Table des matières

Liste des tableaux

Table des figures

INTRODUCTION GÉNÉRALE

Dans la classification multi-label, chaque instance peut appartenir à une ou plusieurs classes simultanément. Très souvent dans le cas d'application réelle, les classes ne sont pas indépendantes les unes des autres.

La classification multi-labels hiérarchique traite des problèmes où les classes sont organisées sous forme d'une hiérarchie. Ce domaine de recherche n'a pas eu une grande attention, et pourtant on retrouve ce problème dans beaucoup d'applications importantes (Les bibliothèques numériques tel que Wikipédia, la reconnaissance d'images. . .).

Cette organisation hiérarchique est essentielle parce qu'elle permet à l'utilisateur de se concentrer sur le niveau approprié de détails. A partir de là, nous pouvons légitimement penser à l'analyse de concepts formels et aux treillis de Galois. En effet, en analyse de concepts formels, les connaissances induites (appelées concepts formels) à partir d'un contexte formel sont hiérarchisées et représentées sous la forme d'un treillis de Galois.

Notre travail consiste à :

• Appliquer l'analyse de concept formel dans la classification multi-label hiérarchique,

• Calculer le degré de corrélation entre différentes classes du fichier arff pour voir la structuration de donnée (Dataset), et ce en proposant une mesure de similarité et puis la modifier afin de calculer ce degré de corrélation.

Ce mémoire contient quatre (04) chapitres qui se répartissent comme suit :

Premier chapitre : Nous avons donné un aperçu général sur l'apprentissage automatique en se basant sur la classification supervisée.

Deuxième chapitre : En premier temps nous avons parlé de la classification multi-label et puis de la classification multi-label hiérarchique.

Troisième chapitre : Nous avons introduit l'analyse de concepts formels et les treillis de galois.

Quatrième chapitre : Nous l'avons divisé en deux partie majeures ; Classification multi-label hiérarchique et treillis de Galois, la mesure de similarité

Chapitre 1

GÉNÉRALITÉS SUR LA CLASSIFICATION

1.1 Introduction

La classification consiste à regrouper les données dans des groupes d'échantillions similaires en se basant sur différents algorithmes de classification. Ces algorithmes sont des ensembles d'instructions explicitement programmés.

Dans ce chapitre nous avons donnée un aperçu général sur différentes notions de base qui vont etre utilisées durant tout ce rapport.

1.2 Généralités

1.2.1 Apprentissage automatique

L'apprentissage automatique fait référence au processus par lequel les ordinateurs développent la reconnaissance de schémas ou l'aptitude à apprendre continuellement et à faire des prévisions basées sur des données, puis à faire des rectifications sans avoir été spécifiquement programmés pour le faire. L'apprentissage automatique est une forme d'intelligence artificielle qui automatise efficacement le processus de création de modèles analytiques et qui permet aux machines de s'adapter à de nouveaux scénarios de manière indépendante.[**1**]

1.2.2 Classification

La classification automatique peut être définie comme la division des données en groupes ou sous-ensembles d'objets similaires. En effet, c'est la catégorisation algorithmique d'objets et elle consiste à attribuer une classe ou catégorie à chaque objet à classer, en se basant sur des données statistiques. Elle fait couramment appel à l'apprentissage automatique dans un cadre d'apprentissage non supervisé qui a pour but d'obtenir des informations sans aucune connaissance préalable, au contraire de l'apprentissage supervisé. [**2**] [**3**]

1.2.3 Intelligence artificielle

Le terme « intelligence artificielle », créé par John McCarthy, est souvent abrégé par le sigle « IA » (ou « AI » en anglais, pour Artificial Intelligence). Il est défini par l'un de ses créateurs, Marvin Lee Minsky, comme « la construction de programmes informatiques qui s'adonnent à des tâches qui sont, pour l'instant, accomplies de façon plus satisfaisante par des êtres humains car elles demandent des processus mentaux de haut niveau tels que : l'apprentissage perceptuel, l'organisation de la mémoire et le raisonnement critique ».

1.2.4 Classifieur

D'après Tom M. Mitchell : «Un programme informatique apprend de l'expérience E par rapport à une classe de tâches T et mesure de performance P si sa performance à des tâches

en T, mesurée par P, s'améliore avec l'expérience E.≫

En apprentissage automatique, le terme de classifieur représente une famille d'algorithmes de classement statistique. Son rôle est de classer dans des classes les échantillons qui ont des propriétés similaires, mesurées sur des observations. Un classifieur linéaire est un type particulier de classifieur, qui calcule la décision par combinaison linéaire des échantillons.

En effet, Les algorithmes de classement (Classifieurs) ont pour objet d'identifier la classe d'appartenance d'objets définis par leur description.

1.2.5 Objets

Un objet noté O est un ensemble de propriétés ou d'attributs $(a_1, ..., a_n)$ qui peut appartenir à une ou plusieurs classes X. $(O \epsilon X\ )$

1.2.6 Apprentissage non supervisé

La classification non supervisé est un problème d'apprentissage automatique, entre autres, les données sont non étiquetées, de sorte que l'algorithme d'apprentissage trouve tout seul des points communs parmi ses données d'entrée. Puisque les données ne sont pas étiquetées, il n'est pas possible d'affecter au résultat de l'algorithme utilisé un score d'adéquation. Cette absence d'étiquetage est ce qui distingue les tâches d'apprentissage non-supervisé des tâches d'apprentissage supervisé[4] . Son objectif est d'organiser (regrouper) les données sous forme des clusters. [5]

1.2.7 Apprentissage Supervisé

L'apprentissage automatique intervient dans des processus de décision et doit permettre de répondre à des questions aussi diverses que : [6]

Quel sera le résultat du prochain match Jsk - Mca ?

La molécule que je désire commercialiser est-elle cancérigène ?

Quel est l'auteur de cette page HTML ? Cette phrase est-elle grammaticalement correcte ?

Les experts humains peuvent consulter sur un bon nombre de ces questions (un médecin pour déterminer le risque encouru par un patient, un amateur de football pour le résultat

du match, etc.). Cependant, il s'agit ici d'automatiser cette prise de décision et donc d'en exclure toute intervention humaine.

Cette motivation a conduit à la définition de systèmes experts (ou systèmes à base de connaissance) : ceux-ci sont capables de mener un raisonnement à partir de faits décrivant le problème à résoudre et d'une expertise sous forme de règles. La difficulté est alors d'obtenir ces règles et, à nouveau, il est naturel de se tourner vers les experts humains. Malheureusement, les experts humains sont le plus souvent incapables d'expliciter leurs raisonnements, en particulier sous une forme contrainte de règles.

Notons cependant que pour certains problèmes, il n'existe pas d'expertise humaine (par exemple pour décider d'emblée si une nouvelle molécule est cancérigène ou non). Dans de tels cas, les exemples étiquetés nous seront fournis par les expériences passées (par exemple en utilisant les molécules qui ont été déterminées cancérigènes ou non par de longs tests en laboratoire sur des rats). Il ne s'agit plus ici d'expliciter une expertise humaine mais bien de découverte scientifique.

C'est l'objectif de l'apprentissage automatique : produire automatiquement des règles.

Plusieurs types d'apprentissage supervisé existe, on cite : ***la régression, la détection d'anomalies et la classification.*** [5]

1.2.8 La classification binaire

La classification binaire ou binomiale est la tâche de classer les éléments d'un ensemble donné en deux groupes (prédire à quelle classe chacun appartient) sur la base d'une règle (Algorithme) de classification. [7]

Certaines des méthodes couramment utilisées pour la classification binaire sont :

Arbres de décision

Bayes

Séparateurs à Vaste Marge (SVM)

K-plus proches voisin (KNN)

Les réseaux de neurones

En effet, un large éventail d'algorithmes d'apprentissage supervisé sont disponibles, chacun avec ses forces et ses faiblesses. Il n'y a pas d'algorithme d'apprentissage unique qui fonctionne le mieux sur tous les problèmes d'apprentissage supervisé. [8]

1.2.9 Extension de la classification binaire

La classification multi-classe est une exigence majeure dans le domaine de la science et de l'ingénierie, car la discrimination multi-classe des objets est un problème sérieux en science et en ingénierie[**7**]. Cette approche qui peut s'appliquer au cas binaire, mais se généralise au cas où on a m classes ; par exemple [**8**], classer un ensemble d'images de fruits qui peuvent être des oranges, des pommes ou des poires[**9**]. La classification multi-classe fait l'hypothèse que chaque échantillon est affecté à une et une seule étiquette : un fruit peut être soit une orange ou une pomme ou une poire mais pas les trois (03) en même temps. [**10**]

1.3 Quelques Méthodes de la classification supervisée

Les méthodes de classification ont pour but d'identifier les classes auxquelles appartiennent des objets à partir de certains paramètres descriptifs. Elles s'appliquent à un grand nombre d'activités humaines et conviennent en particulier au problème de la prise de décision automatisée.

Ces méthodes sont continuellement développées. Pour nos besoins, nous allons présenter quelques méthodes populaires utilisées dans l'apprentissage supervisé, nous allons nous intéresser aux algorithmes suivants : Séparateur à Vaste Marge (***SVM pour Support Vector Machuine en anglais), Arbre de décision, K-plus proches voisins (KNN pour Near-set Neighbor en anglais et Bayes***). [4]

1.3.1 Séparateurs à Vaste Marge (Support Vector Machine, SVM)

Séparateur à vaste marge 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 garan-

tissant 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. [**11**]

1.3.2 Arbre de décision

Un arbre de décision est un ensemble de règles de classification basant leur décision sur des tests associés aux attributs, organisés de manière arborescente.

Les méthodes d'arbre de décision sont essentiellement développé dans le cadre des approches d'apprentissage automatique (machine learning) en Intelligence Artificielle. [**12**] Son but est de créer un modèle qui prédit la valeur d'une variable-cible depuis la valeur de plusieurs variables d'entrée.

Tout arbre de décision définit un classifieur qui se traduit immédiatement en termes de règles de décision. [**13**]

1.3.3 K-plus proches voisins (KNN pour K Nearest Neighbor en anglais)

L'algorithme KNN parmi les plus simples algorithmes d'apprentissage artificiel. Dans un contexte de classification d'une nouvelle observation x, l'idée fondatrice est de choisir pour chaque sommet la classe majoritaire parmi ses k plus proches voisins de l'observation x. [**14**]

La méthode KNN est donc une méthode à base de voisinage, non-paramétrique ; Ceci signifiant que l'algorithme permet de faire une classification sans faire d'hypothèse sur la fonction $y = f(x_1, x_2, x_3, \ldots, x_p)$ qui relie la variable dépendante aux variables indépendantes. [**15**]

L'algorithme Knn a l'objectif de classer des exemples non étiquetés sur la base de leur similarité avec les exemples de la base d'apprentissage.

1.3.4 Bayes

Avant d'entamer l'algorithme Bayésien, il est primordial de connaitre certaines notions en probabilité. On commence par un rappel de probabilité, soient h et D deux évènements :

— La probabilité de h est $P(h)$.

— La probabilité de$D$ est $P(D)$.

— La probabilité d'un évènement est comprise entre 0 et 1 $(0 < P(h)1\, et\, 0 < P(D) < 1)$

— La probabilité d'un évènement certain vaut 1.

— La probabilité d'un évènement impossible vaut 0.

— Si h et Dsont indépendants : $P(h \bigcup D) = P(h) + P(D)$

— $P(nonh) = 1 - P(h)$

— $P(h/D)$ est la probabilité que h survienne si D survient

— $P(h/D) = P(h \bigcap D)/P(D)$

— $P(h \bigcap D) = P(h/D) \times P(D) = P(D/h) \times P(h)$

Alors pour le théorème de Bayes nous avons un espace d'hypothèses H, et un ensemble de données D. Nous définissons les 3 probabilités suivantes **[16]** :

1. $P(h)$ la probabilité que h soit l'hypothèse correcte sans avoir vu aucune donnée. $P(h)$ est dit "prior probability" de h.

Exemple : Chance qu'elle pleuve est 80% si on est prêt de la mer et à la latitude X (aucune donnée n'est vu).

2. $P(D)$ la probabilité de voir les données de D.

3.$P(D|h)$ est la probabilité des données sachant h.

Le théorème de Bayes relie la probabilité à postériori d'une hypothèse h sachant des données, avec les trois probabilités mentionnées dans la définition précédente :

$P(h|D) = P(h \bigcap D)/P(D)$

$P(h \bigcap D) = P(D|h).P(h)/P(D)$

Alors ; $P(h|D) = \frac{P(D|h).p(h)}{P(D)}$

tel que ; $P(h|D)$: Posterior Prbability, $P(D|h)$: Likelihood, , $p(h)$: Prior Probability, $P(D)$: Evidence

1.4 Conclusion

Les systèmes de classification impliquent les utilisateurs dans la génération et l'évolution de leurs propres classifieurs de données. La plupart de ces systèmes se limitent à une classification Mono-Label où une seule étiquette de classe est affectée à chaque instance. D'autres systèmes se limitent à une classification Binaire qui permet de classer les données en deux (02) classes, un objet peut appartenir à l'une des deux (02) classes existantes($O\epsilon(C_1 \, OR \, C_2)$. Et d'autres systèmes se limitent à une classification Multi-Classes qui fait l'hypothèse que chaque échantillon est affecté à une et une seule étiquette, un objet peut appartenir à une seule classe parmi plusieurs, $O\epsilon(C_1 \, OR \, C_2 \, OR \, C_3 \, OR \, ... \, OR \, C_n)$.

Ces classifications sont d'une certaine façon, à un moment donnée, limitées par le progrès des technologies et par l'apprentissage automatique qui est un domaine en développement continu. Elles ne peuvent pas résoudre tous les problèmes rencontrés dans la science et/ou dans l'apprentissage automatique. Par exemple ; une photo prise à la plage peut contenir à la fois, des personnes, du sable, de l'eau (la mer), des animaux etc. . ., cet exemple ne peux pas être placé dans les classifications précédemment citées. En effet, où est ce qu'on peut classer cet exemple ? Or, on se pose la question : est-ce-que un objet peut appartenir à plusieurs classes en même temps ?

($O\epsilon(C_1 \, AND \, C_2 \, AND \, C_3 \, AND \, ... \, AND \, C_n)$?

Pour répondre à cette question et remédier aux problèmes posés, une nouvelle classification est apparue ; c'est la classification Multi-label qui sera le thème de notre prochaine partie.

Chapitre 2

CLASSIFICATION MULTI-LABEL

2.1 Introduction

La classification traditionnelle en une seule étiquette concerne l'apprentissage à partir d'un ensemble d'exemples associés à une seule étiquette l provenant d'un ensemble d'étiquettes disjointes L ,$|L| > 1$. Si $|L| = 2$, alors le problème d'apprentissage est appelé un problème de classification binaire, alors que si $|L| > 2$, il est appelé un problème de classification multi-classe. Dans la classification multi-étiquettes, les exemples sont associés à un ensemble d'étiquettes.

Dans le passé, la classification multi-étiquettes était principalement motivée par les tâches de catégorisation des textes et de diagnostic médical. Par exemple dans la catégorisation des textes, un document d'information pourrait couvrir plusieurs sujets tels que les sports, la vente de billets, histoire et la culture ; De même dans le diagnostic médical, un patient peut souffrir par exemple du diabète et du la tension en même temps.

De nos jours, les méthodes de classification multi-étiquettes sont de plus en plus demandées par les applications modernes, telles que la classification des fonctions protéiques, la classification musicale et la classification sémantique. Dans la classification de scène sémantique, une photographie peut appartenir à plus d'une classe conceptuelle, comme les couchers de soleil et les plages en même temps. De même, dans la catégorisation musicale, une chanson peut appartenir à plus d'un genre.

Dans cette partie nous allons parler des trois (03) grandes familles d'apprentissage Multi-Label qui sont : Approche d'apprentissage par transformation, approche par adaptation et approche d'ensemble. Ainsi nous allons voir les différents critères d'évaluation Multi-Label et nous terminons notre chapitre par la classification Multi-Label-Hiérarchique.

2.2 Approches d'apprentissage multi-label

Les approches d'apprentissage multi-label peuvent être organisées en trois grandes familles [**17**] [**18**] [**19**].

- **Approches d'apprentissage par transformation :** elles transforment le problème d'apprentissage multi-label en un ou plusieurs problèmes de classification ou de régression mono-label.
- **Approches d'apprentissage par adaptation :** elles adaptent des algorithmes d'apprentissage pour des données multi-label.
- **Approches d'apprentissage d'ensemble :** elles utilisent un ensemble de classifieurs issus de la première et de la deuxième famille d'approches (Approche d'apprentissage par transformation et Approche d'apprentissage par adaptation).

Dans ce qui suit, nous allons utiliser les notations suivantes :

(n) : Nombre d'exemples d'apprentissage.

(m) : Nombre d'attributs.

(q) : Nombre de labels.

x_i : Exemples de test.

λ_i : Différents label.

y_i^+ :Labels positifs.

y_i^- : Labels négatifs.

y_i : L'ensemble des vraies classes.

$\hat{y}_5$: L'ensemble des classes prédites.

Remarque :

Les approches d'apprentissage par transformation utilisent toutes un classifieur binaire B ou multi-classes M pour l'apprentissage du ou des modèles (algorithme de construction d'un arbre de décision C4.5 ou un SVM (Support Vector Machine)).

2.2.1 Approches d'apprentissage par transformation :

Nous définissons d'abord dans l'exemple ci-dessous (tableau 2.1) un petit jeu de données multi-label pour illustrer les processus d'apprentissage et de prédiction de chaque méthode d'apprentissage par transformation. Nous lui appliquons ensuite toutes les méthodes (présentées ci-dessous) dans les tableaux : 2.2, 2.3, 2.4 et 2.5 . [17]

Exemple :

Soit un ensemble d'apprentissage multi-label de 4 exemples où chaque exemple x_i est décrit par m attributs et est annoté positivement ou négativement par 4 labels.

Par exemple, x_4 possède 3 labels positifs $y_4^+=\{\lambda_2, \lambda_3, \lambda_4\}$ et un label négatif $y_4^-=\{\lambda_1\}$

À partir de ces exemples d'apprentissage, un classifieur multi-label apprend un modèle de classification h et tente ensuite de prédire l'ensemble des labels les plus appropriés $\hat{y}_5 = h(x_i)$ pour un nouvel exemple x_5.

x_i	f_1	...	f_m	λ_1	λ_2	λ_3	λ_4
x_1				1	0	0	1
x_2				0	0	1	1
x_3				1	0	0	0
x_4				0	1	1	1
x_5				?	?	?	?

TABLE 2.1 – Exemple de jeu de donnée Multi-Label

2.2.1.1 *Binary Relevance (BR)*

Est la méthode la plus populaire et la plus simple de cette classe d'approches. Elle transforme le problème d'apprentissage multi-label en q problèmes de classification mono-label.

Pour l'apprentissage de chaque label $\lambda_i (1 \leq i \leq q)$un classifieur binaire h_{Bi} est utilisé (Tableau 2.2). **[20]**

Pour un nouvel exemple de test (x_5), BR retourne l'union des prédictions de chaque classifieur.

x_i	f_1	...	f_m	λ_1
x_1				1
x_2				0
x_3				1
x_4				0
x_5				?

x_i	f_1	...	f_m	λ_2
x_1				0
x_2				0
x_3				0
x_4				1
x_5				?

x_i	f_1	...	f_m	λ_3
x_1				0
x_2				1
x_3				0
x_4				1
x_5				?

x_i	f_1	...	f_m	λ_4
x_1				1
x_2				1
x_3				0
x_4				1
x_5				?

TABLE 2.2 – Transformation du jeu de données suivant l'approche d'apprentissage

La méthode Binary Relevance (BR) a l'avantage d'être simple et peu couteuse en terme de temps de calcul mais son problème majeur réside dans le fait qu'elle ne tient pas compte des corrélations éventuelles entre les classes puisque cette méthode transforme le problème de classification multi-label en q problèmes de classification mono-labels indépendants **[21]** **[22]**

2.2.1.2 *Classifier Chain (CC)*

Est une amélioration de la méthode BR qui transforme également le problème d'apprentissage multi-label en q problèmes de classification mono-label. Cependant, les classifieurs sont entraines dans un ordre aléatoire défini avant la phase d'apprentissage $[1.., i, ..q]$ tel que chaque classifieur binaire h_{Bi} apprenant un label λ_i et ajoute tous les labels associés aux classifieurs qui le précédent dans la chaine $(\lambda_1, ..., \lambda_{i-1})$ dans son espace d'attributs (Tableau 2.3 où l'ordre des labels est$[\lambda_1, \lambda_2\lambda_3\lambda_4]$). **[23]**

Pour un nouvel exemple (x_i), CC retourne l'ensemble des prédictions générées par l'ensemble des classifieurs.

x_i	f_1	...	f_m	$f_{m+1} = \lambda_1$
x_1				1
x_2				0
x_3				1
x_4				0
x_5				$\hat{y}_5^1$

x_i	f_1	...	f_m	$f_{m+1} = \lambda_1$	$f_{m+2} = \lambda_2$
x_1				1	0
x_2				0	0
x_3				1	0
x_4				0	1
x_5				$\hat{y}_5^1$	$\hat{y}_5^2$

x_i	f_1	...	f_m	$f_{m+1} = \lambda_1$	$f_{m+2} = \lambda_2$	$f_{m+3} = \lambda_3$
x_1				1	0	0
x_2				0	0	1
x_3				1	0	0
x_4				0	1	1
x_5				$\hat{y}_5^1$	$\hat{y}_5^2$	$\hat{y}_5^3$

x_i	f_1	...	f_m	$f_{m+1} = \lambda_1$	$f_{m+2} = \lambda_2$	$f_{m+3} = \lambda_3$	$f_{m+4} = \lambda_4$
x_1				1	0	0	1
x_2				0	0	1	1
x_3				1	0	0	0
x_4				0	1	1	1
x_5				$\hat{y}_5^1$	$\hat{y}_5^2$	$\hat{y}_5^3$	$\hat{y}_5^4$

TABLE 2.3 – Transformation du jeu de données suivant l'approche d'apprentissage CC

Son avantage est sa vitesse d'apprentissage du modèle et sa modélisation des corrélations entre les labels mais sa définition aléatoire de l'ordre d'apprentissage des modèles reste une faiblesse.

2.2.1.3 *Label Powerset (LP)*

Transforme le problème d'apprentissage multi-label en un seul problème d'apprentissage mono-label à plusieurs classes. Elle considère chaque combinaison de labels présente dans l'ensemble d'apprentissage comme une classe et apprend ensuite un classifieur multi-classes h_M (Tableau 2.4). **[24]**

Pour un nouvel exemple (x_i), le classifieur retourne la classe (combinaison de labels) la plus probable.

x_i	f_1	...	f_m	λ_i
x_1				$\lambda_{1,4}$
x_2				$\lambda_{3,4}$
x_3				λ_1
x_4				$\lambda_{2,3,4}$
x_5				?

TABLE 2.4 – Transformation de jeu de données suivant l'approche d'apprentissage LP

L'avantage principal de LP est son exploitation des corrélations entre labels. Néanmoins, quelques classes peuvent être difficiles à apprendre si le nombre de labels est important et le nombre d'exemples est faible. Le nombre de classes est au plus égal à $min(2^q, n)$.

Son inconvénient est qu'elle ne permet pas de prédire de nouvelles classes (combinaisons de labels) qui n'existent pas dans l'ensemble d'apprentissage.

2.2.1.4 *Calibrated Label Ranking (CLR)*

Transforme le problème d'apprentissage multi-label en$\frac{q.(q+1)}{2}$ problèmes de classification mono-label. Pour l'apprentissage de chaque paire de labels(λ_i, λ_j), $(i, j \in \{1, 2, ...q\}$ et $i \neq j)$, l'ensemble d'apprentissage est construit de telle sorte que les exemples positifs appartiennent au premier label λ_i et les exemples négatifs appartiennent au deuxième label λ_j (Tableau 2.5). Ainsi, le nombre d'exemples associés à chaque classifieur est réduit à un nombre plus petit que n. **[25]**

Pour séparer les labels positifs des labels négatifs, CLR introduit un label virtuel λ_0 et le combine avec chaque label $\lambda_i (1 \leq i \leq q)$ et pour apprendre ces q nouvelles paires de labels, l'approche BR est utilisée.

Pour un nouvel exemple (x_5), les prédictions de tous les modèles sont assemblées et un ranking est associé à chaque label.

x_i	f_1	...	f_m	λ_1-λ_2
x_1				1
x_3				1
x_4				0
x_5				?

x_i	f_1	...	f_m	λ_1-λ_3
x_1				1
x_2				0
x_3				1
x_4				0
x_5				?

x_i	f_1	...	f_m	λ_1-λ_4
x_2				0
x_3				1
x_4				0
x_5				?

x_i	f_1	...	f_m	λ_2-λ_3
x_2				0
x_5				?

x_i	f_1	...	f_m	λ_2-λ_4
x_1				0
x_2				0
x_5				?

x_i	f_1	...	f_m	λ_3-λ_4
x_2				0
x_5				?

x_i	f_1	...	f_m	λ_1-λ_0
x_1				1
x_2				0
x_3				1
x_4				0
x_5				?

x_i	f_1	...	f_m	λ_2-λ_0
x_1				0
x_2				0
x_3				0
x_4				1
x_5				?

x_i	f_1	...	f_m	λ_3-λ_0
x_1				0
x_2				1
x_3				0
x_4				1
x_5				?

x_i	f_1	...	f_m	λ_4-λ_0
x_1				1
x_2				1
x_3				0
x_4				1
x_5				?

TABLE 2.5 – Transformation de jeu de données suivant l'approche d'apprentissage CLR

L'avantage de CLR est d'exploiter les corrélations entre les labels.

Les méthodes d'apprentissage par transformation sont flexibles : elles peuvent utiliser n'importe quel classifieur mono-label. Néanmoins, la qualité de leurs performances prédictives dépend principalement du choix de ce classifieur de base. [**26**]

2.2.2 Approches d'apprentissage par adaptation

Les méthodes multi-label qui adaptent, élargissent et personnalisent un algorithme d'apprentissage machine existant pour la tâche d'apprentissage multi-label sont appelées méthodes d'adaptation d'algorithme. Nous présentons ici des méthodes multi-label proposées dans la littérature qui sont basées sur les algorithmes d'apprentissage machine suivants : k-voisins

les plus proches, Arbres de décision et Machines à Vaste Marges. [**26**]

Ces méthodes élargies sont capables de gérer directement les données multi-label, entre autres, elles permettent d'adapter les méthodes de classification mono-label existantes afin de fournir des méthodes de classification multi-label.

2.2.2.1 *k-voisins les plus proches*

Les k-voisins les plus proches en multi-label (ML-kNN) sont une extension de l'algorithme des k-voisins les plus proches (kNN). La récupération des k-voisins les plus proches est la même que dans l'algorithme kNN traditionnel. [**27**]

ML-kNN est une méthode de type BR qui combine l'algorithme standard de kNN avec une inférence bayésienne. En phase d'apprentissage, ML-kNN estime les probabilités a priori et a posteriori de chaque label à partir des exemples d'apprentissage. Pour un nouvel exemple x_i, ML-kNN calcule ses k-plus proches voisins puis mesure la fréquence de chaque label dans ce voisinage. Cette fréquence est ensuite combinée avec les probabilités estimées dans la phase d'apprentissage pour déterminer son ensemble de labels en suivant le principe du maximum a posteriori (MAP). [**28**] [**29**]

2.2.2.2 *Arbres de décision Multi-Label* $C4.5$ *(ML-C4.5)*

Les arbres de décision (Décision Tree DT) sont parmi les modèles de classification les plus faciles à comprendre. Certains algorithmes DT, tels que $C4.5$ [**30**], donnent une performance qui les rend compétitifs face à d'autres approches d'apprentissage. Par exemple, la recherche menée dans [**31**] visait à classer les gènes en fonction de leur emploi, en tenant compte du fait qu'un même gène peut intervenir dans plusieurs fonctions. La solution proposée est fondée sur l'algorithme $C4.5$, modifié de manière appropriée pour traiter plusieurs étiquettes à la fois. Les deux points clés de la méthode adaptée sont les suivants : [**32**]

— Les feuilles de l'arbre contiennent des échantillons associés à un ensemble d'étiquettes, et non à une seule classe.

— La mesure d'entropie originale $Entrepie = -\sum_{i=1,N} P(c_i) log P(c_i)$ est ajustée pour prendre en compte la probabilité de non-adhésion des instances à une certaine étiquette.

$Entrepie = -\sum_{i=1,N}(P(c_i)logP(c_i)) + (1 - P(c_i)log(1 - P(c_i)))$

Où

$P(c_i)$ est la probabilité (fréquence relative) de la classe ci,

$1 - P(c_i)$ est la probabilité de ne pas appartenir à la classe ci.

2.2.2.3 *Machines à vastes marges*

De nombreuses approches ont utilisé des SVM à étiquette unique avec une approche un-contre-tous. De plus, deux mécanismes ont été présentés pour améliorer la qualité de la marge des SVM dans un environnement un-contre-tous. [**33**] [**36**]

— Le premier est la méthode de suppression de bandes (BandSVM), fonctionne au niveau de l'instance. Une fois qu'une SVM un-contre-tous a été apprise, elle enlève des exemples négatifs semblables qui se trouvent à une distance seuil (bande) de l'hyperplan de décision apprise. [**34**]

— Le deuxième est la méthode Rank-SVM, une méthode de classement, basée sur les SVM, qui est devenue une référence dans l'apprentissage multi-label. Rank-SVM définit un ensemble de Q classifieurs linéaires qui sont optimisés pour minimiser une mesure qui évalue la fraction moyenne de paires d'étiquettes. [**35**]

2.2.3 Approches d'apprentissage d'ensemble

Les méthodes d'ensemble, dont les classifieurs de base sont des apprenants multi-label, sont considérées par [**37**] comme un groupe spécial de méthodes parce qu'elles sont développées en plus de la transformation de problème et des approches d'adaptation d'algorithme. Les ensembles de transformation du problème les plus connus sont le système RAkEL de [**38**], HOMER et ensembles de chaînes de classification [**39**].

2.2.3.1 *RAndom k labEL sets (RAkEL)*

Est une amélioration de l'approche LP. Elle entraîne un ensemble de N classifieurs multi-classes h_M suivant l'approche LP tel que chaque classifieur est construit sur un sous-ensemble

de labels aléatoire de petite taille $k < q$. Pour l'apprentissage de chaque sous-ensemble de labels, seuls les exemples qui possèdent au moins l'un de ses labels sont considérés. [**38**]

Pour un nouvel exemple x_i, les prédictions des N modèles sont assembles pour obtenir son ensemble de labels.

L'avantage de la méthode $RAkEL$ est qu'elle permet de prédire de nouvelles combinaisons de labels qui n'existent pas dans l'ensemble d'apprentissage. De plus, elle permet d'exploiter naturellement les corrélations entre les labels. Néanmoins, elle n'explore pas suffisamment tous les sous-ensembles de labels pour capturer toutes les corrélations et se focalise uniquement sur l'apprentissage de quelques sous-ensembles de taille k.

2.2.3.2 *HOMER*

introduit dans [**40**], Hierarchy Of Multilabel classifiERs est un algorithme conçu pour traiter l'ensemble de données Multi-étiquettes (Multi-Label Datasets MLD) ayant un grand nombre d'étiquettes. La méthode forme un classificateur LP avec les instances disponibles, puis les sépare en plusieurs groupes reposant sur un algorithme de clustering. Chaque groupe, avec un sous-ensemble des étiquettes, produit un nouveau classificateur plus spécialisé. L'étape précédente est répétée itérativement, de sorte que chaque groupe est divisé en plusieurs sous-groupes plus petits. Le nombre d'itérations dans ce processus est un paramètre défini par l'utilisateur donné en entrée de l'algorithme. Lorsqu'un échantillon de test arrive, il traverse la hiérarchie depuis sa racine, où le classifieur le plus général fait son travail, jusqu'aux feuilles, en passant par les nœuds intermédiaires activés par la prédiction donnée par le niveau supérieur immédiat.

2.2.3.3 *ECC*

Sont des ensembles de chaînes de classification qui représentent une technique de classification multi-label d'ensemble à base de classifieurs. Cette méthode, d'après [**41**], entraîne m Classifieurs Chaînes (CC) C_1, C_2, ...,C_m. Chaque C_k, est entraîné avec un ordre aléatoire de chaîne et un sous-ensemble aléatoire de X,. Par conséquent, chaque modèle C_k, est susceptible d'être unique et capable de donner des prédictions multi-label différentes. Ces prédictions sont

additionnées par étiquette afin que chaque étiquette reçoive un certain nombre de votes. Un seuil est utilisé pour sélectionner les étiquettes les plus populaires qui forment le jeu prédit final d'étiquettes multiples. La prévision finale est obtenue en additionnant les prédictions par étiquette, puis en appliquant un seuil pour sélectionner les étiquettes pertinentes.

La force de cette approche réside dans la combinaison de différents classifieurs qui modélisent plus finement les corrélations entre les labels.

Il existe plusieurs critères pour évaluer les performance d'un classifieurs Multi-Label. En général, toutes les approches d'apprentissage multi-label optimisent implicitement le critère Hamming Loss. Néanmoins, il y a des approches qui optimisent d'autres critères. Par exemple, LP et $RAkEL$ optimisent implicitement le critère Exact match et CLR optimise explicitement le critère Ranking Loss. Ces critères sont définis dans la partie suivante.

2.3 Évaluation multi-label

Pour évaluer la performance prédictive d'un classifieur multi-label, il existe un grand nombre de critères différents dans la littérature **[42] [43].** La variété des critères d'évaluation est nécessaire pour fournir un aperçu global sur la performance prédictive de chaque classifieur. Ces critères peuvent être classés en deux familles :

2.3.1 Critères d'évaluation de la classification

Ils permettent de comparer deux vecteurs binaires en fonction d'un critère spécifique. La plupart de ces critères sont utilisés dans l'évaluation binaire mono-label et ont été généralisés au cas multi-label. Ces critères se décomposent aussi en deux catégories, nous allons nous intéresser dans cette partie au premier caractère (qui sera expliqué ci-dessous) **[26]** :

2.3.1.1 *Critères d'évaluation de la classification par exemple*

ils évaluent, pour chaque exemple test x_i, la différence entre son vecteur des vrais labels y_i et son vecteur de prédictions $\hat{y}_i$. Ils calculent ensuite la moyenne sur tous les exemples de l'ensemble de test.

Etant donné un ensemble test $T = \{(x_i, y_i), i = 1, ..., N\}$ et un classifieur multi-label H, soient $\hat{y}_i = H(x_i)$ l'ensemble des labels prédits par le classifieur multi-label pour l'exemple test x_i et y_i le vrai ensemble de labels associé à x_i.

Exemple :

On suppose un exemple comme suit :

L'ensemble des vraies classes est $y_i = \{w_1, w_2, w_3\}$

L'ensemble des classes prédites est $\hat{y}_i = \{w_2, w_3, w_4, w_5\}$

— **Coût de Hamming (Hamming Loss) :** Il permet d'évaluer, pour chaque exemple test x_i, le nombre de labels mal classés (Un label qui n'appartient pas à y_i est prédit, ou bien un label qui appartient à y_in'est pas prédit)

$$HLoss = \frac{1}{N} \sum_{i=1}^{N} \frac{y_i \triangle \hat{y}}{Q}$$

Où $\triangle$ est la différence symétrique entre ensembles des vrais classes (labels) y_i et l'ensemble des prédictions $\hat{y}_i$.

Pour l'exemple précédent, le nombre total de classes est $Q = 5$

La différence symétrique est l'ensemble des classes qui appartiennent à y_i ou à $\hat{y}_i$ mais pas à y_i et $\hat{y}_i$ au même temps. $y \triangle \hat{y}_i = \{w_1, w_4, w_5\}$.

Et donc le coût de Hamming pour une seule instance x est : $\frac{3}{5}$

— **Précision :** Cette métrique mesure le degré de similarité entre l'ensemble des classes prédites $\hat{y}_i$ et l'ensemble des vraies classes y_i (le taux de bonnes prédictions) :

$$Précision = \frac{1}{N} \sum_{i=1}^{N} \frac{|y_i \cap \hat{y}|}{|y_i \cup \hat{y}|}$$

Dans l'exemple précédent, la Précision pour une seule instance x est de $\frac{2}{5}$.

— **Rappel (Recall) :** Il permet d'évaluer, pour chaque exemple test x_i , la proportion des labels correctement prédits sur l'ensemble des vrais labels y_i

$$Rappel = \frac{1}{N} \sum_{i=1}^{N} \frac{|y_i \cap \hat{y}|}{|y_i|}$$

Dans l'exemple précédent, le rappel est de $\frac{2}{3}$.

— **Mesure F :** La mesure F correspond à une moyenne harmonique de deux autres métriques, la précision et le rappel.

$$F = \frac{1}{N} \sum_{i=1}^{N} \frac{2|y_i \cap \hat{y}|}{|y_i||\hat{y}|}$$

La mesure F pour l'exemple précédent est de $\frac{4}{12}$.

Remarques :

— Il existe d'autres critères : **Accuracy, exact match**... etc

— Tous ces critères sont définis dans l'intervalle[0..1] et leurs plus grandes valeurs indiquent les meilleures performances sauf pour le Hamming Loss où la plus petite valeur indique la meilleure performance.

2.3.1.2 *Critères d'évaluation de la classification par label*

Pour combiner les performances obtenues pour les différents labels, deux types de moyennes peuvent être utilisées selon le contexte d'application : la micro-moyenne ou la macro-moyenne.

2.3.2 Critères d'évaluation du ranking

Ils permettent d'évaluer, pour chaque exemple test x_i, la différence entre le vrai ranking y_i et le ranking prédit $\hat{y}_i$. Parmi les critères d'évaluation du ranking on cite : **Ranking Loss, One error, coverage** et **average precision.**

Ces critères sont définis dans l'intervalle [0..1] sauf Coverage qui est défini dans l'intervalle $[0..q-1]$. Leurs plus petites valeurs indiquent les meilleures performances à l'exception d'Average précision où la plus grande valeur indique la meilleure performance.

2.4 Données multi-label

Pour évaluer les performances prédictives ainsi que la vitesse de calcul des algorithmes d'apprentissage multi-label, il existe plusieurs jeux de données de caractéristiques différentes et issus de divers domaines d'application (audio, biologie, musique, texte, image). Nous citons :

— **Scene [44]** est un jeu de données où des images sont décrites par 294 attributs numériques. Elles peuvent être annotées avec 6 concepts au maximum.

— **Birds** [**45**] est un petit jeu de données où des enregistrements audio de dix (10) secondes de sons d'oiseaux sont décrits par 260 attributs numériques. Ils peuvent être étiquetés avec 19 espèces d'oiseaux au maximum.

2.5 La classification multi-label hiérarchique

La classification multi-label, dans laquelle une instance peut être associée à plusieurs classes simultanément, est rencontrée dans plusieurs domaines. Par exemple, un patient peut souffrir de plus d'une maladie à la fois, diabète, pression artérielle élevée et taux de cholestérol élevé. De même un film peut appartenir aux catégories sciences fiction, horreur et aventure. Néanmoins, la plupart des algorithmes proposés dans ce domaine ne tiennent pas compte des relation existantes entres les différentes classes et n'exploitent pas leurs structure hiérarchique.

Pour y remédier, La classification multi-label hiérarchique traite des problèmes où les classes sont organisées sous forme d'une hiérarchie. Ce domaine de recherche n'a pas eu une grande attention, et pourtant on retrouve ce problème dans beaucoup d'applications importantes telle que, entre autres, les bibliothèques numériques, Le domaine de la bio-informatique, La reconnaissance d'images.

la classification hiérarchique devient une nécessité dans de nombreux domaines. En effet, lorsque le nombre de classes est important (très grand), il est souvent utile de les organiser en groupes et sous groupes. Et donc pouvoir exploiter la structure hiérarchique des classes.

Un exemple de problème HMC structuré en arbre est illustré à la Figure qui suit, dans lequel un exemple est assigné à deux chemins de la hiérarchie, formés par les classes 11.02.03.04 et 11.06.01.

2.5.1 Définition de l'hiérarchie

Une hiérarchie $H = (N, E)$ est définie comme un graphe acyclique dirigé (DAG) constitué d'un ensemble de nœuds N et un ensemble d'arêtes E où une arête est une paire ordonnée de nœuds (N_p, N_c) tel que $(N_p, N_c) \epsilon E$ et $E \subseteq \{N \times N\}.(N_p, N_c)$ est définie à partir du nœud parent N_p vers le nœud enfant N_c, spécifié par l'opérateur de relation $N_p \implies N_c$qui a

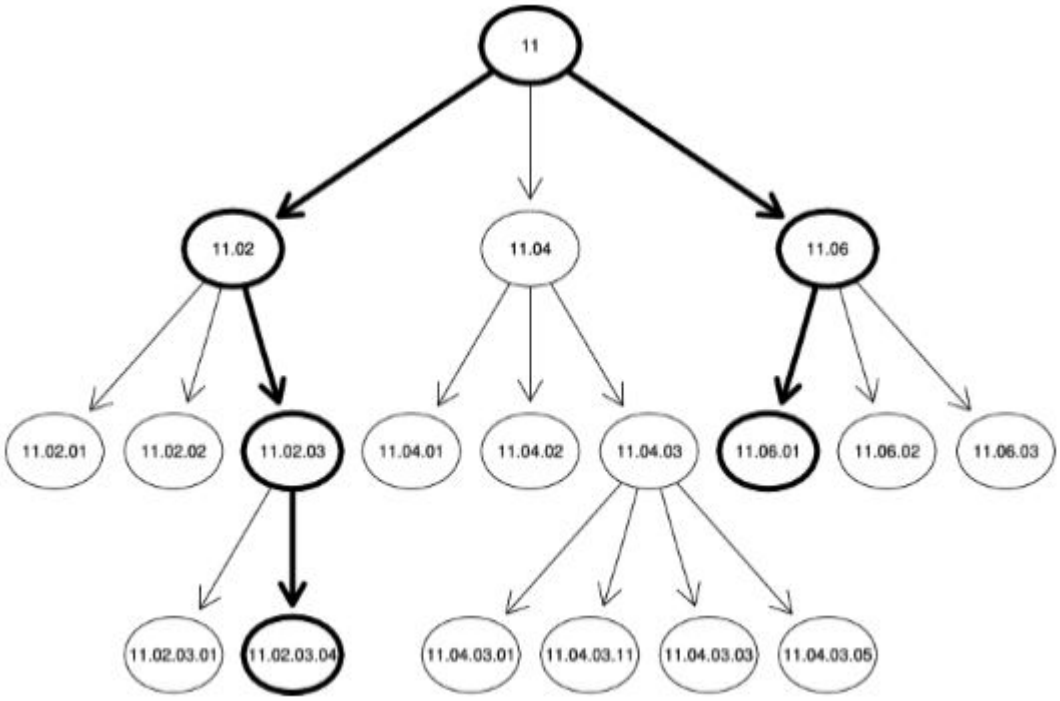


FIGURE 2.1 – Exemple d'une structure hiérarchique d'arbre.

également appelé le chemin direct de N_p à N_c. Un chemin $N_a \longrightarrow N_c$ indique qu'il existe un chemin entre le nœud ancêtre N_a et le nœud enfant N_c. Notez que dans une hiérarchie H avec un chemin $N_a \longrightarrow N_c$ il n'existe aucun chemin $N_c \longrightarrow N_a$ puisque la hiérarchie est acyclique.[**46**]

De plus, il existe exactement un nœud appelé nœud racine N_r d'un graphe H qui n'a pas de parent. Les nœuds sans nœuds enfants sont appelés nœuds feuilles. Tous les nœuds sauf le nœud racine et les nœuds feuille sont appelés nœuds internes.

2.5.2 Méthodes de classification multi-labels hiérarchique

Selon [**46**], la classification hiérarchique diffère selon un certain nombre de critères :

— Le premier critère concerne le type de la structure hiérarchique utilisée. Soit une structure arborescente ou une structure en DAG. La principale difference entre la structure arborescente (en arbre) et la structure DAG est que chaque nœud dans la hiérarchie arborescente a juste un seul parent, tandis que dans la hiérarchie DAG chaque nœud

peut hériter de plusieurs parents, comme le montre la figure suivante :

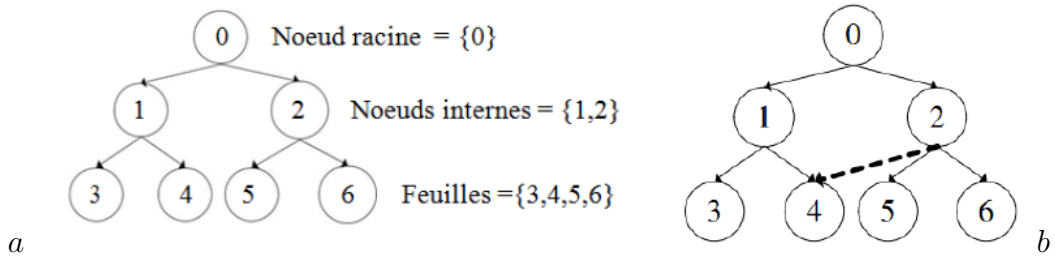


FIGURE 2.2 – Une hierarchie structuree par le DAG (b) et Une hiérarchie arborescente (a)

— Le deuxième critère est lié à la façon dans la classification est effectuée au sein de la structure hiérarchique :

– La méthode de classification hiérarchique peut être mise en œuvre de façon à ce que toute instance est affectée impérativement aux nœud (classes) feuilles[46]

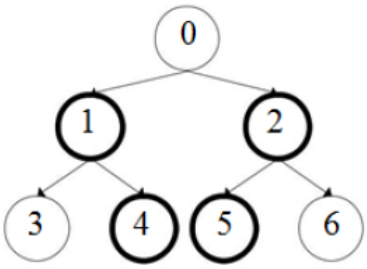


FIGURE 2.3 – Affectation cohérente

– La méthode de classification hiérarchique peut envisager d'arrêter le processus de classification à n'importe quel nœud de n'importe quel niveau de la hiérarchie (NMLNP)[47].

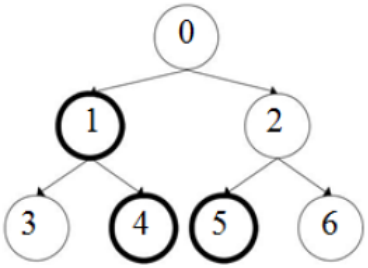


FIGURE 2.4 – Affectation incohérente

— Le troisième critère est lié à la façon dont la structure hiérarchique est exploitée. Dans [47] les auteurs ont résumé les trois approches existantes dans le domaine de la classi-

fication hiérarchique, incluant, l'approche de classification plate, l'approche de classification locale (Top-down) et l'approche globale (Big bang).

2.5.2.1 *Approche de classification plate (Flat classification)*

L'approche la plus simple est de transformer le problème hiérarchique en un problème de classification plate [48]. Cette approche ne tient pas compte de la hiérarchie de classes, et ne traite qu'avec les nœuds feuilles, soit en utilisant un seul classifieur multi-label ou un ensemble de classifieurs binaires (un pour chaque nœud feuille).

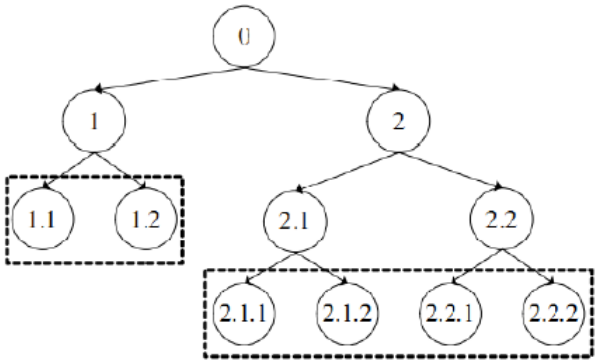


FIGURE 2.5 – Approche de la classifcation plate

2.5.2.2 *Approche de classification locale (Top-down)*

C'est l'approche la plus utilisée dans le domaine de la classification hiérarchique [[48]]. Un ou plusieurs classifieurs sont formés pour chaque nœud (classe) ou pour chaque niveau de la hiérarchie. On obtient ainsi un arbre de classeurs dans la structure est en arbre ou un graphe de classifieurs dans la structure DAG. Ces classifieurs sont construits à partir d'informations locales.

Elle est aussi appelée "approche Top-down" [[48]] parce que les phases d'apprentissage et de test se déroulent de façon descendante.

2.5.2.3 *Approche de classification globale (Big bang)*

Cette approche construit un seul grand modèle de classification pour toute les classes de la hiérarchie [46,48], comme le montre la Figure 2.6.

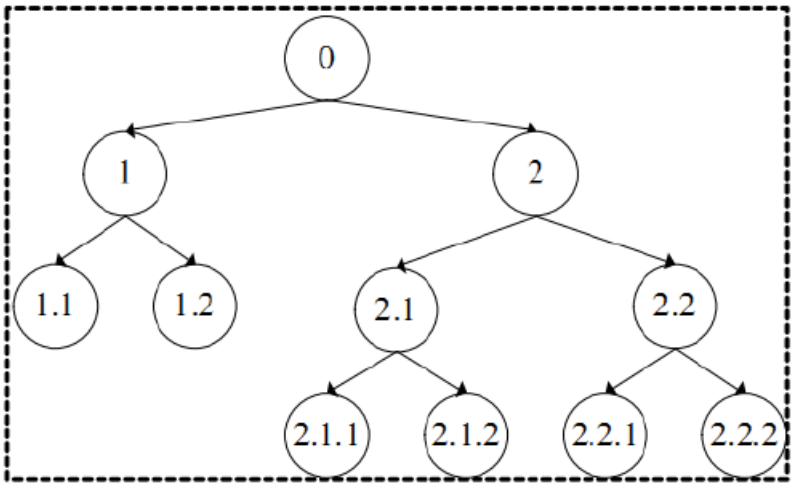


FIGURE 2.6 – Approche de classification globale, le réctangles représente l'unique classifieur utilisé pour toute la hiérarchie de classes

Cette approche présente de nombreux avantages :

1. La taille du modèle de classification globale est généralement plus petite que la taille totale de tous les modèles de classification locale combinés.

2. Les dépendances entre les classes de la hiérarchie sont prises en compte.

Cependant, il y a aussi certains inconvénients :

1. La complexité algorithmique est plus importante que dans les autres approches.

2. Un seul classifieur peut ne pas être en mesure d'identifier de nouvelles instances (inconnues) à travers un très grand nombre de classes.

2.6 Problématique

La classification multi-label hiérarchique diffère de la classification classique de deux (02) façons : *Premièrement*, chaque instance peut appartenir à plusieurs classes simultanément (classification multi-label). *Deuxièmement*, les classes ne sont pas indépendantes les unes des autres. Cette relation entre classes est exprimée sous forme d'une structure arborescente ou de graphe orienté acyclique (DAG) (classification hiérarchique).

Cette différence nous à mener d'une part à structurer ces classes, entre autres, choisir une méthode de classification adaptée, prenant en compte ou non le lien entre les labels ou bien

corriger quelques résultats d'une première classification et d'une autre part de mesurer le degré de corrélation de celles-ci.

2.7 Conclusion

Dans ce chapitre nous avons présenté la classification multi-label avec ses différentes approches qui sont : l'apprentissage par transformation de problème qui transforme le problème multi-label en un ou plusieurs problèmes mono-label en utilisant des classifieurs différents (BR, CC, LP et CLR), l'apprentissage par adaptation qui apprend des algorithmes d'apprentissage pour des données multi-label en se basant sur un ensemble d'algorithmes, entre autres, l'algorithme ML-KNN, ML-C4.5 et SVM, et l'apprentissage d'ensemble qui utilise un ensemble de classifieurs issus de la première ou de la deuxième famille d'approches (RAkEL, HOMER et l'ensembles de chaînes de classification).

Nous avons aussi définit la classification multi-label hiérarchique qui est utilisée pour organiser l'ensemble de labels en une hiérarchie généralement sous forme d'un arbre ou d'un graphe acyclique dirigé (DAG).

Cette hiérarchie est essentielle vu qu'elle nous permet d'avoir plus de détails. Delà qu'on introduit l'analyse de concepts formels et treillis de Galois qui sera l'objet de notre prochain chapitre.

Chapitre 3

ANALYSE DES CONCEPTS FORMELS

3.1 Introduction

L'Analyse de Concepts Formels (que nous désignerons par ACF dans la suite de ce document) a été introduite par Wille en 1982, puis consolidée mathématiquement par Ganter et Wille. L'ACF consiste à induire des paires de sous-ensembles $< objets, propriétés >$, appelées concepts formels, à partir d'une relation binaire entre un ensemble d'objets et un ensemble de propriétés. Elle a été utilisée dans divers domaines : psychologie, sociologie, biologie, médecine, linguistique, mathématiques, informatique, etc...

L'idée de Wille était d'illustrer dans un premier temps les paires (objets/propriétés) ainsi que la relation d'incidence selon une représentation mathématique appelée contexte formel. Ce dernier est considéré comme un outil de description des situations élémentaires sous la forme : l'objet "x" possède la propriété "y". Dans un deuxième temps, il s'agit de découvrir les ensembles maximaux d'objets satisfaisant un certain ensemble de propriétés.

Les connaissances induites appelées concepts formels sont hiérarchisées et représentées sous la forme d'un treillis de Galois. Les treillis de Galois constituent un moyen efficace permettant d'avoir une représentation exhaustive de la réalité étudiée (contexte formel).

Exemple

Soit un ensemble d'animaux {lion, rouge-gorge, aigle, lièvre, autruche} décrit par rapport à certaines de leurs propriétés {prédateur, vole, ovipare, mammifère}. Le contexte formel noté C (autrement dit la relation binaire) est représenté sous forme d'une table avec en lignes les animaux (correspondant aux objets) et en colonnes les propriétés, telle que si l'objet "x_i" vérifie (resp. ne vérifie pas) la propriété "a_j" alors la cellule "c_{ij}" est marquée par une croix (resp. reste vide).

	Prédateur	Vole	Ovipare	Mammifère
Lion	X			X
Rouge_gorge		X	X	
Aigle	X	X	X	
Autruche			X	
Lièvre				X

TABLE 3.1 – Représentation du contexte formel

Pour expliquer la notion de concept formel, nous considérons toutes les propriétés du rouge-gorge {vole, ovipare} et posons-nous la question : quels sont les animaux satisfaisant ces propriétés ? Nous obtenons alors l'ensemble X= {aigle, rouge-gorge}. Nous constatons que l'ensemble X est l'ensemble maximal des animaux satisfaisant toutes les propriétés de l'ensemble A= {vole, ovipare}.

Il résulte que X est l'ensemble de tous les animaux vérifiant toutes les propriétés de A et A est l'ensemble de toutes les propriétés vérifiées par tous les animaux de X. La paire <X,A> est appelé concept formel. Tandis que X est appelé extension et A est appelé intension.

3.2 Hiérarchie entre concepts formels

Entre les concepts formels, il y a une relation d'ordre partiel c.à.d. la relation "Sous-concept, Super-concept". Etant donné deux concepts formels <X,A> et <Y,B>. On dit que <X,A> est un sous-concept de <Y,B>, (dualement <Y,B>. est un super-concept de <X,A>) si l'extension de <X,A> est un sous ensemble de l'extension de <Y,B>. c.à.d. X⊆Y(dualement : l'intension de <X,A> est un sur-ensemble de l'intension de <Y,B>. c.à.d. A⊇B). [49]

Reprenons l'exemple précédent. Etant donné les deux concepts formels suivants : <{Aigle},-prédateur, vole, ovipare}> et <{Aigle, Rouge-gorge},{vole, ovipare}>. <{Aigle},{prédateur, vole, ovipare}> est un sous concept de <{Aigle, Rouge-gorge},{vole, ovipare}>. Dualement, <{Aigle, Rouge-gorge},{vole, ovipare}> est un super-concept de <{Aigle},{predateur,vole,ovipare}>. L'extension {Aigle} du sous-concept est un sous ensemble de l'extension {Aigle, Rouge-gorge} du super-concept. De la même manière l'intension {prédateur, vole, ovipare} du sous-concept est un sur-ensemble de l'intension {vole, ovipare} du super-concept. Dans la figure 3.1, nous représentons la hiérarchie de tous les concepts formels du contexte représenté dans le tableau 3.1.

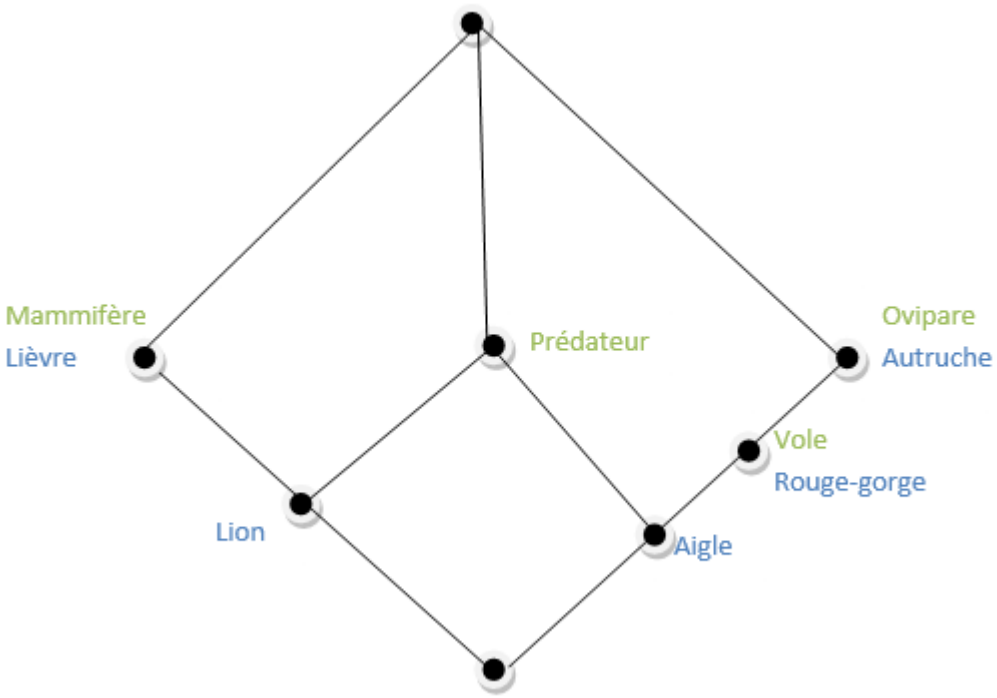


FIGURE 3.1 – Hiérarchie (treillis) de concepts formels

Cette Figure est constituée de nœuds et de segments. Elle comporte aussi les noms de tous les objets et toutes les propriétés du contexte. Chaque nœud correspond à un concept formel. La Figure peut être lue comme suit : A chaque fois qu'un nœud "n" est étiqueté par une propriété "a", tous les objets descendants de ce nœud "n" héritent la propriété "a". De façons duale, à chaque fois qu'un nœud "n" est étiqueté par un objet "x", "x" est hérité vers

le haut et tous les ancêtres du nœud "n" le partagent. Ainsi l'extension "X" d'un concept formel <X,A> correspondant au nœud "n" est obtenue en considérant tous les objets qui apparaissent sur les descendants du nœud "n" dans le treillis et son intension A est obtenue en considérant toutes les propriétés qui apparaissent sur les ancêtres du nœud "n" dans le treillis.

Nous pouvons remarquer que dans cet exemple, l'intension du concept formel sommet correspond à l'ensemble vide, tandis que l'extension du concept formel sommet correspond à l'ensemble de tous les animaux. Nous pouvons trouver toute fois un concept formel sommet différent de l'ensemble vide.

3.3 Rappels mathématiques :

3.3.1 Ordre et ordre partiel

3.3.1.1 *Définition 1 (Relation binaire)*

Une relation binaire R entre deux ensembles O et P est un ensemble de couples d'éléments (x,a) tels que x∈O et a∈ P, autrement dit un sous ensemble du produit Cartésien O×P.

(x ,a)∈R (aussi noté par x R a) signifie que l'élément x est en relation $\mathscr{R}$ avec l'élément a.

Si O=P, on parle de relation binaire sur O. [50]

R^{-1} est la relation inverse de $\mathscr{R}$, i.e. la relation entre P et O telle que a R^{-1} $x \Leftrightarrow xRa$.

3.3.1.2 *Définition 2 (Relation d'ordre (partiel))*

Une relation binaire $\mathscr{R}$ sur un ensemble E est dite relation d'ordre partiel (ou simplement relation d'ordre) sur E si elle vérifie les conditions suivantes pour tout x,y,z∈E [51,52] :

1. (x,x)∈$\mathscr{R}$ ($\mathscr{R}$ est réflexive)
2. si (x,y)∈ R et (y,x)∈R alors x=y ($\mathscr{R}$ est antisymétrique)
3. si (x,y)∈ R et (y,z)∈R alors (x ,z)∈$\mathscr{R}$ ($\mathscr{R}$ est transitive)

Une relation d'ordre $\mathscr{R}$ est souvent notée par $\leq$ R^{-1} est notée par "$\geq$").

3.3.1.3 *Définition 3 (Ensemble ordonné)*

Un ensemble partiellement ordonné (ou simplement ensemble ordonné) est un couple (E,≤) où E est un ensemble et "≤" est une relation d'ordre sur E. Dans un ensemble ordonné (E,≤), deux éléments x et y de E sont dits comparables lorsque x≤y ou y≤x, autrement ils sont dits incomparables. Pour deux éléments comparables et différents, x ≤y et x≠y, on note x<y. Un sous ensemble de (E,≤) dans lequel tous les éléments sont comparables est appelé chaîne. Un sous ensemble de (E,≤) dans lequel tous les éléments sont incomparables est appelé anti-chaîne. [53,54]

3.3.1.4 *Définition 5 (Principe de dualité des ensembles ordonnés)*

Soit (E,≤) un ensemble ordonné, la relation inverse "≥" de "≤" est aussi une relation d'ordre sur E. "≥" est appelée duale de "≤" et (E,≥) est appelé le dual de l'ensemble ordonné (E,≤). Le diagramme de Hasse de (E,≥) peut être obtenu à partir de celui de (E,≤) par une simple symétrie horizontale. De plus, il est possible de dériver les propriétés duales de (E,≥) à partir des propriétés de (E,≤). [53,54]

3.3.2 Contexte formel

Un contexte formel est un triplet $K = (O, P, R)$, où O est un ensemble d'objets, P est un ensemble de propriétés et R est une relation binaire entre O et P appelée relation d'incidence de K et vérifiant $R \subseteq O \times P$ Un couple $(x, a)\epsilon R$(noté aussi xRa) signifie que l'objet $x\epsilon O$ possède la propriété $a\epsilon P$.[49]

Un contexte formel est représenté sous la forme d'un tableau où les lignes correspondent aux objets et les colonnes correspondent aux propriétés (Tableau 3.1).[49]

Un exemple de contexte formel reliant un ensemble d'objets $\{x_1, x_2, x_3, x_4\}$ à un ensemble de propriétés $\{a_1, a_2, a_3, a_4\}$ est illustré à travers la table 3.2. Les cases de la table sont remplies comme suit : si le ième objet x est en relation R avec la jème propriété a alors la case d'intersection de la ligne i et la colonne$j$ contient " × " sinon la case est vide.

Par exemple, dans le contexte formel représenté par la table 3.2, l'objetx_2 possède les

propriétés a_1, a_3, et l'objet $x_3$possède les propriétés a_1, a_2, a_3. Etc...

	$pridateur$	$vole$	$ovipare$	$mammifere$
$Lion$	×			×
$Rouge-gorge$		×	×	
$Aigle$	×	×	×	
$Autruche$			×	
$lièvre$				×

TABLE 3.2 – Représentation du contexte formel

3.3.3 Opérateurs de dérivation de Galois

Etant donné un objet x et une propriété a, soit$R(x) = \{a\epsilon P|xRa\}$ l'ensemble des propriétés satisfaites par l'objet x (xRa signifie que x possède la propriété a) et soit $R(a) + \{x\epsilon O|xRa\}$ l'ensemble des objets possédant la propriété a. On définit en ACF des correspondances entre les ensembles 2^O et 2^P. Ces correspondances sont appelés opérateurs de dérivation de Galois.

Soit Kun contexte formel. Pour tout $X \subseteq O$ et $A \subseteq P$, on définit l'opérateur ensembliste de dérivation de Galois, noté $(.)^\triangle$, comme suit :

— $X^\triangle$est l'ensemble des propriétés communes à tous les objets de X

$$X^\triangle = \{a\epsilon P|\forall x\epsilon O(x\epsilon X \longrightarrow xRa)\} = \{a\epsilon P—X \subseteq R(a)\}$$

— $A^\triangle$est l'ensemble des objets possédant toutes les propriétés de A :

$$A^\triangle == \{x\epsilon O—\forall a\epsilon P(a\epsilon A \longrightarrow xRa)\} = \{x\epsilon O—A \subseteq R(x)\}$$

Les applications $(.)^\triangle : 2^O \longrightarrow 2^P$et $(.)^\triangle : 2^P \longrightarrow 2^O$sont appelées opérateurs de dérivation entre l'ensemble des objets et l'ensemble des propriétés dans un contexte formel.

La composition de ces opérateurs produit deux opérateurs $(.)^{\triangle\triangle} : 2^O \longrightarrow 2^O$ et $(.)^{\triangle\triangle} : 2^P \longrightarrow 2^P$. Le premier opérateur permet d'associer à un ensemble d'objets X l'ensemble maximal d'objets dans O ayant les propriétés communes aux objets de X. Cet ensemble est noté par $X^{\triangle\triangle}$. De façon duale, le second opérateur permet d'associer à un ensemble de propriétés A l'ensemble maximal de propriétés dans P communes aux objets ayant les propriétés dans A. Cet ensemble est noté par $A^{\triangle\triangle}$.

Exemple

Dans la table 3.2, soient deux ensembles ordonnée $(2^O, \subseteq)$ et $(2^P, \subseteq)$

— P est un ensemble de propriétés dont un terme A est un sous-ensemble d'un ensemble de propriétés, $P = \{prédateur, vole, ovipare, mammifère\}$. Ici $A_1 \subseteq A_2$ signifie que le terme A_1est moins spécifique que le terme A_2 (par exemple, $\{ovipare, mammifère\} \subseteq \{prédateur, ovipare, mammifère\}$),

— O est un ensemble d'objets $O = \{x_1, x_2, x_3, x_4, x_5\}$et X un sous ensemble d'un ensemble d'objets

Les deux opérateurs de dérivation sont définis comme suit :

$(.)^{\triangle} : 2^O \rightarrow 2^P$, $(X)^{\triangle} = \{a \,\epsilon\, P \,|\, \forall\, x \,\epsilon\, O \,(x \,\epsilon\, X \Rightarrow x \,R\, a)\}$, et

$(.)^{\triangle} : 2^P \rightarrow 2^O$, $(A)^{\triangle} = \{x \,\epsilon\, O \,|\, \forall\, a \,\epsilon\, P \,(a \,\epsilon\, A \Rightarrow x \,R\, a)\}$

Tel que ;

$(X)^{\triangle}$représente l'ensemble des propriétés communes à tous les objets de X et $(A)^{\triangle}$représente l'ensemble des propriétés communes à tous les objets de A

La paire duale d'opérateurs $< (.)^{\triangle}, (.)^{\triangle} >$ constitue ainsi une connexion de Galois sur 2^O et 2^P qui permet d'induire des concepts formels.

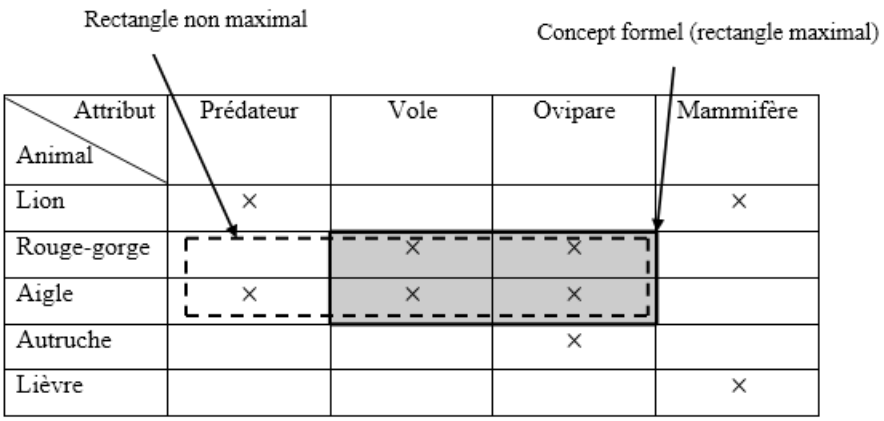
3.3.4 Concept formel

Soit $K = (O, P, R)$un contexte formel. Un concept formel est un couple $< X, A >$ $X \subseteq O$, $A \subseteq P$, tel que $X^{\triangle} = A$ et $A^{\triangle} = X$

X est l'extension (extent) du concept formel $< X, A >$ et A son intention (intent).

Dans un contexte formel, un concept formel correspond à un rectangle maximal de la table formée par la relation binaire du contexte : tout objet de l'extension a toutes les propriétés de l'intension. [49]

Le tableau 3.3 illustre un exemple de concept formel (représenté par le rectangle maximal). Le rectangle en pointillé n'est pas un concept formel car il n'est pas maximal.



Attribut / Animal	Prédateur	Vole	Ovipare	Mammifère
Lion	×			×
Rouge-gorge		×	×	
Aigle	×	×	×	
Autruche			×	
Lièvre				×

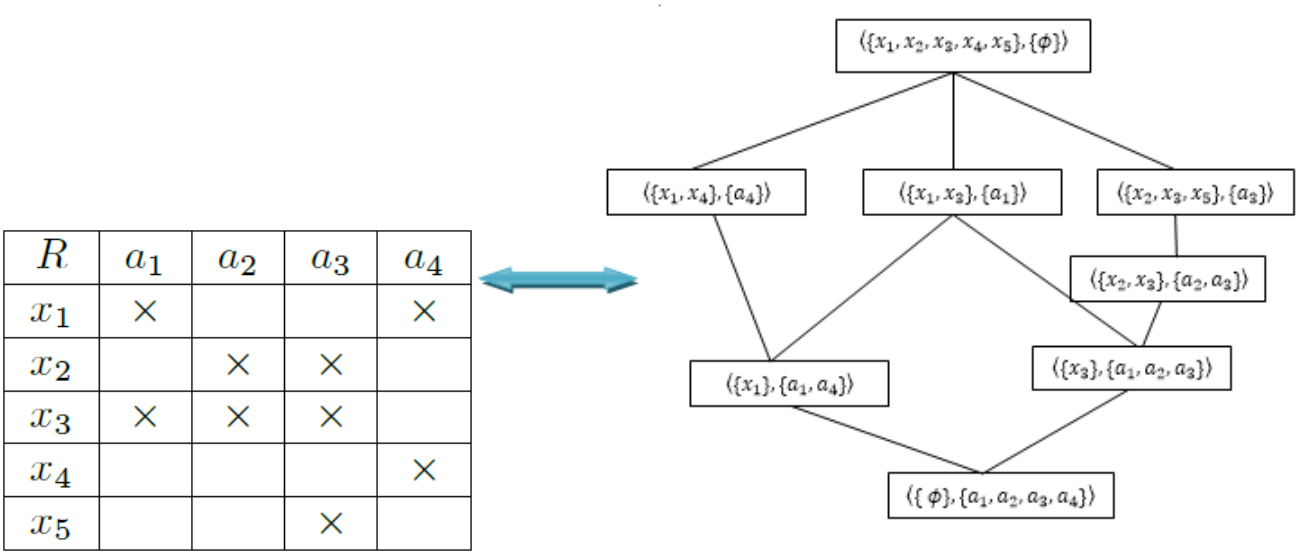
TABLE 3.3 – Exemlpe de concept formel

Il est important de noter que cette notion de rectangle maximal est indépendante de l'ordre des lignes et des colonnes. Ces ensembles maximaux d'objets et de propriétés sont à la base de la définition d'un concept formel. Un sous-ensemble A de P est l'intension d'un concept formel dans $B(O, P, R)$si et seulement si $A^{\Delta\Delta} = A$ (A est l'ensemble maximal de propriétés dans P communes aux objets ayant les propriétés dans A) et, de façon duale, un sous ensemble X de O est l'extension d'un concept formel dans $B(O, P, R)$ si et seulement si $X^{\Delta\Delta} = X$ (X est l'ensemble maximal d'objets dans O ayant les propriétés communes aux objets de X).

la construction du treillis de concepts formels d'une relation binaire donnée peut être décomposée en trois parties :

1. L'énumération des rectangles maximaux (les concepts formels),
2. La recherche de la relation d'ordre partiel entre ces concepts formels,
3. La construction du diagramme de HASSE correspondant au treillis.

L'exemple 2 illustre la relation d'équivalence entre un contexte formel et sa représentation sous forme d'un treillis de concepts formels



R	a_1	a_2	a_3	a_4
x_1	×			×
x_2		×	×	
x_3	×	×	×	
x_4				×
x_5			×	

FIGURE 3.2 – Treillis de concepts équivalent au contexte formel

De plus, la représentation graphique du treillis de concepts formels, sous la forme d'un diagramme de Hasse, facilite la compréhension et l'interprétation de la relation entre les objets et les propriétés d'une part et entre objets ou propriétés d'autre part.

3.4 Conclusion

Dans ce chapitre nous avons parlé sur l'Analyse des concepts formel où nous avons vu la représentation graphique du contexte formel en un treillis de galois.

Dans ce qui suit nous allons présenter deux (02) contributions, la première contribution consiste à utiliser l'ACF dans la classification multi-label hiérarchique et la deuxième c'est la mesure de similarité.

Chapitre 4

CONTRIBUTION

4.1 Introduction

Plusieurs études ont étudié de nouvelles alternatives pour résoudre le problème HMC (classification multi-label hiérarchique). Certains de ces études proposent de nouvelles méthodes comme le DAG et l'arborescente dont nous avons déja parlé dans le chapitre 2.

Dans ce chapitre nous avons proposé deux (02) contribution différentes **:**

— **La classification multi-label hiérarchique et treillis de Galois**

— **La mesure de similarité**

4.2 Contribution 1 (Classification multi-label hiérarchique et treillis de Galois)

Nous avons vu dans le deuxième chapitre que la classification multi-label hiérarchique traite des problèmes où les classes sont organisées sous forme d'une hiérarchie. Cette organisation hiérarchique est essentielle parce que d'une part elle permet de tenir compte des relations qui existent entre les classes et d'autre part elle permet à l'utilisateur de se concentrer sur le niveau approprié de détails.

Et dans le troisième chapitre, nous avons vu qu'en analyse de concepts formels les connaissances induites appelées concepts formels sont hiérarchisées et représentées sous la forme d'un

treillis de Galois qui constitue un moyen efficace permettant d'avoir une représentation de la réalité étudiée (contexte formel). Cette structure en treillis met en évidence les relations qui existent entre les objets d'une part mais aussi les relations entre les propriétés décrivant ces objets d'une autre part.

4.2.1 Enoncé du problème

le problème est défini comme suit : [**46**]

Étant donné un ensemble de données d'apprentissage $D = \{(x_i, Y_i)i = 1, ..., n\}$, l'objectif consiste à induire un classifieur pour effectuer la correspondance :

$\Phi : X \longrightarrow 2^Y$, où Y est un ensemble fini de classes et X un ensemble d'objet, de manière à optimiser les performances de classification.

Dans la classification multi-label hiérarchique, La structure des hiérarchies peut être classée en deux catégories principale, arborescente et graphe dirigé acyclique (DAG) et dans les deux cas chaque nœud de la hiérarchie représente une et une seule classe. Néanmoins, il est possible de représenter une autre forme de structure hiérarchique des classes qui est celle de la hiérarchie entre l'ensemble des parties 2^Y de l'ensemble des classe Y. Et pour cela, les treillis de Galois semblent être la meilleure solution.

la problématique de la classification multi-label hiérarchique est naturellement liée à la notion de concept. En effet, la classification multi-label consiste à regrouper les objets ayant les même classes. La notion de concept formel qui est un regroupement maximal d'objets possédant des propriétés en commun est par conséquent très proche, et le treillis de concepts peut être utilisé dans des applications de classification multi-label.

4.2.2 Approche de classification

La classification basée sur l'analyse de concepts formels consiste à construire des modèles appelés classifieurs à partir des données permettant de prédire les classes des futures données [49]. Elle vise à regrouper tous les groupement possibles entre les classes.

Comme c'est le cas dans la classification supervisée, cette opération est réalisée en deux

parties : une phase d'apprentissage dans laquelle un classifieur est construit pour décrire un ensemble prédéterminé de classes à partir d'un ensemble d'apprentissage et une phase de classification où le classifieur construit est utilisé pour associer un ensemble de classes à chaque nouvel objet.

4.2.2.1 *Phase d'apprentissage*

Dans l'approche présentée, dans le cadre de ce rapport, on commence notre phase d'apprentissage par l'organisation des données d'apprentissage sous forme d'un contexte formel reliant les objet aux classes comme représenté dans la tableau 3.1. Cette relation est représentée sous forme d'une table. Dans notre cas, les lignes représentent les objets et les colonnes les classes, chaque cellule exprime une valeur appartenant a $\{0, 1\}$. [49]

La deuxième phase : consiste à construire le treillis de Galois correspondant pour mettre en évidence les relations qui existent entre les classes. (Voir le chapitre 3) [49]

Plusieurs approches ont été proposés pour la construction de treillis de concepts formels : Kuznetsov [57], Chein [58], Norris [59], Ganter [60], [61], Bordat [62], et ont fait l'objectif de plusieurs comparaisons [63]. Dans leur article, Kuznetsov et Obiedkov [64] analysent plusieurs algorithmes de construction de treillis de concepts formels.

Dans ce qui suit le treillis de Gallois correspondant au contexte formel présenté dans le tableau 3.1

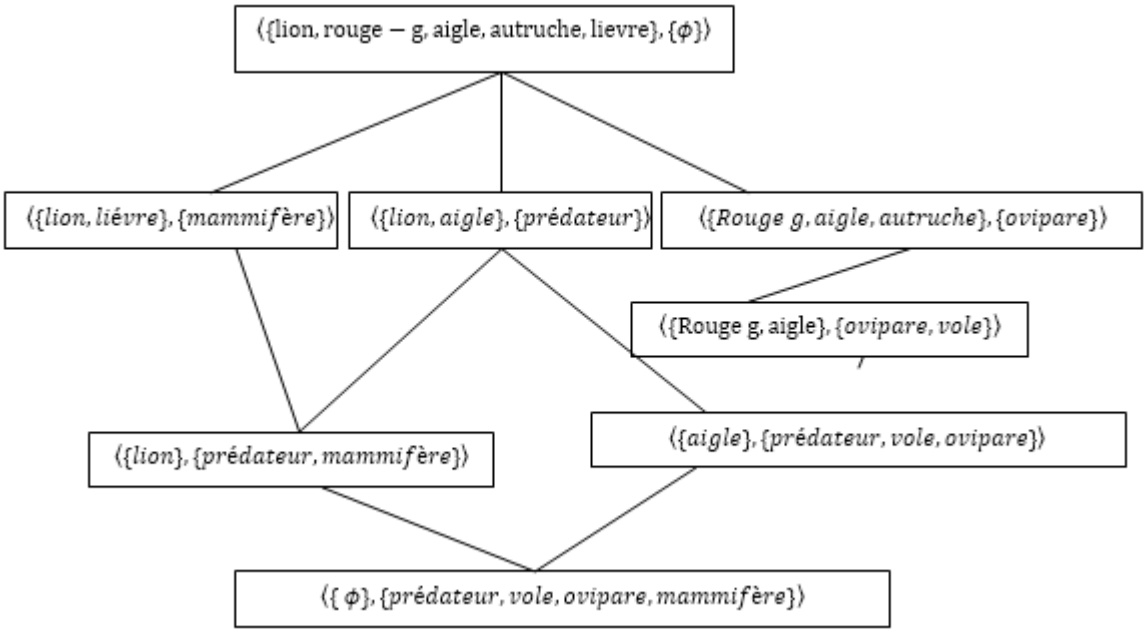


FIGURE 4.1 – treillis de Gallois

Ainsi on obtient une hiérarchie entre les concepts formels. Entre ces concepts formels, il y a une relation d'ordre partiel c.à.d. la relation "Sous-concept, Super-concept".

Etant donné deux concepts $< X, A >$et $< Y, B >$. On dit que $< X, A >$est un sous-concept de $< Y, B >$, (dualement $< Y, B >$est suepr-concept de $< X, A >$) si l'extention de $< X, A >$est un sous ensemble de l'extention de $< Y, B >$. c'est-à-dire $X \subseteq Y$ dualement : l'intention de $< X, A >$est un super-ensemble de l'intention de $< Y, B >$. c-à-d. $A \supseteq B$).

Reprenons l'exemple précédent. Etant donné les deux concepts formels suivants :

$< \{Aigle\}, \{prédateur, vole, ovipare\} >$ et $< \{Aigle, Rouge - gorge\}, \{vole, ovipare\} >$.

$< \{Aigle\}, \{prédateur, vole, ovipare\} >$ est un sous concept de $< \{Aigle, Rouge{-}gorge\}, \{vole, ovipare\} >$

Dualement, $< \{Aigle, Rouge{-}gorge\}, \{vole, ovipare\} >$ est un super-concept de $< \{Aigle\}, \{predateur, vol$

L'extension $\{Aigle\}$ du sous-concept est un sous ensemble de l'extension $\{Aigle, Rouge - gorge\}$ du super-concept. De la même manière l'intension $\{prédateur, vole, ovipare\}$ du sous-concept est un sur-ensemble de l'intension $\{vole, ovipare\}$ du super-concept.

Pour chaque noeud de la hiérarchie (treillis), un ou plusieurs classifieurs mono label sont induit pour prédire si une instance appartient ou pas à une classe présente dans l'intension

du concept formel (nœud) . Les classifieurs sont induits en suivant les contraintes suivantes : [65]

1. Un classifieur mono label h est induit pour chaque classe contenue dans l'intension Y du concept bottom (le plus générale), et le résultat de la prédiction sera la combinaison de tous les classifieurs mono label induits :

soit n le nombre de classe dans l'intention Y du bottom, $h_i(x)$ le classifieur monolabel associé à la classe i tel que :

$$h_i(x) = \{^{0\ Si\ x\ n'appartient\ pas\ à\ la\ classe}_{1\ sinon}$$

alors :

$H_Y(x) = \wedge_{i=0,n} h_i(x)$ tel que $H_Y(x)$est la prédiction que Y satisfait l'objet x

$$H_Y(x) = \{^{0\ Si\ x\ n'appartient\ pas\ à\ l'intention\ Y}_{1\ sinon}$$

2. Soit C_i un concept formel (un nœud du treillis) et$Y_i$son intension ($Y_i$est un ensemble de classes) :

— Si $C_i$contient un seul super concept C_j dans la hiérarchie, alors un classifieur mono label h est induit pour chaque nouvelle classe qui apparait dans Y_i, et le résultat de la prédiction sera la combinaison de ces classifieurs mono label avec le classifieur du super conceptC_j : Soit n le nombre de nouvelles classes apparue dans l'intension Y_i :

$H_{Y_i}(x) = H_{Y_j}(x) \underset{i=0,n}{\wedge} h_i(x)$

-$H_{Y_i}(x)$: La prédiction pour l'intensionY_i du sous concept C_i

$$H_{Y_i}(x) = \{^{0\ Si\ x\ n'appartient\ pas\ à\ l'intention\ Y_i}_{1\ sinon}$$

-$H_{Y_j}(x)$: La prédiction pour l'intensionY_j du super concept C_j

$$H_{Y_j}(x) = \{^{0\ Si\ x\ n'appartient\ pas\ à\ l'intention\ Y_j}_{1\ sinon}$$

-$h_i(x)$: La prédiction pour chaque classe apparue (qui appartient à Y_i mais pas à Y_j)

$$H_i(x) = \{^{0\ Si\ x\ n'appartient\ pas\ à\ la\ classe\ i}_{1\ sinon}$$

Exemple : en reprenant l'exemple précédent :

— Les noeuds $< \{mammifère >\}, < \{prédateur >\}$ et $< \{ovipare >\}$ n'ont qu'un seul super concept et dans chacun de ces nœuds les classes sont nouvelles, donc il faut induire un classifieur mono label pour chacune de ces classes.

— Idem pour le nœud $< \{ovipare, vole >\}$ qui a aussi un seul super concept dont l'intension est $< \{ovipare >\}$ alors on va construire un classifieur mono label pour chaque nouvelle classe apparue. Et dans ce cas la seule classe apparue est $< vole >$, et le résultat de la prédiction sera donc :

$$H_{<\{ovipare,vole>\}}(x) = H_{<vole>}(x) \wedge H_{<ovipare>}(x)$$

3.Soit $C_i$un concept formel (un nœud du treillis) et Y_i son intension (Y_i est un ensemble de classes) :

— Si$C_i$contient plusieurs super concept C_j $(j = 0, ..., n)$ dans la hiérarchie, alors un la prédiction sera la combinaison de ces classifieurs des supers concepts C_j :

$$H_{Y_i}(x) = \underset{j=0,n}{\wedge} H_{Y_j}(x)$$

Exemple :

le nœud $< \{prédateur, vole, ovipare\} >$ a deux supers concepts, $< \{ovipare, vole\} >$ et $< \{prédateur\} >$, le résultat de prédiction pour nœud sera la combinaison des prédiction de ces supers concepts :

$$H_{Y_i}(x) = \underset{j=0,n}{\wedge} H_{Y_j}(x) = H_{<\{ovipare,vole>\}}(x) \wedge H_{<\{prédateur\}>}(x)$$

les classifieurs, pour l'exemple précédent sont induit comme le montre la figure 4.2, ne nous interessons qu'aux intensions des concepts formels :

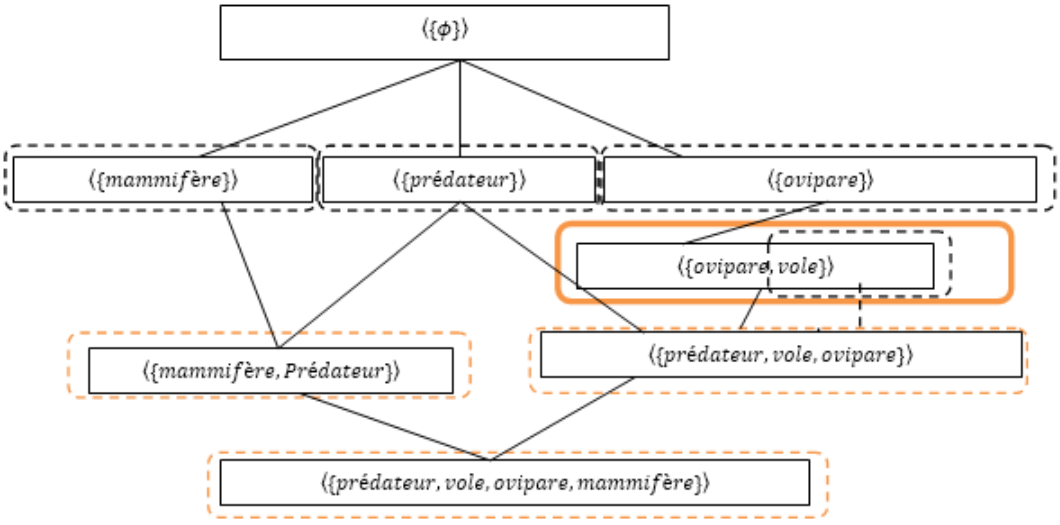


FIGURE 4.2 –

. . . Classifieur pour une nouvelle

Classifieur qui combine la prédiction d'une nouvelle classe avec les la prédiction du super concept

Classifieur qui combine les prédictions des classifieurs des super

4.2.2.2 *Phase de classification*

Une instance est classifiée de manière descendante (TOP-DOWN). On commence par le concept (TOP) : [46,48]

- Si l'intension du TOP est un ensemble vide alors on passe directement au sous concept.

-Sinon, l'instance est soumise aux classifieurs monolabel induit pour chaque classe réspectivement, et le résultat de la prédiction sera la combinaison des différentes sortie des ces classifieurs (comme vu dans la phase apprentissage).

Lorsqu'elle est affectée, l'instance sera soumise à une nouvelle classification qui concerne chaque intension de chaque sous concept du concept TOP (noeud racine). Dans notre exemple, l'instance sera soumise à trois classifieurs mono label afin de prédire si elle appartient aux classes mammifère, prédateur et ovipare respectivement. Cette procédure est répétée jusqu'à ce qu'on atteint le noeud bottom ou jusqu'à ce qu'aucune prédiction n'est possible.

Supposons qu'une instance inconnue est prédite dans les classes mammifère et prédateur

mais pas dans la classe ovipare. Ainsi cette instance n'a pas besoin de continuer le processus de classification de niveau inférieur concernant tous les sous concept du concept contenant *ovipare* comme intension $< \{ovipare, vole\} >$,$< \{prédateur, vole, ovipare\} >$, $< \{prédateur, vo-le, ovipare, mammifère\} >$ En prenant en considération la classification multi-label une instances inconnue peut être prédite par plusieurs classifieurs mono label, alors l'instance sera automatiquement prédite le classifieur du sous concept (reliant ces deux concepts). Puisque l'instance est prédite dans les classes mammifère et prédateur respectivement alors elle est aussi prédite dans l'intension du sous concepts les reliant $< \{mammifère, Prédateur\} >$.

4.2.3 Les règles d'association

Une quantité importante de données s'accumulent lors des opérations quotidiennes dans des entreprises commerciales. A tire d'exemple, le tableau 4.1 illustre la nature de données collectées par les achats des consommateurs dans les grands magasins. Chaque ligne de la table correspond à une transaction identifiée par un numéro (TID) ainsi que les produits achetés. Pour mieux comprendre le comportement des clients, les commerçants s'intéressent à l'analyse de données de chacun. [66]

Exemple 1 :

TID	items
1	Pain, riz
2	pain, céréales, lait, oeufs
3	riz, céréales, lait, coca
4	pain, riz, céréales, lait
5	pain, lait, céréales, coca

TABLE 4.1 – Panier de la ménagère

Le tableau 4.1 : montre qu'il existe une relation forte entre la vente des céréales et la vente de lait. Par conséquent, les clients qui achètent les céréales achètent aussi de lait. On représente cette relation par une règle d'association : céréales $\longrightarrow$ lait

D'une manière générale on représente une règle d'association par :

Antécédent $\longrightarrow$ Conséquent

Cette règle est lue comme suit : si une condition existe, alors forcément, un résultat issu de celle-ci existe aussi. [67]

Il y a deux problèmes clés qui doivent être considères quand on utilise des règles d'association : [68]

1. D'abord l'extraction de motifs peut être numériquement couteuse lorsque les bases de données sont importantes.
2. Deuxièmement, certaines associations sont potentiellement fausses ou sans intérêt, elles apparaissent simplement par hasard.

4.2.3.1 *Définitions*

La section suivante est consacrée à la définition concepts impliqués dans la recherche et l'extraction des règles d'association à savoir : transaction, item, support, confiance, règle d'association :

Transaction :

- **Théoriquement :** Une transaction est une succession d'items exprimée selon un ordre donné ; de même, l'ensemble de transactions contient des transactions de longueurs différentes. [69]
- **Mathématiquement :** On considère $\Gamma = I_1, ..., I_n$ un ensemble d'éléments binaires (items) et T une base de données des transactions telle que $T = t_1, .., t_m$. Chaque transaction T est représentée comme un vecteur binaire, avec$T[k] = 1$ si la transaction T achète l'item $\Gamma_k$sinon $T[k] = 0$. [66] [67]

Chaque transaction est identifiée par une clé unique.

Exemple 2 :

Soit$\Gamma = pain, riz, céréales, lait, œufs, coca$ l'ensemble de tous les items de panier et $T = 1, 2, 3, 4, 5$ l'ensemble de toutes les transactions. Le tableau 4.1 pourrait se mettre sous forme binaire comme suit :

TID	pain	riz	céréales	lait	oeufs	coca
1	1	1	0	0	0	0
2	1	0	1	1	1	0
3	0	1	1	1	0	1
4	1	1	1	1	0	0
5	1	1	1	0	0	1

TABLE 4.2 – Panier de la ménagère présenté en binaire

Items : est une liste limitée d'atomes ou d'attributs. [69]

Itemset, Itemset fréquent et support : [66] [67] Itemset : est un ensemble d'items ou encore est une succession d'items exprimée dans un ordre donné et prédéfini.

— **Mathématiquement :** Soient$\Gamma = I_1, ..., I_n$ l'ensemble de tous les items. L'ensemble I est un itemset si et seulement si $I \subseteq \Gamma$.

K-Itemset : est un Itemset de k Items.

— Un Itemset est dit fréquent si et seulement si son support est supérieur à un support minimum.

Support d'un itemset : Support d'un itemset est définit par le nombre de transactions qui le contient, divisé par le nombre total des transactions de T.

— **Mathématiquement** : Soit A un Itemset de n éléments. Dans une base de données transactionnelle T, le support de A est :

$support\,(A)$ ou $\sigma(A) = \frac{car(A)}{car(T)}$

Exemple 3 :

Dans les données de tableau 4.1 le support (céréales, lait) = 3/5 =0.6

Règle d'association, support et confiance :

— **La règle d'association :** est une méthode d'apprentissage non supervisé, permet de découvrir à partir d'un ensemble de transactions, un ensemble de règles qui exprime une possibilité d'association entre différents items. [69]

Soient A et B deux sous-ensembles d'items disjoints. Une règle d'association est de la forme $A \longrightarrow B$ possède deux métriques importantes pour mesurer sa force à savoir : le support et la confiance.

— **Le support d'une règle d'association :** c'est un indicateur de la fiabilité de la règle. Le support d'une règle d'association $A \longrightarrow B$ est le support $\sigma(A \cup B)$ divisé par le nombre total des transactions de T.

$\sigma(A \longrightarrow B) = \frac{car(A \cup B)}{car(T)}$

— **La confiance d'une règle d'association :** c'est un indicateur de précision de la règle. La confiance d'une règle d'association $A \longrightarrow B$ est le rapport entre le nombre de transactions de T contenant $(A \cup B)$, et le nombre de transactions de T contenant A. Ou encore c'est le support$\sigma(A \cup B)$, divisé par le support $\sigma(A)$:

$confiance$, $c(A \longrightarrow B) = \frac{\sigma(A \cup B)}{\sigma(A)}$

Remarque :

Une règle est dite bonne si et seulement si son support et sa confiance sont élevées.

4.2.4 Extraction des règles d'association

L'extraction de règles d'association est un processus constitué de trois phases allant de la sélection et la préparation des données jusqu'à la production des règles d'association, en passant par la phase de recherche des itemset fréquents : [70]

4.2.4.1 *Préparation des données*

Cette phase consiste à réduire la quantité des données (attributs et objets) tout en gardant seulement les plus pertinentes et transformer ces données en un contexte d'extraction(ou jeu de données). Ce contexte est un triplet $A = (O, I, R)$ dans lequel O est un ensemble d'objets, I est un ensemble d'attributs (items), et R est une relation binaire entre O et$I$.

Exemple 4 :

Soient$I = \{prédateur, vol, ovipare, mammifère\}$ un ensemble d'items et $O = \{lion, rouge-gorge, aigle, autruche, lièvre\}$ un ensemble d'objets tels que présentés dans le tableau 4.3. Soit R une relation binaire entre les items de l'ensemble I et les objets de l'ensemble O. Chaque couple de la relation binaire, associant un item $i \epsilon I$et un objet $o \epsilon O$ est représenté par : oRi.

Items Objets	Prédateur	Vole	ovipare	mammifère
Lion	×			×
Touge-gorge		×	×	
Aigle	×	×	×	
Autruche			×	
lièvre				×

TABLE 4.3 – L'ensemble des items et des objets.

4.2.4.2 *Recherche des itemset fréquents*

L'Idée est d'extraire du contexte tous les itemsets fréquents (F) en minimisant les calculs, notamment le nombre d'accès à la base de données. Ces itemsets forment un treillis dont la représentation sous forme de diagramme de Hasse pour le contexte D est présentée dans la Figure 4.3. Cette étape est très coûteuse en temps d'exécution. Pour un ensemble de n items par exemple, le nombre d'ltemsetsr fréquents qui peut être générés est de2^n.

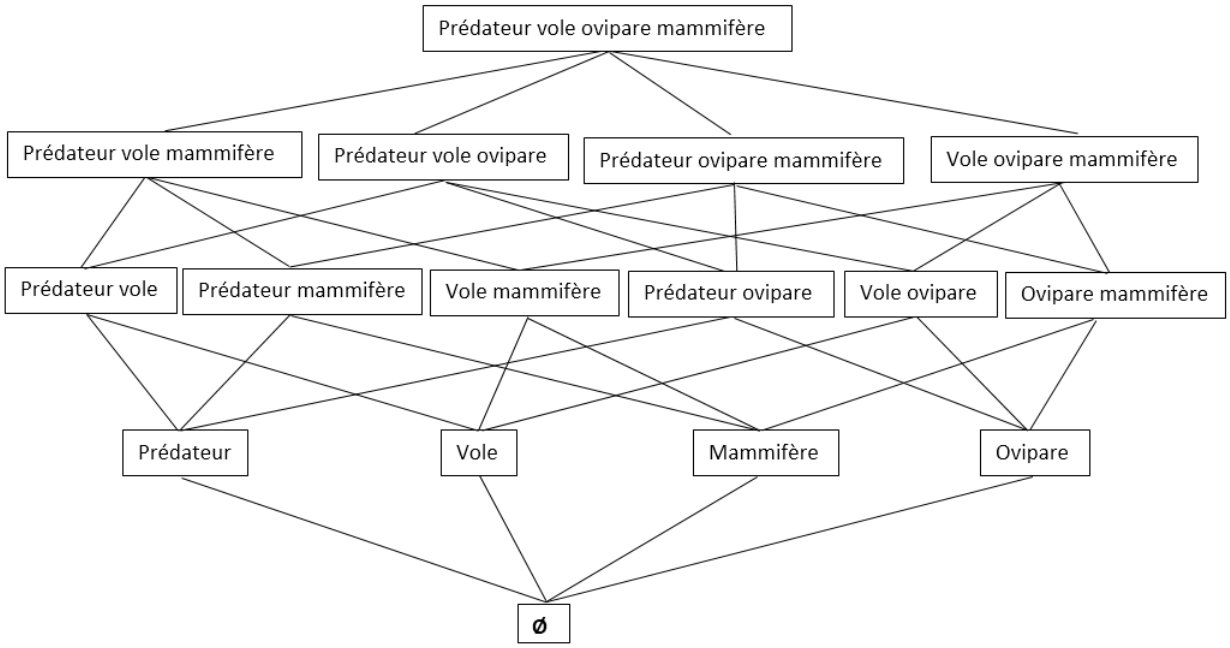


FIGURE 4.3 – Diagramme de Hasse représentant le treillis des itemsets.

4.2.4.3 *La production des règles d'association*

Cette phase consiste à trouver toutes les règles d'association ayant un support supérieur ou égal à minsup et une confiance supérieur ou égal à minconf où minsup et minconf sont des seuils pour le support et la confiance.

— Pour chaque itemset fréquent nommé IF, générer tous les sous-ensembles non vide IF .

— Pour chaque sous-ensemble non vide e de IF, sauvegarder la régle $e \Rightarrow IF - e$ si confiance $(e \Rightarrow IF - e) \geq min\,conf$

Remarque :

si la régle $AB \Rightarrow CD$ n'est pas valide alors les régles : $A \Rightarrow BCD$ et $B \Rightarrow ACD$ ne sont pas valide.

L'algorithme apriori : Dans cette section on décrit comment la mesure de support aide à réduire le nombre d'ensembles d'items candidats considérés durant la génération des ensembles d'items fréquents. Plusieurs algorithmes ont été proposés permettant de minimiser le nombre d'itemsets candidats. [71]

Définition : L'algorithme Apriori (Agrawal et Srikant, 1994) basé sur le couple support-confiance est le premier modèle efficace qui traite le problème de l'extraction des règles d'association. C'est un algorithme par niveaux qui permet la recherche des itemset les plus fréquents. Le but est de limiter le nombre des itemset candidats en traquant les itemsets les plus fréquents répondant au seuil de support minimal. [72

Principe de l'algorithme apriori [73] :

— Génération d'ensemble d'itemset.

— Calculs des fréquences des ensembles d'itemset.

— On garde les ensembles d'itemset fréquents avec un support minimum.

Cet algorithme se base principalement sur les deux règles suivantes [2] :

Soit S un Itemset et $S' \subseteq S$ alors :

1. Si S est non fréquent, alors les Itemsets qu'on construit de S (c.à.d S') sont aussi non fréquents.
2. Si S est fréquent, alors S' est aussi fréquent.

Déroulement d'Apriori : [67] Le déroulement de l'algorithme Apriori peut être décomposé en plusieurs étapes :

1. On commence par chercher les itemsets fréquents de longueur 1.
2. Combinaison des itemsets pour obtenir des itemsets de longueur 2 et on garde que les fréquents parmis eux.
3. On combine ces itemsets pour obtenir des itemsets de longueur 3 et on garde que les fréquents parmis eux.
4. On continue jusqu'à la longueur maximale.

Nous avons utilisé les notations suivantes dans les deux (02) algorithmes ci-dessous :

L : les nouveaux condidats.

k : itemset fréquent.

C : Les condidats contenus dans la base de données.

t : la base de données

L'Algorithme Apriori : Input : Contexte B ; seuil minimum de support minsupp.

Output : Ensemble $L_k$des k-itemsets fréquents.

$1 : L_1 \longleftarrow \{\, 1 - itemsets\ fréquents \,\}$

2 : for ($k \longleftarrow 2$; $L_{k-1} \neq \phi$; $k++$) do

3 : $C_k \longleftarrow$Apriori-Gen (L_{k-1})

4 : for all (instance $t \,\epsilon\, B$) do

5 : $C_t \longleftarrow$ Subset C_k, t)

6 : for all (candidat $c \epsilon\, C_t$) do

7 : c.support ++

8 : end for

9 : $L_k \longleftarrow \{c \epsilon C_k \,|\, c.support \geq minsupp\}$

10 : End for

11 : Retourner $U_k\, L_k$

12 : end for

— **Procédure Apriori-Gen :**

Input : Ensemble L_{k-1}de $(k-1)$ itemsets fréquents

Output : Ensemble $C_k$de k-itemsets candidats

1 : insert to C_k

2 : select $p.item_1 \ldots, p.item_{k-1}, q.item_{k-1}$

3 : form L_{kp}, L_{kq}

4 : where $p.item_1 = q.item_1$, …, $p.item_{k-2} = p.item_{k-2}$, $p.item_{k-1} < q.item_{q-1}$

5 : for all (itemsets $c \epsilon C_k$) do

6 : for all ($(k-1)$ subsets s of C) do

7 : if ($s \notin L_{k-1}$)then

8 : delete c from C_k

9 : end if

10 : end for

11 : end for

12 : Retourner C_k

— **Points faible d'Apriori :** [7]

1. Le calcul des supports est couteux.
2. La génération des règles est couteuse.
3. Le parcours des données initiales est récurrent.

Exemple 5 : Supposons $min\,supp = 0,25$

$C1$

Items objets	Prédateur	vole	ovipare	mammifère
1	×			×
2		×	×	
3	×	×	×	
4			×	
5				×

$\Longrightarrow$

$L1$

items	support
{Prédateur}	$\frac{2}{5}$
{Vole}	$\frac{3}{5}$
{Ovipare}	$\frac{3}{5}$
{Mammifère}	$\frac{2}{5}$

$\Downarrow_{parcours\,t}$

$L2$

Items	Support
{Prédateur Vole}	$\frac{1}{5}$
{Prédateur Ovipare}	$\frac{1}{5}$
{Prédateur Mammifère}	$\frac{1}{5}$
{Vole Ovipare}	$\frac{3}{5}$
{Vole Mammifère}	0
{Ovipare Mammifère}	0

$\Longleftarrow$

$C2$

Itemsets
{Prédateur Vole}
{Prédateur Ovipare}
{Prédateur Mammifère}
{Vole Ovipare}
{Vole Mammifère}
{Ovipare Mammifère}

$\Downarrow_{parcours\,t}$

$C3$

Itemsets
{Vole Ovipare}

$\Rightarrow$

$L3$

Itemsets	Support
{Vole Ovipare}	$\frac{2}{5}$

TABLE 4.4 – Exemple 5

$C1 = \{prédateur, vole, ovipare, mammifère\}$

Du tableau $C1$ ***vers*** $L1$ ***:*** Génération des candidats de taille 1, tous les supports supérieures à $minsup$ donc tous les condidats sont des itemsets fréquents d'où $L1 = C1$

Du tableau $L1$ ***vers*** $C2$ ***:*** Génération des candidats de taille 2, $C2 = \{prédateur\,vole, prédateur\,ovipar$

Du tableau $C2$ ***vers*** $L2$ ***:*** Calcul des supports.

Du tableau $L2$ ***vers*** $C3$ ***:*** Suppression des condidats dont le support est inférieur au minsupp $C3 = \{Vole\,Ovipare\}$ et $L2 = C3$.

Du tableau $C3$ ***vers*** $L3$ ***:*** Calcul de support et génération des condidats de taille 3, ce qui est impossible, alors $C4 = \phi$donc $L3 = \{Vole\,Ovipare\}$

— L'algorithme appriori retourne alors l'ensemble des itemsets fréquents : $L1 \cup L2 \cup L3$

— $L1 \cup L2 \cup L3 = \{prédateur, vole, ovipare, mammifère, vole\,ovipare\}$

A partir de l'exemlpe précédent nous avons pu extraire une seule régle d'association qui est $Vole \Rightarrow Ovipare$ tandis que les autres sont toutes fausses.

4.2.5 Avantages et inconvénients des règles d'association

4.2.5.1 *Les Avantages*

— Les règles d'association peuvent être appliquées dans plusieurs domaines soit dans des activités commerciales, sociales ou humaines. [66][67]

— Grace aux règles d'association des données et des connaissances utiles sont découvertes dans la base de données.

— Elles facilitent et elles simplifient l'interprétation et la compréhension des résultats.

— Le formalisme non supervisé et général.

4.2.5.2 *Les Inconvénients*

— La production d'un nombre important des règles d'association dont la plupart sont triviales et inutiles et qui n'apportent pas de nouvelles informations. [66][67]

— Le temps de recherche des Itemsets fréquents est énorme.

— Les algorithmes utilisés ont trop de paramètres, par conséquent l'extraction de données, pour les non experts, devient compliquée.

4.3 Contribution 2 : Mesure de similarité

4.3.1 L'écart type

Les mesures de dispersions sont des indices qui caractérisent l'étalement des valeurs d'une distribution d'une variable autour d'une valeur centrale. Elles permettent de savoir si les scores individuels se rapprochent ou s'éloignent beaucoup des mesures de tendance centrale. Il existe différentes façons d'exprimer la dispersion : La plus courante est l'écart-type.[**74**]

4.3.1.1 *Définition*

L'Ecart type nommé S est un outil statistique qui permet de calculer la dispersion d'un ensemble de valeurs par rapport à la moyenne de ces valeurs. L'écart type se calcule par la racine carrée de la variance[75]. Plus l'écart type est grand, plus les données sont éloignées de chaque côté de la moyenne et vice versa pour un écart type qui est petit. [76]

Alors, pour calculer l'écart type, il suffit juste de suivre les étapes qui suivantes : [77]

1. On calcule la moyenne arithmétique de l'échantillon.

$$\bar{x} = \frac{1}{n} \sum x_i$$

2. On calcule le carré de l'écart à la moyenne de chacune des valeurs de l'échantillon.

$$\mid x_i - \bar{x} \mid^2$$

3. On calcule la somme des valeurs obtenues.

$$\sum \mid x_i - \bar{x} \mid^2$$

4. On divise par la taille de l'échantillon.

$$\frac{\sum |x_i - \bar{x}|^2}{n}$$

5. Et on calcul la racine carrée de résultat obtenu.

$$\sqrt{\frac{\sum |x_i - \bar{x}|^2}{n}}$$

$$\text{Donc}\,;\; S = \sqrt{\frac{\sum |x_i - \bar{x}|^2}{n}}$$

Tel que : [77]

$\sum$: signifie « somme de ».

x_i représente la $i^{éme}$ valeur de l'échantillon.

$\bar{x}$ représente la moyenne de l'échantillon (la moyenne arithmétique) :

où

$\bar{x} = \frac{x_1 + x_2 + x_3 + ... + x_n}{n} = \frac{1}{n} \sum x_i$

n représente la taille de l'échantillon (l'effectif total).

4.3.1.2 *Propriétés de l'écart-type*

• *S=0* si toutes les valeurs d'un ensemble de données sont les mêmes (parce que chaque valeur est égale à la moyenne).

• *S>0.* sinon l'écart type est toujours supérieure à zéro (02)

Exemple :

Soit l'ensemble A tel que $A = \{4, 2, 9, 7, 8\}$

On calcule l'écart type de cet ensemble :

On a n = **5**

La première étape : on calcul la moyenne arithmétique $\bar{x}$:

$\bar{x} = \frac{1}{n}\sum x_i = \frac{4+2+9+7+8}{5} = \frac{30}{5} = 6$

La deuxième étape : on calcul $\mid x_i - \bar{x} \mid^2$:

x	$\mid x_i - \bar{x} \mid^2$
4	$\mid 4-6 \mid^2 = 4$
2	$\mid 2-6 \mid^2 = 16$
9	$\mid 9-6 \mid^2 = 9$
7	$\mid 7-6 \mid^2 = 1$
8	$\mid 8-6 \mid^2 = 4$

TABLE 4.5 – Calcul de $\mid x_i - \bar{x} \mid^2$

La troisième étape : on calcul la somme des résultats obtenu dans la deuxième étape :

$\sum \mid x_i - \bar{x} \mid^2 = 4 + 16 + 9 + 1 + 4 = 34$

La quatrième étape : on divise le résultat de la troisième étape par la taille de l'échantillon :

$\frac{\sum |x_i - \bar{x}|^2}{n} = \frac{34}{5} = 6,8$

Et dans la dernière étape on calcule la racine carrée de résultat obtenu dans l'étape numéro 4 :

$\sqrt{\frac{\sum |x_i - \bar{x}|^2}{n}} = \sqrt{6,8} \simeq 2,6$

Alors $S \simeq 2,6$

4.3.1.3 *Application de l'écart type*

Dans ce travail, nous avons calculé l'écart type pour toutes les classes d'attribut du fichier ARFF (Pour tout le DATASET).

L'idée est de parcourir tout le DATASET afin de trouver les répétitions de chaque objet pour qu'à la fin on déduise un nouvel ensemble (une nouvelle classe) contenant l'effectif de chaque objet sur lequel on appliquera le calcul de l'écart type en suivant les mêmes étapes d'un calcul ordinaire (un seul écart type pour tout le DATASET).

Le calcul de l'écart type :

Dans l'exemple ci-dessous nous avons cinq (04) classes et onze (06) objets

$C_n(O_1, O_2, O_3, O_4, O_5, O_6)$

$C_1(1, 1, 1, 1, 0, 0)$

$C_2(1, 0, 1, 1, 1, 1)$

$C_3(1, 0, 1, 1, 0, 0)$

$C_4(0, 1, 0, 0, 0, 1)$

En effet, on peut déduire une série statistique contenant les répétitions de chaque objet dans toutes les classes :

Objets	O_1	O_2	O_3	O_4	O_5	O_6
Effectifs	3	2	3	3	1	2

TABLE 4.6 – Tableau des effectifs

Alors ; $S(3, 2, 3, 3, 1, 2)$

Le calcul de l'écart type se basera sur cette nouvelle série obtenue

On a : $n = 6$ (le nombre d'objet)

La première étape : on calcul la moyenne arithmétique $\bar{x}$:

$\bar{x} = \frac{1}{n}\sum x_i = \frac{3+2+3+3+1+2}{6} = \frac{14}{6} = \frac{7}{3}$

La deuxième étape : on calcul $\mid x_i - \bar{x} \mid^2$:

x	$\mid x_i - \bar{x} \mid^2$
3	$\mid 2 - \frac{7}{3} \mid^2 = \frac{4}{9}$
2	$\mid 3 - \frac{7}{3} \mid^2 = \frac{1}{9}$
3	$\mid 5 - \frac{7}{3} \mid^2 = \frac{4}{9}$
3	$\mid 2 - \frac{7}{3} \mid^2 = \frac{4}{9}$
1	$\mid 1 - \frac{7}{3} \mid^2 = \frac{16}{9}$
2	$\mid 1 - \frac{7}{3} \mid^2 = \frac{1}{9}$

TABLE 4.7 – Calcul de $\mid x_i - \bar{x} \mid^2$

La troisième étape : on calcul la somme des résultats obtenu dans la deuxième étape :

$\sum \mid x_i - \bar{x} \mid^2 = \frac{4}{9} + \frac{1}{9} + \frac{4}{9} + \frac{4}{9} + \frac{16}{9} + \frac{1}{9} = \frac{30}{9} = \frac{10}{3}$

La quatrième étape : on divise le résultat de la troisième étape par la taille de l'échantillon :

$\frac{\sum |x_i - \bar{x}|^2}{n} = \frac{\frac{10}{3}}{6} = \frac{10}{18} = \frac{5}{9}$

Et dans la dernière étape on calcule la racine carrée de résultat obtenu dans l'étape numéro 4 :

$\sqrt{\frac{\sum|x_i-\bar{x}|^2}{n}} = \sqrt{\frac{5}{9}} \simeq 0,745\,;$ Alors $S \simeq 0,745$

Une fois le calcul terminé, on peut déduire dans, notre cas, que la dispersion est forte puisque l'écart-type est inférieur à $\frac{7}{6}$ ($\frac{1}{2}$ moyenne).

Or, puisque l'écart type est inférieur à $\frac{1}{2}$ de la moyenne arithmétique alors il y a de forte corrélation entre les classes du fichier ARFF

4.3.1.4 *L'interprétation de l'écart type*

- En cherchant dans l'aide de l'Excel, on trouve ceci : "L'écart type est une mesure de la dispersion des valeurs par rapport à la moyenne".
- Si l'écart type est faible les classes sont très resserrées (corrélées), s'il est important elles sont plus dispersées (pas de forte corrélation).
- Afin de déduire la corrélation des classes dans un fichier ARFF, on compare l'écart type à la $\frac{1}{2}$ de la moyenne arithmétique

Si : Ecart-type$>\frac{1}{2}$ moyenne$\rightarrow$ Faible corrélation

Si : Ecart-type$<\frac{1}{2}$ moyenne$\rightarrow$Forte corrélation

4.3.1.5 *L'écart type d'un ensemble de fichier ARFF*

En se basant sur l'idée de l'exemple précédent, nous avons écrit un programme en java qui nous permet de calculer l'écart type de n'importe quel fichier ARFF.

Voici quelques fragments de code java sous eclipse

1. La saisie du fichier arff par l'utilisateur

```
//Saisir le fichier arff par le user
Scanner sc1 = new Scanner(System.in);
System.out.println("\n Veuillez saisir le chemin vers votre fichier Arff : ");
String path = sc1.nextLine();
```

2. La récupération du fichier arff

```
//Récupération du fichier ARFF
DataSource source = new DataSource(path);
Instances dataset = source.getDataSet();
```

3. Récupération du nom de la relation pour récupérer le nombre des attributs classes

```
//Récupérer le nom de la relation
String str = dataset.relationName();
```

4. Récupération de la taille de nom de la relation

```
//La taille du nom de la relation
int taile1 = str.length();
System.out.println("la taille du nom de la relation est "+taile1);
```

5. Récupération de nombre d'attributs classes

```
ArrayList al=new ArrayList();
boolean bln=false;
int b=0;
while ((b<taile1)&&(bln==false)){
    System.out.print(str.charAt(b));
    if ((str.charAt(b)=='-') && (str.charAt(b+1)=='C')&& (str.charAt(b+2)==' ')&& (bln==false)){
        bln=true;
        System.out.println();
        for(int j=b+3;j<taile1;j++){
            al.add(str.charAt(j));
        }
    }
    b++;
}
```

```
int t []=new int [al.size()];
int atclas=0;
int attrib=0;
char cara []=new char [al.size()];
String str2="";
int foo;

for(int k = 0; k < al.size(); k++)
{
if ( al.get(k)!="-"){
        cara[k]=(Character) al.get(k);
        // t : le nombre des attributs classes
        t [k]=Character.getNumericValue(cara[k]);
        System.out.print(t[k]);
        str2=str2+t[k];
}else {
    cara[k]=(Character) al.get(k);
        // t : le nombre des attributs classes
       t [k]=Character.getNumericValue(cara[k]);
       System.out.print(t[k]);
       str2=str2+t[k];
}
}
foo = Integer.parseInt(str2);
```

6. Une fois que le nombre d'attributs classes est récupéré, on doit vérifier son signe, positif ou négatif

```
if (foo<0){//Si le nombre est négatif
    System.out.println("ce nombre est négatif \n");
    atclas=0;
    atclas=-(foo+10);
    if(foo<-10){
        atclas=0;
        atclas=(foo+10);
        if(foo<-100){
            atclas=0;
            atclas=(foo+100);
            if(foo<-1000){
                atclas=0;
                atclas=(foo+1000);
                    if(foo<-10000){
                    atclas=0;
                    atclas=(foo+10000);
                        if(foo<-100000){
                        atclas=0;
                        atclas=(foo+100000);
                        }
                    }
            }
        }
    }
}
else if (foo>0){//Si le nombre est positif
    System.out.println("Ce nombre est positif");
    atclas=foo;
}
```

7. Après la vérification de signe, on doit récupérer :

Les classes qui se trouvent à la droite du dataset (Nombre d'attributs classes positif)

Les classes qui se trouvent à la gauche du dataset (nombre d'attributs classe négatif)

```
int nbatr=dataset.numAttributes();//Nombre total des attributs, interval droit
System.out.println("Le nombre total des attributs est : "+nbatr);
int interv=nbatr+atclas;//L'intreval gauche des attributs Class
int sous;//La soustraction
sous=nbatr-interv;//La soustraction
System.out.println("Voilà le nombre d'attrinuts dont on a besoin : "+atclas);
int diff1=nbatr + atclas;
int diff2=nbatr - atclas;
```

Afin de récupérer ces classes ;

8. On doit tout d'abord enlever les virgule de tous le dataset

```
//Faire un split pour enlever les virgules
tableau = dataset.get(z).toString().split(",");
```

Une fois que tout ça est fait, on passe au calcul de l'écart type

9. On calcule le nombre d'objets dans chaque classe

```
//Calculer le nombre des objets dans chaque classe
if (foo>0){
    int table4 []=new int [a];
    double table5 []=new double [a];
    for(int y = 0; y < a; y++){
            int nbr4=0;//Pour calculer le nombre d'objet
            //a c'est pour parcourrir toutes les lignes
            for(int k = 0; k < atclas; k++){ // nbatr : le nombre total des attributs
                //Pour calculer le nombre d'objets
                if (matrice[y][k].equals("1")){
                    nbr4++;
                }
        }
            table4[y]=nbr4;
            System.out.println("L'objet "+y+ " apparait "+ table4[y] + " fois dans toutes les classes "
            total=total+table4[y];
    }
    System.out.println(" \n L'effectif total de tous les objets est : "+total);
    System.out.println("le nombre d'objet est "+a);
```

10. On calcule la moyenne arithmétique (1 étape de l'écart type)

```
moy=(double)total/a;
System.out.println(" \n La moyenne arithmétique (X barre) est " +moy);
```

11. la 2 et la 3 étape de l'écart type

```
for(int y = 0; y < a; y++){
    table5[y]=(table4[y]-moy)*(table4[y]-moy);//2 étape de l'écart type
    som2=som2+table5[y];
}
//3 étape de l'ecart type
System.out.println("le somme de la 2 étape de l'écart type est : "+som2);
```

12. la 4 étape de l'écart type

```
somdiv=(double)som2/a;
System.out.println("La 4 étape en divisant la somme sur les objets totaux égale à : " +somdiv);
```

13. En faisant la racine carré de l'étape du résultat précédent, on aura le résultat de l'écart type :

```
//la racine carrée de la 4 étape
System.out.println("\n L'ecart type est :" +Math.sqrt(somdiv)+"\n");
```

Ci-dessous les résultats de quelques tests effectués sur un ensemble quelconque de fichiers ARFF *birds, CAL500, flags, mediamill, Music, scene, solar_flare* et *water-quality*.

Fichiers ARFF	**birds**	**CAL500**	**flags**	**mediamill**	**Music**	**scene**	**solar_flar**	**water_ quality**
Moy	1,0139	26,0438	3,3917	4,3755	1,8699	1,0739	0,1857	5,0726
$\frac{1}{2}$ **Moy**	0,5069	13,0219	1,6958	2,1877	0,9349	0,5369	0,0928	2,5363
Ecart type	1.1739	5.7426	1.1802	2.3337	0.6715	0.2632	0.4683	2.1808
Obser	Faible corrélation	Forte corrélation	Forte corrélation	Faible corrélation	Forte corrélation	Forte corrélation	Faible corrélation	Forte corrélation

TABLE 4.8 – Ecart type

Après avoir calculé l'écart type de différents fichiers arff, nous avons obtenu différents résultats qui ont une forte ou une faible corrélation en les comparant à $\frac{1}{2}$ de la moyenne.

Les résultats du tableau 4.8 montrenet que :

— Le fichier **birds.arff** relève une faible corrélation avec un taux= $1,1739$ qui est supérieur à $\frac{1}{2}$ de la moyenne ($> 0,5069$).

— Le fichier **CAL500** relève une forte corrélation avec un taux= $5,7426$ qui est inférieur à $\frac{1}{2}$ de la moyenne ($< 13,0219$).

— Le fichier **flags.arff** relève une forte corrélation avec un taux= $1,1802$ qui est inférieur à $\frac{1}{2}$ de la moyenne ($< 1,6958$).

— Le fichier **mediamill.arff** relève une faible corrélation avec un taux= $2,3337$ qui est supérieur à $\frac{1}{2}$ de la moyenne ($> 2,1877$).

— Le fichier **Music.arff** relève une forte corrélation avec un taux= $0,6715$ qui est inférieur à $\frac{1}{2}$ de la moyenne ($< 0,9394$).

— Le fichier **scene.arff** relève une forte corrélation avec un taux= $0,2632$ qui est inférieur à $\frac{1}{2}$ de la moyenne ($< 0,5369$).

— Le fichier **solar_flare.arff** relève une faible corrélation avec un taux= $0,4683$ qui est supérieur à $\frac{1}{2}$ de la moyenne ($> 0,0928$).

4.3.2 Introduction à Meka

L'apprentissage automatique est lié à l'intelligence artificielle(AI) et fournit une méthodologie et une technologie pour améliorer les applications du monde réel.

Il existe plusieurs techniques d'apprentissage automatique qui sont une partie essentielle d'un nombre croissant d'applications en science, ingénierie, systèmes d'information et de l'éducation, tels que la reconnaissance vocale, faciale, etc . . .

Plusieurs domaines émergents tels que l'exploration de données et l'exploration du Web utilisent des techniques d'apprentissage machine de base, y compris les arbres de décision, réseaux de neurones et l'apprentissage Bayésien. Weka est un ensemble d'algorithmes d'apprentissage machine pour les tâches d'exploration de données. Meka fournit une implémentation open source de méthodes d'apprentissage et d'évaluation multi-labels .

4.3.2.1 *Présentation de WEKA*

WEKA a été développé à l'Université de Waikato en Nouvelle-Zélande ; le nom signifie Waikato Environment for Knowledge Analysis. En dehors de l'université, le WEKA, prononcé pour rimer avec La Mecque, est un oiseau sans vol avec une nature curieuse trouvée seulement sur les îles de la Nouvelle-Zélande. [78]

WEKA est une collection d'algorithmes d'apprentissage automatique et d'outils de prétraitement de données qui inclut la plupart des algorithmes d'intelligence artificielle, dont les arbres de décision et les réseaux de neurones. Les algorithmes peuvent soit être directement appliquées à un ensemble de données ou appelés à partir de code Java. Weka est conçu pour nous permettre d'essayer rapidement les méthodes existantes sur de nouveaux jeux de données (Data-sets) de manière flexible. Il permet la préparation des données d'entrée, l'évaluation statistiquement des schémas d'apprentissage, et la visualisation des données d'entrée et le résultat de l'apprentissage. En plus d'une grande variété d'algorithmes d'apprentissage, il comprend une vaste gamme d'outils de prétraitement. Cette boîte à outils diversifiée et complète est accessible via une interface commune afin que ses utilisateurs puissent comparer différentes méthodes et identifier celles qui sont les plus appropriées pour le problème en question. [79]

Weka est un logiciel open source publié sous GNU General Public License et son système est écrit en Java. Il fonctionne sur presque toutes les plateformes et a été testé sous Linux, Windows et Macintosh.

Il se compose principalement : [79]

— 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.

— De classes permettant de visualiser les résultats.

On peut l'utiliser à deux (02) niveaux : [79]

— 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.

Presentation de l'interface graphique : [78][80] Le logiciel Weka peut être utilisé à travers deux modes principaux. Le premier mode est une interface de ligne de commande **(Simple CLI)** (Figure 4.4) . Le deuxième mode est une interface graphique **Explorer** qui donne accès à toutes ses installations en sélectionnant le menu et en remplissant le formulaire comme le montre la (Figure 4.5).

Pour commencer, il y a six panneaux différents, sélectionnés par les onglets en haut, correspondant aux diverses tâches d'exploration de données que WEKA supporte. D'autres panneaux peuvent être disponibles en installant des paquets appropriés :

— **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.

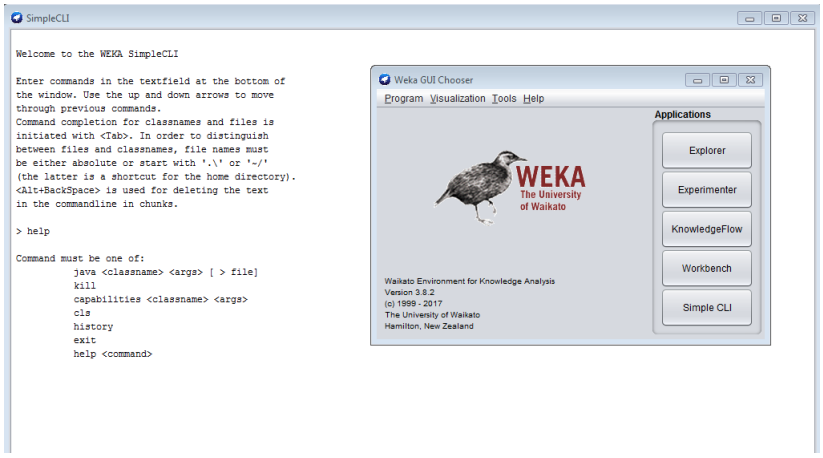


FIGURE 4.4 – Interface de ligne de commande Weka

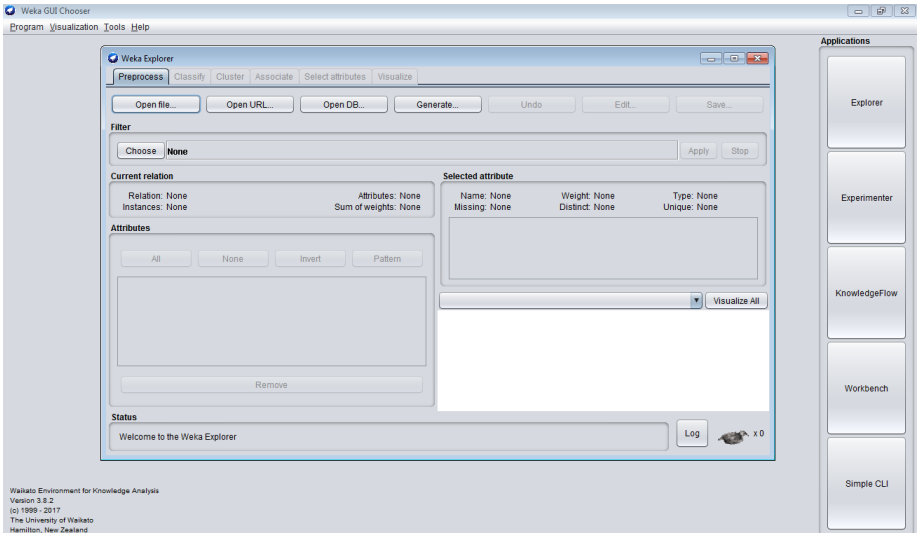


FIGURE 4.5 – Interface Graphique Weka

Le traitement de données : [79] [80] Les données sont de plus en plus volumineuses dans l'industrie et le traitement de données constitue un véritable défi pour les systèmes informatiques. Le logiciel Weka se présente comme une solution efficace pour le traitement de données même celles de grandes envergures appelées « Big Data ». Pour être traitées, les données doivent être entrées sous les formats ARFF (Attribute Relation File Format), CSV (Comma-separated values), Binaire, BDD(Base De Données), SQL(Structured Query Language) et URL(Uniform Resource Locator).

Le format le plus utilisé est le format **ARFF** (nous allons aborder plus en détails ce point dans MEKA) dont la structure des données se compose :

— du nom de la relation : **@relation** 'NomDeLaRelation'

— de la liste des noms d'attributs **@attribute** NomAttribut Type

— de la liste des instances **@data** : chaque ligne représente une description de la liste des valeurs de chacun de ses attributs.

Les attributs peuvent être des réels (réal), des chaînes de caractères (string) et des dates (date).

On doit respecter l'ordre des attributs en saisissant les instances.

Exemple :

```
@relation 'Testmemoire'

@attribute Titre String
@attribute Promoteur {Mr1,Mr2,Mr3}
@attribute Annee numeric
@attribute Note real

% Ceci est un commentaire

@data
memoire1,Mr1,2018,17.75
memoire2,Mr2,2016,16
memoire3,Mr3,2017,15.5
memoire4,Mr2,2010,18
memoire5,Mr1,2013,16.5
```

FIGURE 4.6 – Le fichier ARFF sous weka

4.3.2.2 *Presentation de meka*

Dans le problème de plusieurs labels, une instance de données peut être associée à plusieurs étiquettes. Ceci est par opposition à la tâche traditionnelle de classification en un seul label (c'est-à-dire, multi-classe, ou binaire) où chaque instance est seulement associée à une seule étiquette de classe . [81]

Il existe maintenant un grand nombre et une grande variété de méthodes pour ce type de classification.

MEKA est un logiciel open source basé sur WEKA, il fournit un soutien (des interfaces) pour le développement, le fonctionnement et l'évaluation des classifieurs Multi-labels (ce que WEKA ne fait pas). MEKA est développé en Java ; par conséquent, pour l'utiliser, la première condition à satisfaire est l'installation de **JRE** (Java Runtime Environment) dans le système. Ensuite, la version la plus récente de MEKA peut être téléchargée depuis ***https ://adams.cms.waik-ato.ac.nz/snapshots/meka***. Le processus d'installation consiste simplement à extraire les fichiers du fichier compressé dans un dossier. Le logiciel comprend deux scripts, un pour Windows ***(run.bat)*** et un autre pour les systèmes UNIX ***(run.sh***), destinés à faciliter le lancement du logiciel. Le lancement de MEKA par le script approprié ouvrira la fenêtre principale du programme (Figgure 4.7). Les options de l'onglet **Tools** exécutent les outils MEKA essentiels (Data Viewer, Explorer et Experimenter).

FIGURE 4.7 – Meka Interface

Le format de jeu de données de Meka : [81] [83] [86] Le fichier ARFF a été développé par l'Université de Waikato pour une utilisation avec le logiciel WEKA et/ou MEKA(Dans notre cas c'est MEKA). Le format ARFF est un moyen standard de représenter l'ensemble de données qui consiste en des instances indépendantes, non ordonnées et n'im-

plique pas de relation entre les instances. Le fichier ARFF comporte deux sections. La première section est la section En-tête, suivie de la section Informations sur les données. La section d'en-tête aura le nom du fichier et les attributs. Le nom du fichier commence par @relation et le nom des attributs commence par @attribute. Il existe quatre types d'attributs : ils sont numériques, chaîne, date et attributs nominaux. L'attribut numérique prendra une valeur numérique et l'attribut nominal aura un ensemble de valeurs qu'ils peuvent prendre. La section de données commence par @data suivi de quelques valeurs. Chaque ligne représente une instance.

Le format de fichier sous MEKA est également basé sur ARFF qui utilise plusieurs attributs dont le nombre d'attributs classes est spécifié avec -C ou –c et est déclaré dans l'en-tête après le nom de la relation .

Dans le carde de notre travail nous allons utiliser des fichiers ARFF avec deux façons différentes pour la déclarations des attributs classes

1. Si le nombre d'attributs classes est positif alors ils seront presentés dans de début de chaque dataset

```
@relation 'TestPositif : -C 5'

@attribute amazed-suprised {0,1}
@attribute happy-pleased {0,1}
@attribute relaxing-clam {0,1}
@attribute quiet-still {0,1}
@attribute sad-lonely {0,1}
@attribute nom {bellal,aitkaid,bellil}
@attribute prenom {lounis,yassmine,fateh,malik}
@data
0,0,1,0,1,bellal,lounis
1,0,1,0,1,aitkaid,yassmine
1,1,1,1,1,bellil,fateh
1,0,1,0,0,bellil,malik
0,0,0,1,0,bel,yassmine
0,1,0,0,0,aitkaid,lounis
1,1,0,0,0,bellal,fateh
0,0,0,0,0,bellal,malik
1,1,0,0,0,aitkaid,fateh
0,0,1,1,1,aitkaid,malik
0,1,1,0,0,bellil,yassmine
```

FIGURE 4.8 – Attributs classe présentés au début du Data-set

2. Si le nombre d'attributs classes est négatif alors ils seront présentes dans la fin de chaque dataset

```
@relation 'Testfinal : -C -5'

@attribute nom {bellal,aitkaid,bellil}
@attribute prenom {lounis,yassmine,fateh,malik}
@attribute amazed-suprised {0,1}
@attribute happy-pleased {0,1}
@attribute relaxing-clam {0,1}
@attribute quiet-still {0,1}
@attribute sad-lonely {0,1}

@data
bellal,lounis,0,0,1,0,1
aitkaid,yassmine,1,0,1,0,1
bellil,fateh,1,1,1,1,1
bellil,malik,1,0,1,0,0
bel,yassmine,0,0,0,1,0
aitkaid,lounis,0,1,0,0,0
bellal,fateh,1,1,0,0,0
bellal,malik,0,0,0,0,0
aitkaid,fateh,1,1,0,0,0
aitkaid,malik,0,0,1,1,1
bellil,yassmine,0,1,1,0,0
```

FIGURE 4.9 – Attributs Classe présentes à la fin du Data-set

Remarque :

Dans l'ensemble des attributs : *amazed-suprised, happy-pleased,relaxing-clam,quiet-still,sad-lonely* ; **1** signifi la presence de l'attribut (Objet) et **0** signifie l'absence de l'attribut(Objet)

Utilisation de MEKA :

La ligne de commande : [81] À l'exception de l'utilisation différente de l'option -c, de nombreuses options de la ligne de commande de Weka pour l'évaluation fonctionnent également de la même manière dans Meka. On peut obtenir une liste en exécutant n'importe quel classifieur avec l'option –h.

Exemple :

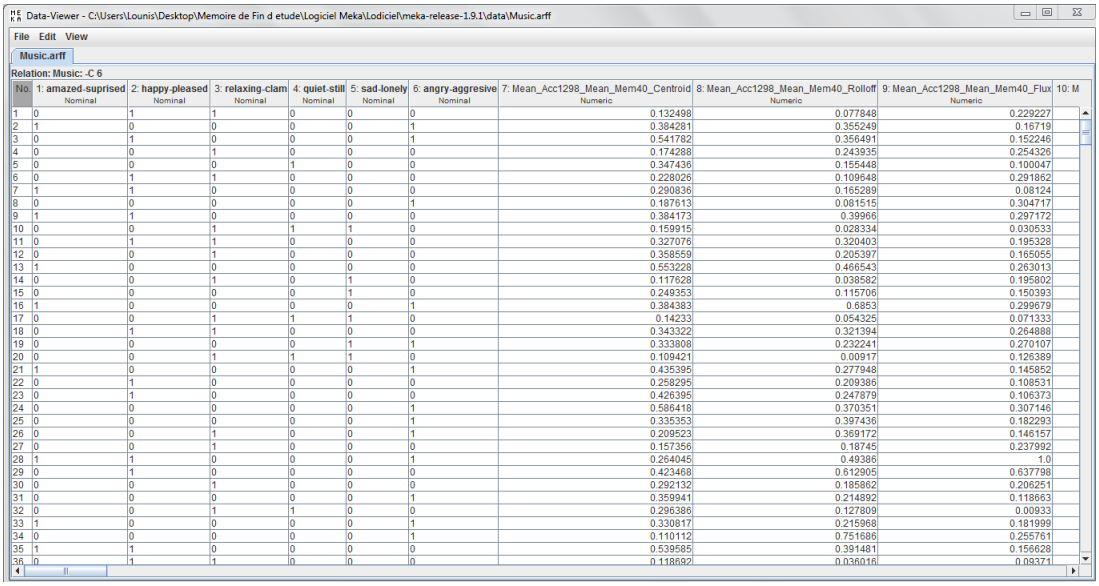
Exécutant java meka.classifiers.multilabel.BR -h sur la ligne de commande.

L'interface graphique (GUI) : [81] L'interface graphique de MEKA est pleinement fonctionnelle que la ligne de commande. En fait, il a des fonctionnalités de visualisation supplémentaires.

Une fois la GUI est ouverte on aura à choisir entre '**Explorer**' et '**Experimenter**', comme on peut les selectionner en cliquant sur l'onglet **Tools** mais ici on aura un autre choix de plus qui est **Data viewer.**

— **Data viewer (Le visualiseur ARFF)** [86]

MEKA inclut une visionneuse capable d'ouvrir n'importe quel fichier ARFF, accessible via l'option **Tools** → **Data viewer** dans la fenêtre principale, énumérant ses attributs, y compris les étiquettes, et les valeurs qui leur sont assignées. Une fois qu'un jeu de données a été chargé, son contenu est affiché dans une grille similaire à celle de la figure 4.10.



No.	1: amazed-suprised Nominal	2: happy-pleased Nominal	3: relaxing-clam Nominal	4: quiet-still Nominal	5: sad-lonely Nominal	6: angry-aggresive Nominal	7: Mean_Acc1298_Mean_Mem40_Centroid Numeric	8: Mean_Acc1298_Mean_Mem40_Rolloff Numeric	9: Mean_Acc1298_Mean_Mem40_Flux Numeric	10: M
1	0	1	1	0	0	0	0.132498	0.077848	0.229227	
2	1	0	0	0	0	1	0.384281	0.355249	0.16719	
3	0	1	0	0	0	1	0.541782	0.356491	0.152246	
4	0	0	1	0	0	0	0.174288	0.243935	0.254326	
5	0	0	0	1	0	0	0.347436	0.155448	0.100047	
6	0	1	1	0	0	0	0.228026	0.109648	0.291862	
7	1	1	0	0	0	0	0.290836	0.165289	0.08124	
8	0	0	0	0	0	1	0.187613	0.081515	0.304717	
9	1	1	0	0	0	0	0.384173	0.39966	0.297172	
10	0	0	1	1	1	0	0.159915	0.028334	0.030533	
11	0	1	1	0	0	0	0.327076	0.320403	0.195328	
12	0	0	1	0	0	0	0.358559	0.205397	0.165055	
13	1	0	0	0	0	0	0.553228	0.466543	0.263013	
14	0	0	1	0	1	0	0.117628	0.038582	0.195802	
15	0	0	0	0	1	0	0.249353	0.115706	0.150393	
16	1	0	0	0	0	1	0.384383	0.6853	0.299679	
17	0	0	1	1	1	0	0.14233	0.054325	0.071333	
18	0	1	1	0	0	0	0.343322	0.321394	0.264888	
19	0	0	0	0	1	1	0.333808	0.232241	0.270107	
20	0	0	1	1	1	0	0.109421	0.00917	0.126389	
21	1	0	0	0	0	1	0.435395	0.277948	0.145852	
22	0	1	0	0	0	0	0.258295	0.209386	0.108531	
23	0	1	0	0	0	0	0.426395	0.247879	0.106373	
24	0	0	0	0	0	1	0.586418	0.370351	0.307146	
25	0	0	0	0	0	1	0.335353	0.397436	0.182293	
26	0	0	1	0	0	1	0.209523	0.369172	0.146157	
27	0	0	1	0	0	0	0.157356	0.18745	0.237992	
28	1	1	0	0	0	1	0.264045	0.49386	1.0	
29	0	1	0	0	0	0	0.423468	0.612905	0.637798	
30	0	0	1	0	0	0	0.292132	0.185862	0.206251	
31	0	0	0	0	0	1	0.359941	0.214892	0.118663	
32	0	0	1	1	0	0	0.296386	0.127809	0.00933	
33	1	0	0	0	0	1	0.330817	0.215968	0.181999	
34	0	0	0	0	0	1	0.110112	0.751686	0.255761	
35	1	1	0	0	0	0	0.539585	0.391481	0.156628	
36	0	1	1	0	0	0	0.118692	0.036016	0.09371	

FIGURE 4.10 – La visionneuse MEKA ARFF permet d'afficher et d'éditer tout contenu de jeu de données ARFF

L'option **Properties** du menu **File** affiche le nombre d'instances, d'attributs et de classes(Figure 4.11).

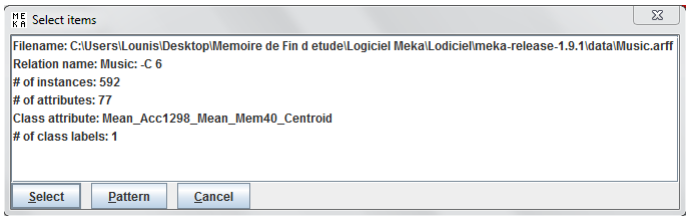


FIGURE 4.11 – Les options de dataset

En fait, cet outil est également un éditeur, car les valeurs peuvent être modifiées, les instances et les attributs peuvent être supprimés et l'ordre des échantillons de données peut être modifié. Tout changement sera perdu à moins que les données en mémoire sont enregistrées, généralement avec un autre nom de fichier.

— **Explorer** [86]

L'explorateur MEKA est accessible directement en cliquant sur **Explorer** de la fenetre principale ou en sélectionnant l'onglet **Tools→ Explorer** dans le menu. Cet outil vise à tester de manière interactive les algorithmes de prétraitement et de classification. On ouvre un fichier ARFF avec **Open** dans le menu **File** et une fois qu'il est chargé, la page Pré-traitement affiche deux listes contenant tous les attributs de l'ensemble de données (Figure 4.12). En haut de la liste de gauche, il y a un résumé avec le nombre d'instances et d'attributs. Si le jeu de données a été chargé à partir d'un fichier au format MEKA, le programme détectera automatiquement les attributs qui sont des attributs classe. Ceux-ci seront mis en évidence en gras et seront toujours au début dans les deux listes. Le chargement de jeu de données à partir d'autres formats de fichiers ARFF est autorisé, mais le programme ne sera pas en mesure d'identifier les étiquettes(attributs classes). L'utilisateur doit les marquer dans la liste de droite, puis cliquer sur **Use class attributs**. En sélectionnant n'importe quel attribut dans la liste de gauche, un résumé de son domaine sera affiché sous la bonne liste.

Comme on peut enregistrer le fichier en utilisant **Save** dans le menu **File.**

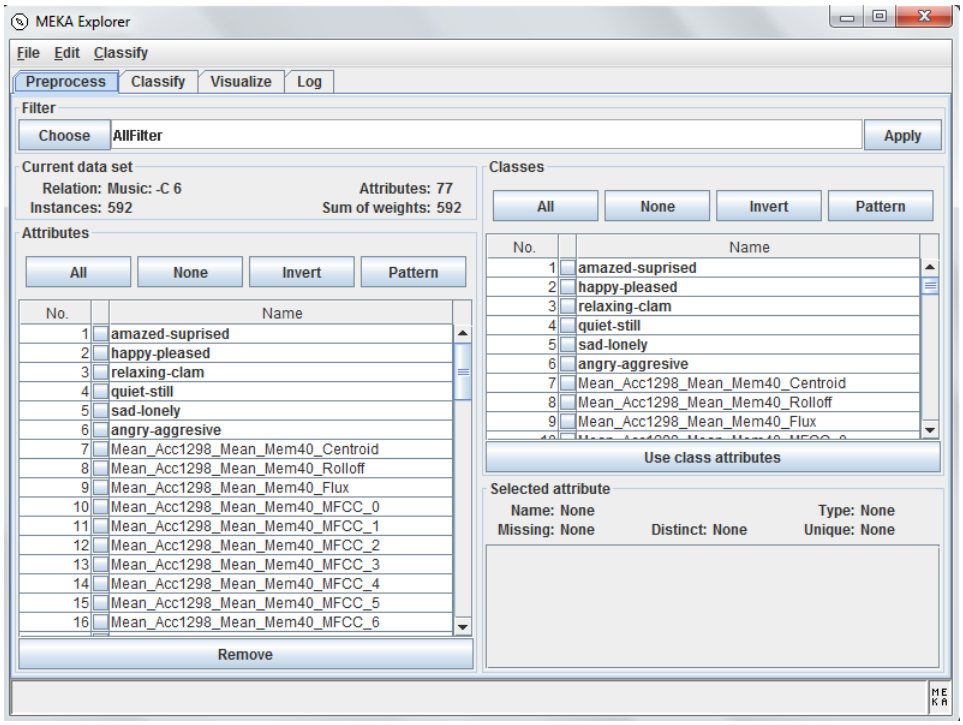


FIGURE 4.12 – Interface graphique de MEKA après avoir chargé le Data Set Music.

Traitement d'un fichier ARFF :[81]

Dans l'interface graphique de **l'Explorer** on voit un certain nombre d'onglets : **Preprocess**, **Classify**, **Visualize** et **Log**. (Figure 4.13)

Les ensembles de données fournis avec Meka sont déjà fournis avec l'indicateur -C spécifié correctement, on n'a donc pas besoin de définir ces informations.

Afin de comprendre la manipulation des fichiers arff sous l'**Explorer** meka, nous allons suivre les démarches suivantes :

Une fois que le fichier ARFF est chargé :

1. Cliquez sur l'onglet **Classify**.

2. On choisit un classifieur multi-labels.

3. On Clique sur l'étiquette à droite de ce bouton et on définit des options spécifiques pour le classifieur. Dans la plupart des cas, ces options impliquent également la définition d'un classifieur de base en une seule étiquette Weka. Pour les méta-classifieurs de Meka, on doit d'abord choisir un classifieur de base multi-labels, puis un classifieur de base à étiquette unique (Weka) pour ce classifieur. (Figure 4.13), en utilisant un ensemble BaggingML de CC avec SMO comme classifieur de base à étiquette unique.

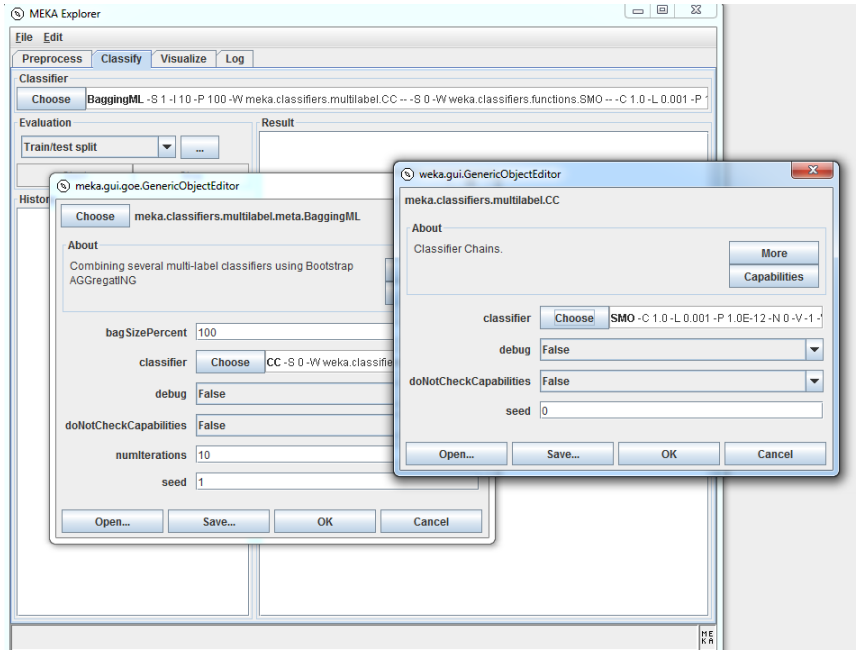


FIGURE 4.13 – L'interface graphique de MEKA, mise en Bagging de chaînes de classificateur avec SMO.

4. Dans le panneau d'évaluation, on configure le type d'évaluation qu'on veut effectuer, et certaines des options données. Par exemple, une division train / test 55/45, avec un étalonnage de seuil PCutL, comme spécifié dans la Figure 4.14.

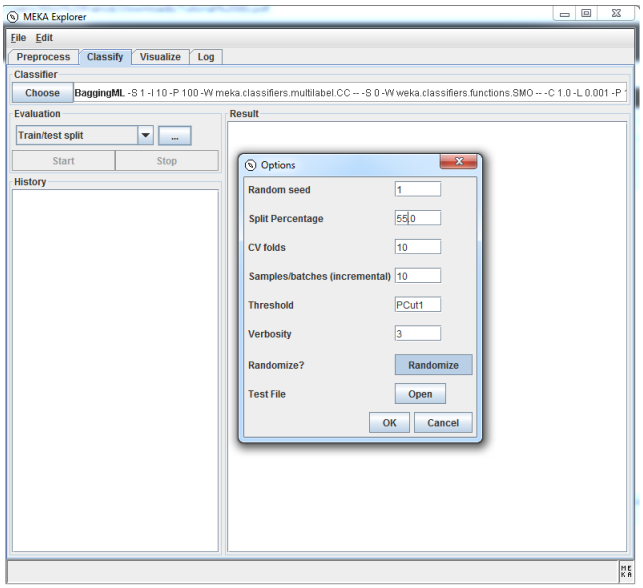


FIGURE 4.14 – L'interface graphique de MEKA, mise en place une division train / test.

5. Lorsque on clique sur **Start**, l'expérience sera exécutée.

Une fois terminé, le résultat apparaîtra dans le panneau **History**, (Figure 4.15).

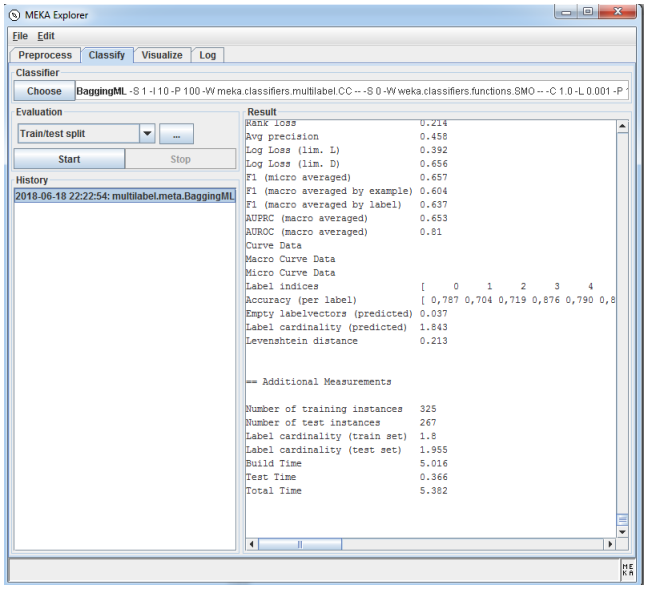


FIGURE 4.15 – L'interface graphique de MEKA,résultat.

6. (Facultatif) On affiche les courbes **Precision-Recall** en cliquant avec le bouton droit sur les résultats et en sélectionnant **Show Precision-Recall**. (Figure 4.16)

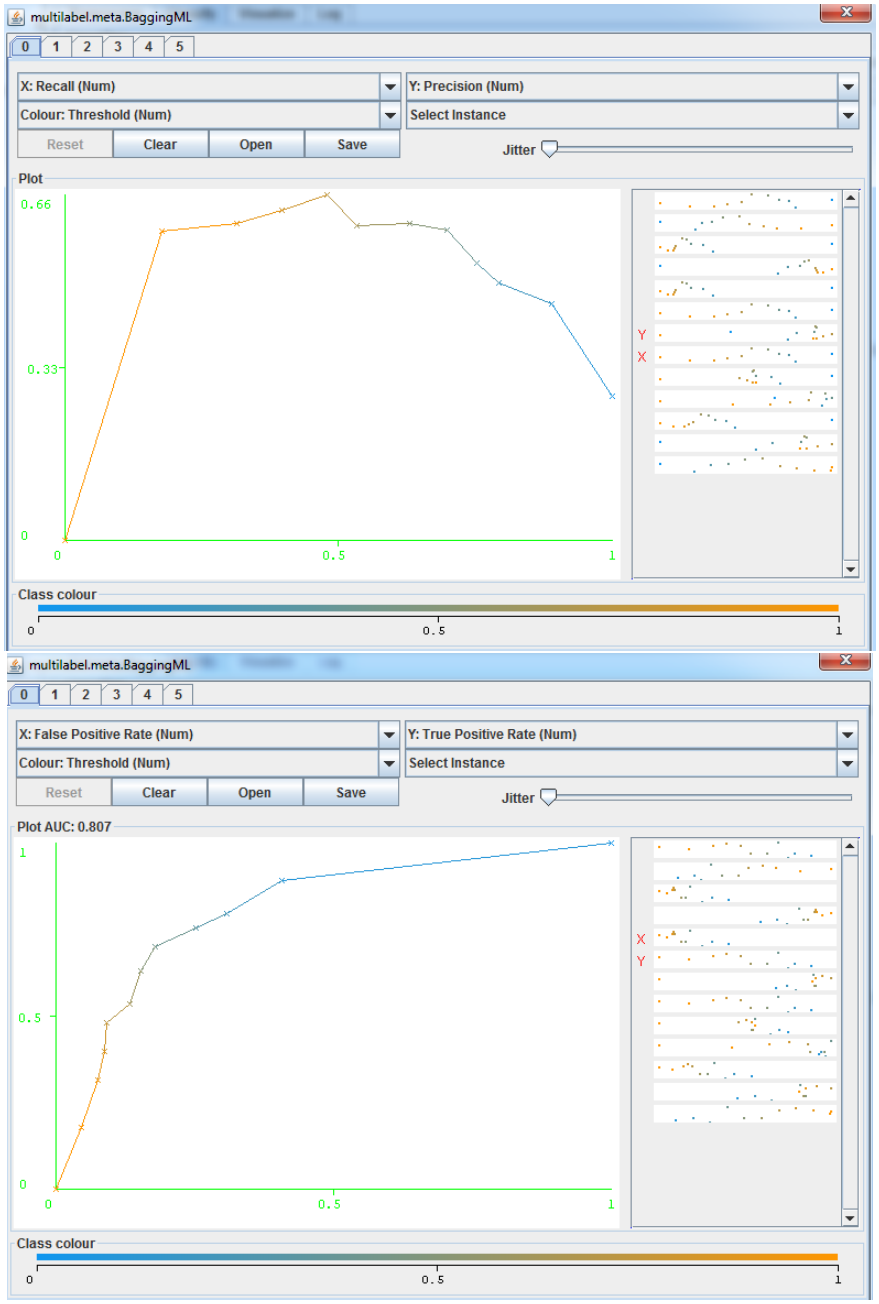


FIGURE 4.16 – L'interface graphique de MEKA, visualisation des performances incrémentielles dans le temps (à gauche) et une courbe ROC (à droite).

7. (Facultatif) Si on a choisi un classifieur MultiLabelDrawable (tel que BR, CC, LP ou tout dérivé) et un classifieur de base Drawable tel que J48, on peut faire un clic droit sur les résultats et sélectionner **Show graphs**. (Figure 4.17)

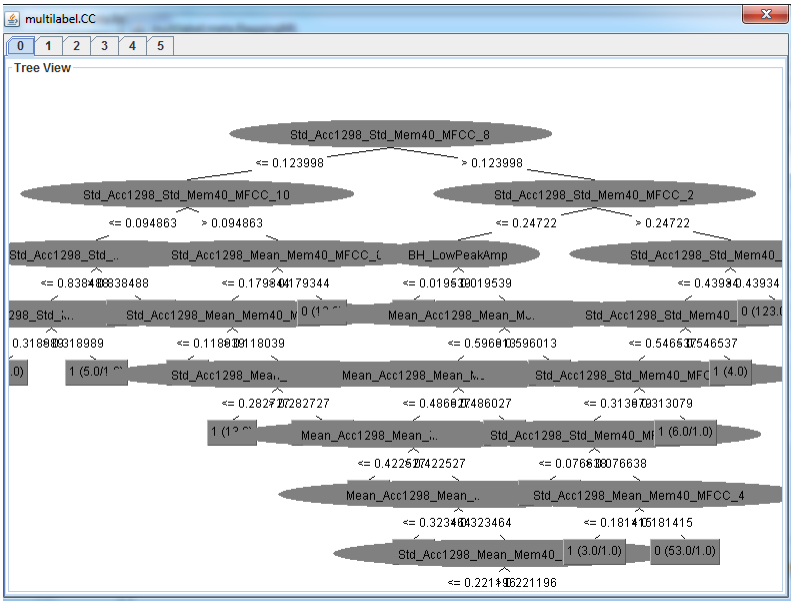


FIGURE 4.17 – L'interface graphique de MEKA, l'affichage d'un modèle d'arbre de décision de CC. Duplication du 9

8. (Facultatif) on clique sur l'onglet **Visualize** pour inspecter l'ensemble de données (Figure 4.18).

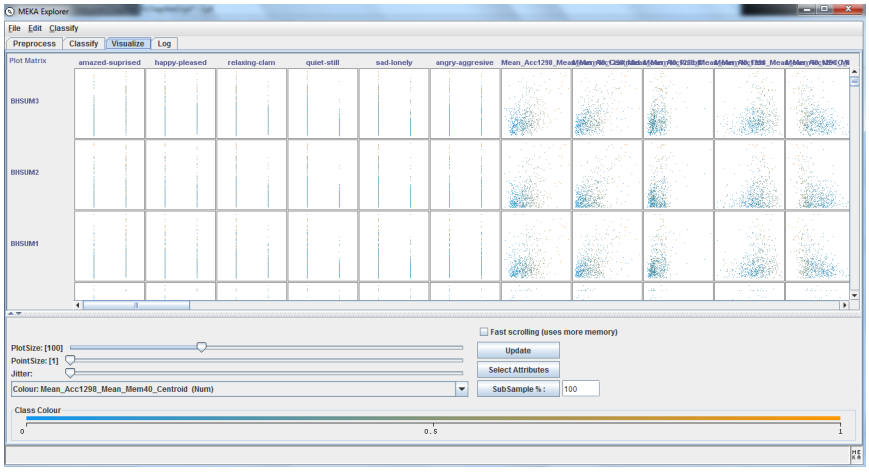


FIGURE 4.18 – L'interface graphique de l'onglet Visualize

9. (Facultatif) Pour qu'on souvienne plus tard de ce qu'on a fait, on enregistre le journal sous l'onglet **Log** (Figure 4.19).

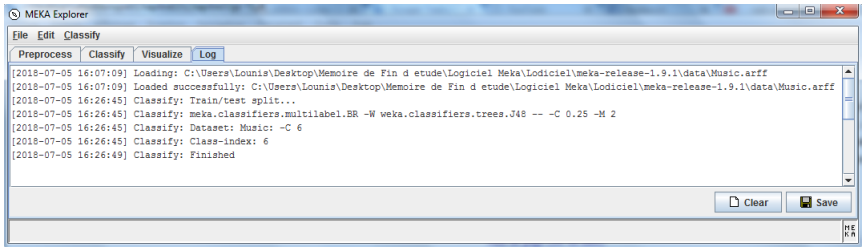


FIGURE 4.19 – L'interface graphique de l'onglet Log

— **L'Experimenter : [81]**

Comme dans l'explorer on voit un certain nombre d'onglets. (Figure 4.20)

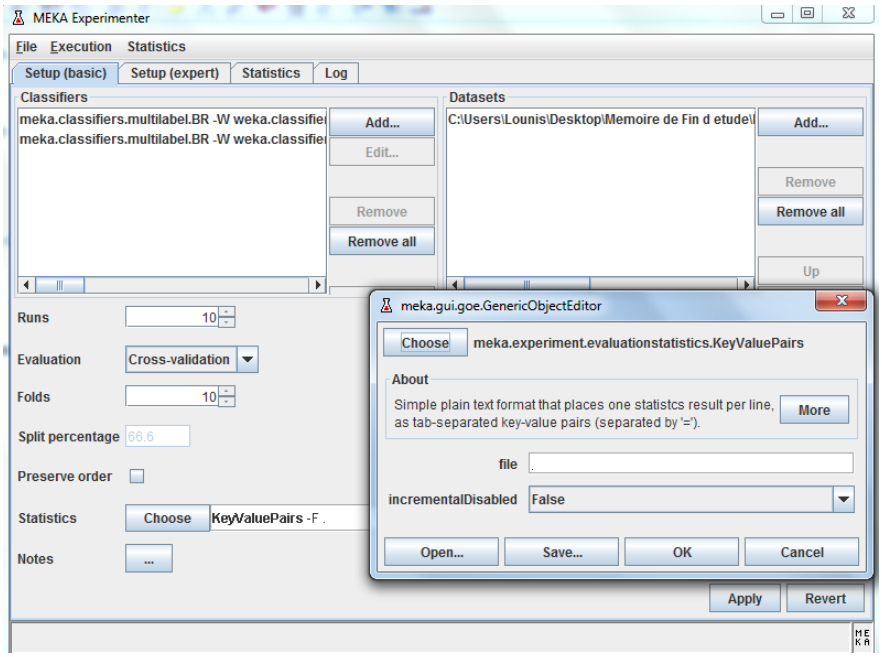


FIGURE 4.20 – L'Exprimenter de MEKA.

Afin de comprendre la manipulation des fichiers arff sous l'**Experimenter** meka, nous allons suivre les démarches suivantes :

1. On sélectionne **File** → **New** → **Default Experiment**.
2. On clique sur **Add** (dans le volet de gauche) pour ajouter une configuration de classifieur.
3. On répète le processus d'ajout d'un classifieur.
4. On clique sur **Add** (dans le volet de droite) pour ajouter une configuration de classifieur.
5. On configure le nombre d'exécutions et de replis pour **cross-validation.**
6. On clique sur l'étiquette **KeyValuePairs**, puis sur les étiquettes pour sélectionner un fichier pour enregistrer l'expérience.
7. On clique sur **OK**, puis sur **Apply** dans l'onglet principal.
8. On sélectionne **Execution** → **Start** dans le menu.
9. L'icône MEKA s'anime et on verra **Running** dans la barre des tâches.
10. Une fois terminé, on clique sur l'onglet **Statistics**.
11. On affiche **Aggregated subtab**.
12. On clique sur **Statistics** → **Save as** (**aggregated**) pour enregistrer cette table en tant que fichier **tsv(T**abulation **S**eparated **V**alues) ou **tex**.
13. On affiche le sous-onglet **Mesurement**.
14. On sélectionne une mesure qui nous intéresse particulièrement.
15. Cliquez sur **Statistics** → **Save as** (**measurement**) pour enregistrer cette table sous forme de fichier **tsv** ou **tex**.

Quelque méthodes disponible dans MEKA : [85] Les méthodes (Algorithmes) données ci-dessous sont seulement un résumé avec quelques exemples en ligne de commande.

— *M* indique une méthode meta, peut être utilisé avec n'importe quel autre classificateur Meka. Seuls les exemples sont donnés ici.

Classe		**Nom**	**Exemples**
BR		BENARY RELEVANCE	java meka.classifiers. multilabel.BR -t data.arff -W weka.classifiers. functions.SMO
CC		Classifieurs chains	java meka.classifiers.multilabel. CC -t data.arff -W weka.classifiers. functions.SMO
Meta. BaggingML	*M*	Ensembles of classieurs chains (ECC)	java meka.classifiers. multilabel.meta. BaggingML -I 10 -P 100 -t data.arff -W meka.classifiers. multilabel.CC - -W weka.classifiers. functions.SMO
LC		Label Combination / Label Powerset	java meka.classifiers.multilabel.LC -t data.arff -W weka.classifiers.functions.SMO
PS		Pruned Sets (Pruned Label Powerset)	java meka.classifiers.multilabel.PS -P 1 -N 1 -t data.arff -W weka.classifiers.functions.SMO

TABLE 4.9 – Quelques méthodes sous MEKA

4.3.2.3 *Tests d'application sous Meka*

Nous avons testé différents fichiers ARFF (**birds, Music, Scene et Solar_flare**) en leurs appliquant un ensemble d'algorithme Multi-Label ; trois (03) algorithmes qui tiennent compte des corrélations (**Classifier Chain (CC), Laber Powerset (LP), Pruned Set (PS)**) et un (01) algorithme qui ne tient pas compte des corrélations (**Binary Relevance (BR)**).

Afin de réaliser chaque test sous MEKA avec tous les algorithmes présentés précédemment, nous devons toujours spécifier un classifier de base dans WEKA, qui est dans notre cas, Séquential Minimal Optimization (**SMO**).

Dans ce qui suit nous allons nous intéresser aux résultats des mesures suivantes : **ACCURACY, EXACT MATCH** et **HAMMING LOSS**.

Et comme nous l'avons déjà vu dans le deuxième chapitre ; Toutes les mesures sont définies dans l'intervalle [0..1] et leurs plus grandes valeurs indiquent les meilleures performances (Forte corrélations) sauf que pour le Hamming Loss où la plus petite valeur indique la meilleure performance (Forte corrélations).

4.3.2.4 *Execution d'un fichier .ARFF sous meka*

1. Téléchargement du fichier ARFF : (birds.arff)

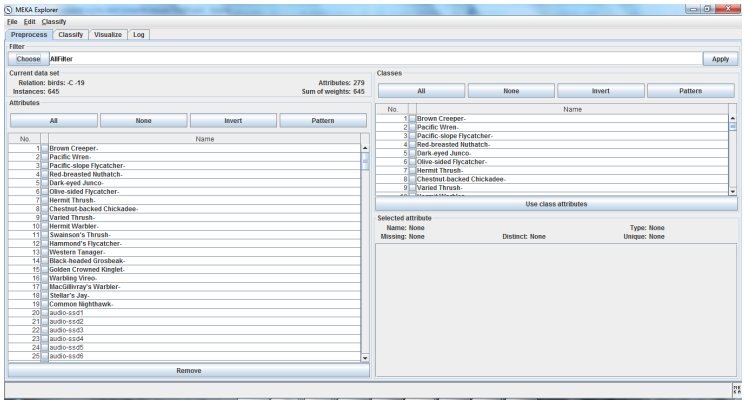


FIGURE 4.21 – Téléchargement du fichier birds.arff

2. Application d'algorithme Meka : Binary Relevance (BR) (Par exemple)
3. Application de Classifier de base Weka : (SMO).

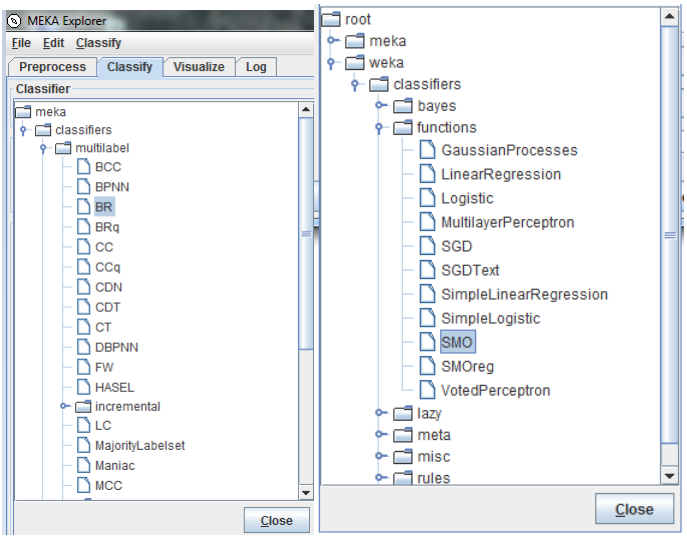


FIGURE 4.22 – Application d'algorithme Meka : Binary Relevance (BR) (A gauche),3. Application de Classifier de base Weka : Sequential Minimal Optimization (SMO).(A droite)

4. Résultats du test

On clique sur START pour lancer le test, et en attendant un moment (peut-être des heures), et cela dépendera de la taille du fichierr ARFF, on aura un ensemble de résultats :

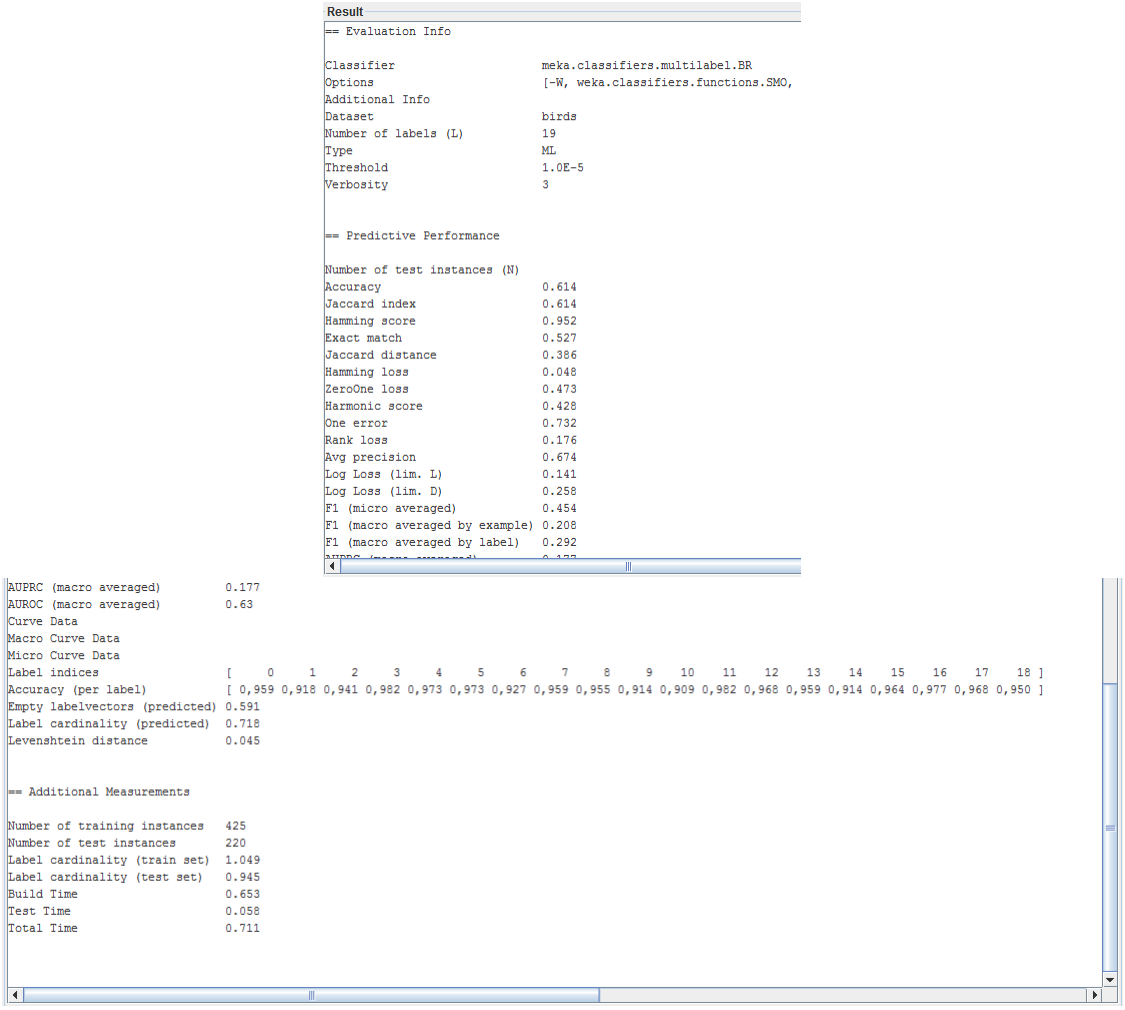


FIGURE 4.23 – Résultats du test

Dans le tableau ci-dessous on a illustré l'ensemble des résultats obtenus après l'execution du fichier birds.arff sous meka

Mesures	Classifieurs	Tient compte des corrélations			Ne tient pas compte des corrélations
	Fichiers	**CC**	**LP**	**PS**	**BR**
Accuracy	birds	0.628	0.635	0.635	*0.614*
Exact Match		0.532	0.541	0.541	*0.527*
Hamming loss		0.045	0.047	0.047	*0.048*

TABLE 4.10 – Résulats de test sur le fichier birds.arff

En comparant les résultats obtenus des classifieurs qui tiennent compte des corrélation en ceux qui ne tiennent pas compte des corrélation, appliqués sur le fichiers birds.arff, on remarque que ce fichier est bien structuré c'est-à-dire qu'il y a une forte corrélation entre ses classes.

Mesures	Classifieurs	Tient compte des corrélations			Ne tient pas compte des corrélations
	Fichiers	**CC**	**LP**	**PS**	**BR**
Accuracy	Music	0.55	0.591	0.591	*0.542*
Exact Match		*0.277*	*0.351*	*0.351*	0.282
Hamming loss		0.211	0.2	0.2	*0.196*
Accuracy	scene	0.696	0.734	0.734	*0.586*
Exact Match		0.656	0.695	0.695	*0.509*
Hamming loss		0.103	0.09	0.09	*0.107*
Accuracy	solar_flare	Pas de résultat			
Exact Match		0.782	0.782	0.782	*0.791*
Hamming loss		0.094	0.091	0.091	*0.088*

TABLE 4.11 – Autres tests

Autres tests Dans le tableau ci-dessus on a appliqué les mêmes classifieurs sur un ensembles de fichier arff et on a déduit differents résultats que nous allons expliquer ci-dessous :

De la même manière , nous comparons les résultats obtenus des classifieurs qui tiennent compte des corrélations en ceux qui ne tiennent pas compte des corrélations, appliqués sur les fichiers Music, scene et solar-flare, nous remarquons que :

Les deux fichiers Music et scene sont bien structurés c'est-à-dire qu'il y a une forte corrélation entre leurs classes.

Par contre le fichier solar-falre n'est pas structuré c'est-à-dire que ses classes ne sont pas corrélées

4.4 Conclusion

Dans ce chapitre nous avons présenté deux contributions, la première contribution est la classification multi-label hiérarchique et le treillis de Galois qui permet de construire des classifieurs permettant un regroupement maximal d'objets qui ont des propriétés en commun. La deuxième contribution est la mesure de similarité dont nous avons proposé une mesure qui permet de calculer la dispersion d'un ensemble d'objets qui est l'écart type et ainsi des résultats du code java sous eclipce.

Nous avons présenteé aussi le logiciel meka et un ensemble des résultats obtenus en appliquants différents classifieurs sur un ensemble de fichiers arff.

CONCLUSION GENERALE

1. Synthèse

Le domaine de la classification a particulièrement progressé ces dernières années grâce aux techniques de l'apprentissage automatique. Mais la classification multi-label hiérarchique n'a pas eu une grande attention, et pourtant on retrouve ce problème dans beaucoup d'applications importantes. Donc il est important de progresser dans ce domaine afin de proposer des solutions performantes pour améliorer cette classifications

A travers ce mémoire, nous avons présenté d'une façon complète les différentes notions nécessaires pour la compréhension de notre sujet. Nous avons organisé le mémoire en quatre chapitres, chacun traitant une partie de l'objectif global. Le dernier chapitre résume les deux (02) contributions qui répondent à notre thème. Les différents résultats obtenus ont montré l'intérêt des deux (02) contributions adoptées. Aussi nous avons montré l'intérêt des résultats obtenus sous meka.

Les deux contributions proposées ont tellement d'influence sur la classification multi-label hiérarchique que nous avons pensés à proposer d'autres solutions plus pertinenetes.

2. Perspective

Modification du MEKA en se basant sur l'injection des règles d'association pour corriger les erreurs des classifieurs, cela nous mène à faire une extention au logiciel.

Bibliographie

[1] https ://www.hpe.com/fr/fr/what-is/machine-learning.html

[2] https ://fxjollois.github.io/Jollois-These.pdf

[3] https ://lipn.univ-paris13.fr/˜bennani/THESES/These Rogovschi.pdf

[4] https ://www.supinfo.com/articles/single/6041machinelearningintroduction-apprentissage-automatique

[5] https ://docs.microsoft.com/frfr/azure/machinelearning/studio/algorithm-choice#algorithm-notes

[6] http ://www.grappa.univlille3.fr/˜torre/Enseignement/Cours/Apprentissage-Automatique/

[7] IFT3390/6390, Fondements de l'apprentissage machineProfesseur :Pascal-Vincent Cinqui_eme b cours : Rrégression multiple et classiffieur multiclasse.Rappels de proba. Classiffieur de Bayes. Classi_eur de Bayes Naiff,http ://www.iro.umontreal.ca/˜vincentp/ift3390

[8] Global Journal of Computer Science and Technology Interdisciplinary Volume 12 Issue 11 Version 1.0 Year 2012 Type : Double Blind Peer Reviewed International Research Journal Publisher : Global Journals Inc. (USA) Online ISSN : 0975-4172 & Print ISSN : 0975-4350 Multiclass Classi_cation and Support Vector Machine By Yashima Ahuja & Sumit Kumar Yadav Lovely Professional University, Jalandhar (Punjab) India

[9] http ://scikit-learn.org/stable/modules/multiclass.html

[10] Survey on Multiclass Classi_cation Methods Mohamed Aly malaa@caltech.edu November 2005

[11] Classiffieurs SVM et R_eseaux de Neurones

[12] Construction d'arbres De décision.

[13] Arbres de d´ d_ecision C_ecile Capponi cecile.capponi@lif.univmrs.fr Université Aix-Marseille M2 MASS.

[14] Algorithme des k plus proches voisins pond_er_es et application en diagnostic Eve Mathieu-Dupas.

[15] Fouille-de-donnees-JV-PART2.

[16] Apprentissage Bay_esien (R_ef_erence : Tom Mitchell, Machine Learning, Mc- Graw Hill, 1997).

[17] Zhang, M. and Zhou, Z. (2013). A review on multi-label learning algorithms. 20, 34, 35, 54

[18] Tsoumakas, G., Katakis, I., and Vlahavas, I. (2010). Mining multilabel data. In Data mining and knowledge discovery handbook, pages 667{685. Springer. 20, 34, 35, 54

[19] Madjarov, G., Kocev, D., Gjorgjevikj, D., and Dzeroski, S. (2012). An extensive experimental comparison of methods for multi-label learning. Pattern recognition, 45(9) :3084{3104. 20, 21, 34, 35, 41, 44, 52, 55, 56, 58, 62, 63, 64, 70, 94

[20] Schapire, R. E. and Singer, Y. (2000). Boostexter : A boosting-based system for text categorization. Machine learning, 39(2-3) :135{168. 20, 36, 56

[21] G. Tsoumakas, I. Katakis, and I. Vlahavas. Random k-Labelsets for Multi-Label Classi_cation. IEEE Transactions on Know-ledge and Data Engineering, 23(7) : 1079-1089, 2011.

[22] Sawsan Kanj, Fahed Abdallah, Thierry Denoeux, The evidential RAkEL method for multi-label classi_cation. Rencontres Francophones sur la Logique Floue etses Applications, 2012.

[23] Read, J., Pfahringer, B., Holmes, G., and Frank, E. (2009). Classi_er chains for multi-label classi_cation. In Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases : Part II, ECML PKDD '09, page 254{269, Berlin, Heidelberg. Springer-Verlag. 37, 43

[24] Tsoumakas, G. and Katakis, I. (2007). Multi-label classiffication : An overview. International Journal of Data Warehousing and Mining (IJDWM), 3(3) :1{13. 20, 34, 38, 50, 56

[25] Furnkranz, J., Hullermeier, E., Menc__a, E. L., and Brinker, K. (2008). Multilabel classi_cation via calibrated label ranking. Machine Learning, 73(2) :133{153. 39, 55, 56

[26] VRAIN, M. C. (2015). Classification interactive multi-label pour l'aide à l'organisation personnalisée des données (Doctoral dissertation, Orange Labs).

[27] Zhang ML, Zhou ZH. ML-KNN : A lazy learning approach to multi-label learning. Pattern Recognition ; 2007, 40(7) :2038-2048.

[28] Wieczorkowska A, Synak P, Ras Z. Multi-label classi_cation of emotions in music, in : Intelligent Information Processing and Web Mining, Springer, Berlin/Heidelberg, 2006, 307-315.

[29] Spyromitros E, Tsoumakas G, Vlahavas I. An empirical study of lazy multi-label classi_cation algorithms. In : SETN'08 : Proceedings of the 5th Hellenic Conference on Arti_cial Intelligence, Berlin, Heidelberg ; 2008, 401-406.

[30] Quinlan, J.R. : C4.5 : Programs for machine learning (1993)

[31] Clare, A., King, R.D. : Knowledge discovery in multi-label phenotype data. In : Proceedings of the 5th European Conference Principles on Data Mining and Knowledge Discovery, PKDD'01, vol. 2168, pp. 42-53. Springer (2001)

[32] Clare A, King RD. Knowledge discovery in multi-label phenotype data. In : PKDD'01 Proceedings of the 5th European Conference on Principles of Data Mining and Knowledge Discovery, Lecture Notes in Computer Science, vol. 2168 ; 2001, 42-53.

[33] Boutell M, Luo J, Shen X, Brown C. Learning multi-label scene classiffication. Pattern Recogn 2004, 37 :1757-1771.

[34] Gon_calves T, Quaresma P. The impact of NLP techniques in the multilabel text classi_cation problem. In : Proceedings of Intelligent Information Processing and Web Mining (IIPWM'04), Advances in Soft Computing ; 2004, 424-428.

[35] Lewis, D. D., Yang, Y., Rose, T. G., and Li, F. (2004). Rcv1 : A new benchmark collection for text categorization research. The Journal of Machine Learning Research, 5 :361-397. 50

[36] Elissee_ A, Weston J. A kernel method for multi-labelled classification. In : Advances in Neural Information Processing Systems (NIPS), vol. 14 ; 2001, 681-687.

[37] Madjarov G, Kocev D, Gjorgjevikj D, D_zeroski S. An extensive experimental comparison of methods for multi-label learning. Pattern Recognition ; 2012, 45 :3084-3104.

[38] Tsoumakas, G. and Vlahavas, I. (2007). Random k-labelsets : An ensemble method for multilabel classi_cation. In Proceedings of the 18th European conference on Machine Learning, ECML '07, page 406-417, Berlin, Heidelberg. Springer-Verlag. 42, 62

[39] Read J, Pfahringer B, Holmes G, Frank E. Classifier chains for multilabel classiffication, in : Proceedings of the 20th European Conference on Machine Learning, 2009, 254-269.

[40] Tsoumakas, G., Katakis, I., Vlahavas, I. : Effective and écient multilabel classification in domains with large number of labels. In : Proceedings of ECML/PKDD Workshop on Mining Multidimensional Data, MMD'08, pp, 30-44 (2008)

[41] Read, J., Pfahringer, B., Holmes, G., and Frank, E. (2009). Classi_er chains for multi-label classi_cation. In Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases : Part II, ECML PKDD '09, page 254-269, Berlin, Heidelberg. Springer-Verlag. 37, 43

[42] Madjarov G, Kocev D, Gjorgjevikj D, D_zeroski S. An extensive experimental comparison of methods for multi-label learning. Pattern Recognition ; 2012, 45 :3084-3104.

[43] Zhang, M. and Zhou, Z. (2013). A review on multi-label learning algorithms. 20, 34, 35, 44, 54

[44] Boutell, M. R., Luo, J., Shen, X., and Brown, C. M. (2004). Learning multi-label scene classification. Pattern recognition, 37(9) :1757-1771. 20, 34, 49

[45] Briggs, F., Huang, Y., Raich, R., Eftaxias, K., Lei, Z., Cukierski, W., Hadley, S. F., Hadley, A., Betts, M., Fern, X. Z., et al. (2013). The 9th annual mlsp competition : New methods for acoustic classification of multiple simultaneous bird species in a noisy environment. In Machine Learning for Signal Processing (MLSP), 2013 IEEE International Workshop on, pages 1-8. IEEE. 49

[46] University of Miami Scholarly Repository Open Access Dissertations Electronic Theses and Dissertations 2012-04-21 Hierarchical Multi-Label Classification : Going Beyond Generalization Trees Peerapon Vateekul University of Miami, peerapon.vateekul@gmail.com

[47] Classiffication hiérarchique multi étiquettes utilisant des réseaux de neurones locaux-ScienceDirect

[48] Multi-Label Hierarchical Classiffication for Protein Function Prediction HWorld Academy of Science, Engineering and Technology International Journal of Computer and Information Engineering

[49] http ://dlibrary.univboumerdes.dz :8080/bitstream/123456789/881/1/Seddoud%20Ferroudja.pdf

[50] http ://sites.millersville.edu/bikenaga/math-proof/relations/relations.html

[51] http ://mathworld.wolfram.com/PartialOrder.html

[52] https ://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-042j-mathematics-for-computer-science-fall-2010/readings/MIT6_042JF10_chap07.pdf

[53] http ://web.cs.elte.hu/˜karolyik/SER.pdf

[54] https ://www.whitman.edu/mathematics/higher_math_online/section05.03.html

[56] Djouadi, Y. (2012). Généralisation des opérateurs de dérivation de Galois en recherche d'information basée sur l'analyse formelle de concepts. In CORIA (pp. 373-386).

[57] S. Kuznetsov. A fast algorithm for computing all intersections of objects in a _nite semilattice. Automatic Documentation and Mathematical Linguistics, 27(5) :11-21, 1993.

[58] M. Chein. Algorithme de recherche des sous-matrices premières d'une matrice. Bull. Math.Soc. Sci. Math. R.S. Roumanie, 13 :21-25, 1969.

[59] E.M. Norris. An algorithm for computing the maximal rectangles in a binary relation. Revue Roumaine de Mathématiques Pures et Appliquées, 23(2) :243-250, 1978.

[60] B. Ganter. Two basic algorithms in concept analysis. FB4-Preprint 831, Technische Hochschule Darmstadt, 1984.

[61] B. Ganter and K. Reuter. Finding all closed sets : A general approach. Order, 8(3) :283-290,September 1991.

[62] J. Bordat. Calcul pratique du treillis de Galois d'une correspondance. Mathématiques et Sciences Humaines, 96 :31-47, 1986.

[63] E. Mephu Nguifo and P. Njiwoua. Using Lattice-based Feamework as a Tool for Feature Extraction, chapter 13, pages 205_216. Kluwer Academic Publishers, 1998.

[64] S.O. Kuznetsov and S.A. Obiedkov. Comparing Performance of Algorithms for Generating Concept Lattices. Journal of Experimental and Theoretical Artificial Intelligence,14 :189-216, 2002.

[65] http ://arima.inria.fr/CARI04/pdf/CARI04_08.pdf

[66] université du québec mémoire présenté à l'université du québec à trois-rivières comme exigence partielle de la maîtrise en mathématiques et informatique appliquées par labiadali sélection des mots clés basée sur la classification et l'extraction des règles d'association juin 2017

[67] université du québec mémoire présenté à l'université du québec à trois-rivières comme exigence partielle de la maîtrise en mathématiques et informatique appliquées par hassane hilali application de la classification textuelle pour l'extraction des règles d'association maximales avril 2009

[68] Chapitre5 règles d'association.pdf

[69] Chapitre 1 Fouille de données : Approche de règles d'association Maria Malek - EISTI - Deuxième année

[70] http ://eric.univ-lyon2.fr/˜ricco/cours/slides/regles_association.pdf

[71] Extraction de règles d'association selon le couple support-MGK : Graphes implicatifs et Applications en didactique des mathématiques Parfait Bemarisika.

[72] HEC Montréal les régles d'Assiciation Séquentielles par Hassen M'zali sciences de la gestion Mémoire en vue de l'obtention de la M.Sc. en Intelligence d'Affaires Avril 2006 ©Hassen M'zali, 2006.

[73] https ://fr.slideshare.net/ssuserab1db8/les-algorithmes-de-gnration-des-rgles-d-association.

[74] https ://www.webdepot.umontreal.ca/Usagers/p0706916/MonDepotPublic/CRI%201600G/Cours%20
%20Mesures%20de%20dispersion.pdf

[75] https ://www.andlil.com/definition-decart-type-130536.html

[76] http ://www.alloprof.qc.ca/BV/pages/m1508.aspx

[77] https ://fr.khanacademy.org/math/probability/data-distributions-a1/summarizing-spread-distributions/a/calculating-standard-deviation-step-by-step.

[78] See discussions, stats, and author pro_les for this publication at : https ://www.researchgate.net/publication/274314023 Data Mining avec Weka Thesis . January 2015

[79] The WEKA Workbench Eibe Frank, Mark A. Hall, and Ian H. Witten Online Appendix for \Data Mining : Practical Machine Learning Tools and Techniques" Morgan Kaufmann, Fourth Edition, 2016

[80] Meka : A Multi-label/Multi-target Extension to Weka.

[81] Tutorial. Meka 1.9.1 April 2017 A Multilabel/multitarget Extension to WEKA. http ://meka.sourceforge.net

[82] Meka : A Multi-label/Multi-target Extension to Weka.

[83] Multilabel Classi_cation : Problem Analysis, Metrics and Techniques.

[84] Journal of Machine Learning Research 17 (2016) 1-5 Submitted 5/12 ; Revised 6/15 ; Published 2/16 Meka : A Multi-label/Multi-target Extension to Weka Jesse Read jesse.read@aalto._ Helsinki Institute for Information Technology (HIIT), and Aalto University, Dept. of Computer Science, Espoo, Finland Peter Reutemann fracpete@waikato.ac.nz Bernhard Pfahringer bernhard@cs.wai- kato.ac.nz Geo_ Holmes geo_@cs.waikato.ac.nz Department of Computer Science University of Waikato, New Zealand

[85] meka.sourceforge.net

[86] Livre NAGAPPA