken to build model (full training data) : 0.05 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       789 ( 39%)
1      1211 ( 61%)


Class attribute: class
Classes to Clusters:

   0   1  <-- assigned to cluster
 197   3 | 1
   1 199 | 2
   2 198 | 3
   0 200 | 4
   1 199 | 5
   0 200 | 6
 198   2 | 7
   0 200 | 8
 196   4 | 9
 194   6 | 10

Cluster 0 <-- 7
Cluster 1 <-- 4

Incorrectly clustered instances :       1602.0    80.1     %
```

Figure V.15 : résultats de la classification avec l'algorithme simple K-Means pour le corpus morphology

Parmi le total de 2000 instances, le nombre d'instances classés correctement est (398 instances) indique une précision de 19.9%, les 1602 instances restantes (ou 80.1%) étant classés de manière incorrecte. Les résultats obtenus en utilisant l'algorithme K-Means clustering fournissent une faible précision avec une vitesse de traitement élevée.

- **Algorithme Hierarchical clustering**

La figure V.16 représente les résultats de la classification avec l'algorithme **Hierarchical clustering**

```
Time taken to build model (full training data) : 17.84 seconds

=== Model and evaluation on training set ===

Clustered Instances

0      1808 ( 90%)
1       192 ( 10%)


Class attribute: class
Classes to Clusters:

   0   1  <-- assigned to cluster
 200   0 | 1
 200   0 | 2
 200   0 | 3
 200   0 | 4
 200   0 | 5
 200   0 | 6
 200   0 | 7
 200   0 | 8
   8 192 | 9
 200   0 | 10

Cluster 0 <-- 1
Cluster 1 <-- 9

Incorrectly clustered instances :       1608.0   80.4    %
```

Figure V.16 : la classification avec l'algorithme Hierarchical clustering pour le corpus morphology

Parmi 2000 instances, le nombre d'instances classés correctement (392 éléments) indique une précision de 19.6%, les 1608 éléments restants (ou 80.4%) étant classés de manière incorrecte. Les résultats obtenus en utilisant l'algorithme fournissent une faible précision avec un temps de traitement élevée.

Dans le tableau suivant nous allons énumérer les deux algorithmes avec le nombre d'instances correctement classé et le nombre d'instances incorrectement cluster et le temps d'exécution pour chacun d'eux.

	Nombre de d'instances correctement cluster	**Nombre de d'instances incorrectement cluster**	**Temps d'exécution (en seconds)**
K-Means	95	119	0.30
Hierarchical clustering	78	136	0.72

Tableau V.10: Evaluation des performances de clustering pour le corpus Mfeat_morphology

Le tableau qui suit illustre les pourcentages d'instances correctement classifiés pour les différents algorithmes pour le corpus morphology

	Nombre de d'instances correctement classifiés	Nombre de d'instances incorrectement classifiés	Temps d'excution (s)
K-Means	44.39	55.61	0.30
Hierarchical clustering	36.43	63.57	17.84

Tableau V.11 : Evaluation des performances de clustering pour le corpus morphology

C. Troisième collection page blocks :

- **Algorithme k-Means**

```
kMeans
======

Number of iterations: 42
Within cluster sum of squared errors: 443.2144049693578

Initial starting points (random):

Cluster 0: 8,24,192,3,0.328,0.625,1.75,63,120,36
Cluster 1: 1,206,206,206,0.99,1,204,204,206,1

Missing values globally replaced with mean/mode

Final cluster centroids:
                        Cluster#
Attribute    Full Data          0          1
             (5473.0)   (1089.0)   (4384.0)
=========================================
height         10.4732    16.7612     8.9113
lenght         89.5682   267.2167    45.4398
area         1198.4056 4218.8724   448.1118
eccen           13.754    34.6129     8.5725
p_black          0.3686     0.2521     0.3976
p_and            0.7851     0.5952     0.8322
mean_tr          6.2193    11.9513     4.7954
blackpix       365.9308   1076.831   189.3408
blackand       741.1082 2232.1644   370.7249


Time taken to build model (full training data) : 0.28 seconds

=== Model and evaluation on training set ===

Clustered Instances

0      1089 ( 20%)
1      4384 ( 80%)


Class attribute: class
Classes to Clusters:

    0    1  <-- assigned to cluster
  929 3984 | 1
   81  248 | 2
   10   18 | 3
    3   85 | 4
   66   49 | 5

Cluster 0 <-- 2
Cluster 1 <-- 1

Incorrectly clustered instances :       1408.0   25.7263 %
```

Figure V.17 : résultats de la classification avec l'algorithme simple KMeans pour le corpus pageblocks

Parmi 5473 instances, le nombre d'instances classés correctement (4065 éléments) indique une précision de 74.27%, les 1408 éléments restants (ou 25.73%) étant classés de manière incorrecte. Les résultats obtenus en utilisant l'algorithme k-means clustering fournissent une très bonne précision avec une très longue vitesse de traitement.

- **Algorithme hirarchical clustering**

La **figure V.18** représente les résultats de la classification avec l'algorithme **Hierarchical clustering**

```
=== Clustering model (full training set) ===

Cluster 0
((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((

Time taken to build model (full training data) : 372.02 seconds

=== Model and evaluation on training set ===

Clustered Instances

0      5472 (100%)
1         1 (  0%)


Class attribute: class
Classes to Clusters:

    0    1  <-- assigned to cluster
 4913    0 | 1
  328    1 | 2
   28    0 | 3
   88    0 | 4
  115    0 | 5

Cluster 0 <-- 1
Cluster 1 <-- 2

Incorrectly clustered instances :        559.0        10.2138 %
```

Figure V.18 : les résultats de la classification avec l'algorithme Hierarchical clustering pour le corpus pageblocks

Parmi 5473 instances, le nombre d'instances classés correctement (4914éléments) indique une précision de 89.79%, les 559 éléments restants (ou 10.21%) étant classés de manière incorrecte. Les résultats obtenus en utilisant l'algorithme Hierarchical clustering fournissent une très bonne précision avec une très longue vitesse de traitement.

Dans le tableau suivant nous allons énumérer les deux algorithmes avec le nombre d'instances correctement classé et le nombre d'instances incorrectement cluster et le temps d'exécution pour chacun d'eux.

	Nombre de d'instances correctement cluster	Nombre de d'instances incorrectement cluster	Temps d'exécution (en seconds)
K-Means	4065	1408	0.30
Hierarchical clustering	4914	559	372.02

Tableau V.10: Evaluation des performances de clustering pour le corpus pageblocks

Le tableau qui suit illustre les pourcentages d'instances correctement classifiés pour les différents algorithmes pour le corpus morphology

	Nombre de d'instances correctement classifiés	Nombre de d'instances incorrectement classifiés	Temps d'excution (s)
K-Means	74.27	25.73	0.30
Hierarchical clustering	89.79	10.21	372.02

Tableau V.11 : Evaluation des performances de clustering pour le corpus pageblocks

Le tableau suivant résume Evaluation des performances de clustering pour les trois corpus.

corpus		Glass			Morphologie			Page blocks		
Nbr d'instances		214			2000			5473		
		Instance correctement classifié	Instance incorrectement classifié	Temps (s)	Instance correctement classifié	Instance incorrectement classifié	Temps (s)	Instance correctement classifié	Instance incorrectement classifié	Temps (s)
K-Means	Nbr	95	119	0.3	398	1602	0.5	4065	1408	0.28
	%	**44.39**	55.61		**19.9**	80.1		**74.27**	25.73	
Hirarchical clustering	Nbr	78	136	0.72	392	1608	17.84	4914	559.0	**372.02**
	%	36.43	63.57		19.6	80.4		**89.79**	10.21	

Tableau V.12 : Evaluation des performances de clustering pour les trois corpus

c. Expérimentation et résultats

K-means est un algorithme de clustering il est largement utilisé pour la mise en cluster de grands ensembles de données.

Nous avons effectué une analyse sur les deux méthodes : k-means et l'algorithme hiérarchique. Les résultats expérimentaux montrent que lorsque la taille des données est petite l'algorithme kmean est plus performant que l'algorithme hiérarchique (c'est le cas pour les deux corpus Glasse et morphologie) et prend moins de temps à exécuter. Mais dans le cas des grandes corpus (cas de pageblock avec 5473 instances), l'algorithme hiérarchique est plus performant que kmeans sauf qu'il prend énormément de temps dans l'exécution.

Après avoir analysé les deux algorithmes nous avons conclu que :

À mesure que le nombre d'instances augmente, les performances de l'algorithme hiérarchique croient et le temps exécution devient accru.

L'algorithme K-means augmente également son temps d'exécution, mais par rapport à l'algorithme hiérarchique, ses performances est mieux.

CONCLUSION

L'analyse comparative de divers algorithmes de classification (K-means, Hierarchical clustering) a été faite. Les résultats ont été validés à l'aide des trois corpus (glass, morphologie et pagesblocks) provenant des référentiels UCI.

Approprié les méthodes peuvent être utilisées en fonction de leur utilisation.

Ce travail nous a amené au développement d'une étude comparative entre les méthodes de classification supervisé utilisant les trois algorithmes (RN, SVM, arbre de décision) et une autre étude pour les méthodes de classification non supervisé en utilisant les deux algorithmes (K-Means, hiérarchique) appliqués aux trois bases de données sélectionnées (Glass, Mfeat_morphology, page-blocks).

L'objectif de notre travail est de donner la possibilité aux professionnels d'étudier et d'appliquer les différentes méthodes d'apprentissage supervisé et non supervisé à travers l'outil de classification Weka et d'analyser les résultats afin d'évaluer les performances de chaque méthode.

Pour ce faire, nous avons tout d'abord étudié les approches et les notions fondamentales de la classification des données. En premier lieu, nous avons présenté les différentes techniques intervenant dans la classification. Puis, nous avons distingué les différentes méthodes d'apprentissage supervisé et non supervisé. Cette analyse nous a permis de constater l'importance de chaque méthode de classification, pour ensuite aborder les différents outils de classification, nous nous intéressons particulièrement à exposer l'outil utilisé : «WEKA». Enfin, nous présentons les limites actuelles de notre étude comparative entre les résultats et les difficultés rencontrées car on a conclue qu'il est difficile d'affirmer qu'une telle méthode est meilleure par rapport à une autre. Globalement, cette étude a permis d'exposer concrètement la problématique de classification de données dans des différents domaines.

Notre travail est une nouvelle contribution apportée aux nombreuses études sur la classification. Pour une évaluation plus complète, et d'après ce qu'on a pu remarquer à travers ce travail on pourrait envisager quelques perspectives afin de noter ce qu'on peut améliorer dans celui-ci, il serait judicieux dans le futur d'élargir le volume et le type des données. Comme, il serait pertinent de tester d'autres types de classifieurs et d'utiliser d'autres logiciels comme TANAGRA ou MATLAB pour voir le comportement de ces algorithmes.

[1] Mooers, Calvin.N, ‟Application of Random Code to the gathering of statistical information‟, thèse de maitre, Massachusetts Institute of technology 1948.

[2] Page Lawrence, Sergey brin, Rajeev Motwani et Terry Winograd, “The Page rank Citation ranking: Bringing Order to the Web”, rapport technique, Stanfort Digital Library Technologies project, 1998.

[3] MATALLAH, Hocine. *Classification Automatique de Textes Approche Orientée Agent*. Diss. 2011.

[4] Sebastiani Fabrizio. «*Machine Learning in A utomated Text Categorization*». ACM Computing Surveys, 34(1): 1–47, (2002)

[5] Yang Y, Slattery S, Ghani R. «*A Study of Approaches to Hypertext Categorization*». Journal of Intelligent Information Systems JIIS, 18(2-3) : 219–241, (2002)

[6] Denoyer L. « *Apprentissage et inférence statistique dans les bases de documents structurés : Application aux corpus de documents textuels* ». Thèse de Doctorat, Université Paris 6, (2004)

[7] I. Dhillon, J. Fan, and Y. Guan. E_cient clustering of very large document collections. In G. Kamath R. Grossman and R. Naburu, editors, Data Mining for Scienti_c and Engineering Applications. Kluwer Academic Publishers, 2001

[8] M. Steinbach, G. Karypis, and V. Kumar. A comparison of document clustering techniques, 2000.

[9] M. Ester, A. Frommelt, H. Kriegel, and J. Sander. Spatial data mining: Database primitives, algorithms and e_cient dbms support, 2000.

[10] X. Xu, M. Ester, H. P. Kriegel, and J. Sander. A distribution-based clustering algorithm for mining in large spatial databases. In ICDE '98: Proceedings of the Fourteenth International Conference on Data Engineering, pages 324{331, Washington, DC, USA, 1998. IEEE Computer Society.

[11] R. Cooley, B. Mobasher, and J. Srivastava. Data preparation for mining world wide web browsing patterns. Knowledge and Information Systems, 1(1) :5{32, 1999.

[12] A. Foss, W. Wang, and O. Zaane. A non-parametric approach to web log analysis, 2001.

[13] A. Ben-Dor, R. Shamir, and Z. Yakhini. Clustering gene expression patterns. Journal of Computational Biology, 6(3/4) :281{297, 1999.

[14] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society, Series B, 39(1) :1{38, 1977.

[15] V. Ganti, J. Gehrke, and R. Ramakrishnan. CACTUS- clustering categorical data using summaries. In KDD '99 : Proceedings of the _fth ACM SIGKDD international conference on Knowledge discovery and data mining, pages 73{83, New York, NY, USA, 1999. ACM Press.

[16] Denoyer L. «*Apprentissage et inférence statistique dans les bases de documents structurés : Application aux corpus de documents textuels*». Thèse de Doctorat, Université Paris 6, (2004)

[17] Jain A.K, Dubes R.C. *Algorithms for Clustering Data*. Prentice-Hall advanced reference series: Computer Science, Prentice-Hall, Inc, Upper Saddle River, NJ, New Jersey, (1988)

[18]Mariam TANANA « Evaluation formative du savoir-faire des apprenants à l'aide d'algorithmes de classification », thèse Doctorat de l'INSA de Rouen (spécialité Informatique) ,2009.

[19] Mounzer BOUBOU : "contribution aux méthodes de classification non supervisée via des approches prétopologiques et d'agrégations d'opinion'', thèse de doctorat, université Claud Bernard –Lyon1, 2007

[20] Aggarwal G., Feder T., Kenthapadi K., Motwani R., Panigrahy R., Thomas D., Zhu A.:Approximation Algorithms for *k*-anonymity. *Journal of Privacy Technology*, paper 20051120001, 2005.

[21]. D. Bensalem, C. Bounouar, Z. Boudia, Classification automatique de documents: de la classification classique à la classification utilisant une ressource externe, Département Informatique de l'UMMTO, 2014

[22] *Aggarwal C. C., Yu P. S.: On Variable Constraints in Privacy-Preserving Data Mining. SIAM Conference, 2005.*

[23] Aggarwal C. C.: On Randomization, Public Information and the Curse of Dimensionality.

ICDE Conference, 2007.

[24] A-G. Bosser. Réplications Distribuées pour la Définition des Interactions de Jeux Massivement Multi-Joueurs. PhD thesis, Université de Paris 7, France, Novembre 2005.

[25].Les arbres de décision (decisiontrees), Christine Decaestecker, ULB, Marco Saerens, UCL,LINF2275

[26] I. Foster and C. Kesselmann. The Grid : Blueprint for Future Computing Infrastructure. Morgan Kaufmann, San Francisco, 1999.

[27] Samarati P.: Protecting Respondents' Identities in Microdata Release. IEEE Trans. Knowl. Data Eng. 13(6): 1010-1027 (2001).

[28] Bettini C., Wang X. S., Jajodia S.: Protecting Privacy against Location Based Personal Identification. *Proc. of Secure Data Management Workshop*, Trondheim, Norway, 2005.

[30] A. Imine. Conception Formelle d'Algorithmes de Réplication Optimiste Vers l'Edition Collaborative dans les réseaux Pair-a-Pair. PhD thesis, Doctorat de l'université Henri Poincaré Nancy 1, France, Novembre 2006.

[31] http://www.cs.utk.edu/netSolve

[32] B.Liu. Web Data Mining, Exploring Hyperlinks, Contents and Usage Data. Springer, Berlin, 2011.

[33] P.Jaskowiak. On the Evaluation of Clustering Results: Measures, Ensembles and Gene Expression Data Analysis. Thése de doctorat en sciences en informatique, Université de San Carlos, Cebu, Philippine, 2016.

[34] G.Forestier. Connaissances et Clustering Collaboratif d'Objets Complexes Multisources. Thése de doctorat en sciences en informatique, Université de Strasbourg, France, 2010.

[35] Boubou, Mounzer. *Contribution aux méthodes de classification non supervisée via des approches prétopologiques et d'agrégation d'opinions*. Diss. 2007.

[36] SOUHILA DJERROUD : Accélération de la recherche d'image par le contenu, Intégration de la méthode de CHAMELEON, thèse pour obtenir le grade de magister, 2007.

[37]Park, Hae-Sang et Chi-Hyuck Jun. "Un algorithme simple et rapide pour la mise en grappes de K-medoids." *Systèmes experts avec applications* 36.2 (2009): 3336-3341.

[38] Forgy, E. (1965). Cluster analysis of multivariate data: efficiency vs interpretability of classifications. . *Biometrics, 21*, 768-769.

[39] B. DEVEZE & M. FOUQUIN, Data mining c4.5 -dbscan, Cours, Ecole d'ing_enieurs en informatique EPITA, France, 2004.

[40] A. Imine. Conception Formelle d'Algorithmes de Réplication Optimiste Vers l'Edition Collaborative dans les réseaux Pair-a-Pair. PhD thesis, Doctorat de l'université Henri Poincaré Nancy 1, France, Novembre 2006

[41] A.Rosenberg and J.Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, page 410–420, Prague, 2007.

[42] M.Koskela, J.Laaksonen, and E.Oja. Entropy-based measures for clustering and som topology preservation applied to content-based image indexing and retrieval. In Proceedings of the 17th International Conference on Pattern Recognition, pages 1005 – 1008, Cambridge, Rayaume Uni, 2004.

[43] www.cs.waikato.ac.nz/ml/weka/

[44] www.ics.uci.edu