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
LIDIA BEDHOUCHE

VERS UNE APPROCHE DE RÉDUCTION DU NOMBRE DE RÈGLES D'ASSOCIATION

Septembre 2023

Résumé

Les règles d'association sont une technique d'exploration de données qui nous permet d'étudier et d'identifier les relations entre des éléments dans un ensemble important de données.

Le problème avec cette technique est le grand nombre de règles engendrées et la redondance. Plusieurs travaux ont été présentés afin de résoudre cette limitation. Néanmoins, les techniques présentées causent la perte de règles d'association significatives et importantes.

Dans ce travail, nous proposons une stratégie pour réduire le nombre de règles d'association en utilisant la méthode Quine-McClusckey en deux étapes. Premièrement, l'approche Quine-McCluskey est utilisée pour réduire le nombre de règles d'associations initiales. Deuxièmement, nous vérifions le taux de perte engendré par notre méthode.

Nos tests ont démontré que notre méthode réduit le nombre de règles de façon notable, tout en conservant les informations les plus pertinentes.

LIDIA BEDHOUCHE

Abstract

Association rules are a data mining technique that allows us to study and identify relationships between elements in a large data set.

The problem with this technique is the large number of rules generated and the redundancy. Several works have been presented to solve this limitation. Nevertheless, the presented techniques cause the loss of significant and important association rules.

In this work, we propose a strategy to reduce the number of association rules using the Quine-McClusckey method in two steps. First, the Quine-McCluskey approach is used to reduce the number of initial association rules. Second, we test the loss rate generated by our method.

Our tests show that our method reduces the number of rules significantly while retaining the most relevant information.

LIDIA BEDHOUCHE

Remerciement

C'est avec un immense plaisir que je réserve ces quelques lignes en signe de gratitude et de reconnaissance à tous ceux qui ont contribué de près ou de loin à l'élaboration de ce travail.

Je tiens d’abord à remercier très chaleureusement mon directeur de recherche Ismail Biskri pour sa disponibilité, sa patience et son précieux suivi tout au long de la réalisation de ce travail.

Je tiens également à remercier les membres du jury qui ont accepté de consacrer une partie de leur temps à la lecture de ce mémoire et pour l’intérêt qu’ils lui ont porté.

Mes remerciements vont également à mes chers parents et à mon mari ainsi qu’à toutes les personnes qui m’ont soutenue et encouragée pour réussir dans mon parcours scolaire.

Je remercie également tous les enseignants du département de mathématiques et d'Informatique de l’Université du Québec à Trois-Rivières.

Table des matières

Liste des Figures

Liste des Tableaux

Liste des algorithmes

Chapitre I: Introduction

Introduction

L'augmentation considérable des données, la puissance de calcul disponible des ordinateurs et la taille des données dans les réseaux ont provoqué l'émergence des techniques d'extraction des données, afin d'assurer une meilleure prise de décision. Le besoin de comprendre ce grand nombre de données complexes est incontournable dans tous les domaines du marketing, de la science ainsi que de l'ingénierie.

La recherche des données est en constante évolution, grâce au développement de l'intelligence artificielle et au Big Data. En fait, toute transaction engendre un certain nombre de données dans différents formats (des textes, des images, etc.), et les professionnels ont comme objectif de trouver un moyen permettant d'automatiser l'analyse de ces données, afin de n'extraire que les informations les plus importantes.

L'une des méthodes les plus importantes pour la recherche de données concerne les règles d'associations, le principe consistant à étudier et à analyser les bases de données, afin d'en tirer les informations les plus pertinentes. Ces informations sont présentées sous forme conditionnelle entre des antécédents et des conséquents. En d'autres termes, les règles d'associations permettent de trouver une relation entre les éléments qui sont souvent utilisés ensemble.

L'utilisation des règles d'associations est un incontournable pour analyser les données. Elle permet de trier les informations essentielles avec un effet d'explicabilité qui fournit les informations dans un format plus accessible, tout en évitant l'effet boîte noire, dont la représentation abstraite et le volume croissant des données. Néanmoins, cette méthode peut générer un nombre important de règles d'associations dont certaines sont redondantes et non pertinentes, ce qui présente un problème à résoudre, sur lequel plusieurs travaux de recherches se sont penchés. Il existe plusieurs méthodes de réduction du nombre de règles générées, cependant elles peuvent engendrer une perte de quelques importantes associations ou même une redondance.

L'évaluation d'approches d'algorithmes d'extraction de règles d'associations s'appuie principalement sur l'optimalité, le temps d'exécution, l'exhaustivité, ainsi que sur la consommation mémoire. La qualité recherchée est de pouvoir réduire le nombre de règles d'associations sans perdre des règles utiles et importantes.

Notre défi dans ce travail de recherche consiste à concevoir une approche permettant de réduire le nombre de règles d'associations sans perte d'informations pertinentes. Nous avons utilisé une méthode de minimisation des fonctions booléennes, à savoir la méthode de Quine-McCluskey. Celle-ci peut être appliquée à tout type de données et ne nécessite que des données binarisées. De plus, aucun paramètre ou métrique n'est requis de la part de l'utilisateur.

Les différentes parties de notre mise en œuvre permettent d'appliquer la méthode de Quine McClusckey aux règles d'associations en les transformant en forme booléenne. Elles permettent également de convertir les règles de retour de la forme booléenne à la forme originale des règles d'associations, afin de faciliter la lisibilité pour l'utilisateur.

Dans ce travail, nous avons testé l'effet de la réduction du nombre de règles d'associations sur la perte d'informations importantes en utilisant la classification de textes arabes. En effet, si des informations importantes sont perdues lors de la réduction, cela aura un impact négatif sur la précision de la classification. Pour étudier la perte d'informations, il nous faudra analyser la performance du système avant et après la réduction. Si la performance est constante, cela signifie que les informations importantes ont été conservées, mais si cette dernière diminue, cela indique une perte d'informations importante.

Ce mémoire est constitué de six chapitres. Dans le **deuxième chapitre**, nous présentons les règles d'associations, qui constituent une technique couramment utilisée dans l'exploration des données. Nous présenterons en détails les principales mesures utilisées dans l'extraction de ces règles. Par la suite, nous explorerons différents algorithmes utilisés pour l'extraction de règles d'associations, avec des exemples d'applications et une comparaison, afin de mieux comprendre leurs fonctionnements ainsi que leurs limites. Ce chapitre nous permettra donc d'obtenir toutes les connaissances théoriques nécessaires pour la suite de notre travail.

Dans le **troisième chapitre**, nous présentons la méthode de Quine-McCluskey, qui constitue un élément essentiel dans notre approche de réduction du nombre des règles d'associations. Nous commençons par une présentation de la méthode et de ses origines. Ensuite,

nous expliquons les principes de la méthode et son fonctionnement par des exemples. À la fin de ce chapitre, nous serons capables d'appliquer cette méthode plus efficacement pour la simplification des règles d'associations.

Dans le **quatrième chapitre,** nous présentons notre méthodologie pour la réduction du nombre de règles d'associations. D'abord, nous présentons l'architecture de notre système, et nous expliquons son fonctionnement. Ensuite, nous exposons les différentes étapes d'application de notre méthodologie ainsi que les méthodes de passage de règles d'associations en binaire.

Dans le **cinquième chapitre**, nous présenterons l'implémentation de notre système ainsi que les scripts et le langage utilisé pour mettre en œuvre notre approche pour la réduction du nombre de règles d'associations. Nous exposons aussi les résultats des expérimentations et nous évaluons les résultats obtenus en termes de perte d'informations et de performance, en comparent les résultats avant et après l'application de notre méthodologie.

Dans le **sixième chapitre**, nous exposons la conclusion finale en nous appuyant sur les résultats obtenus. Nous traitons aussi des avantages que notre méthodologie peut offrir.

Chapitre 2 : Les règles d'associations

Chapitre 2

Les règles d'association

2.1. Introduction

Les techniques d'exploration de données sont en évolution continue, en raison de l'augmentation considérable des données. L'extraction de règles d'associations (Association Rule Mining : ARM) est l'une des techniques les plus utilisées dans ce domaine.

En 1994, Rakesh Agrawal et Ramakrishnan Srikant ont apporté une contribution fondamentale à l'avancée de l'extraction de règles d'association en présentant l'algorithme Apriori [1]. Cet algorithme a marqué une avancée significative dans le domaine de l'analyse de données, en établissant les bases pour de nombreux autres algorithmes utilisés dans l'extraction de règles d'association dans un grand ensembles de données .

Les règles d'associations sont utilisées dans plusieurs domaines. Les premières utilisations ont été effectuées dans le domaine du marketing pour l'analyse des transactions commerciales, afin d'identifier les produits qui sont souvent achetés ensemble, dans le but de comprendre les habitudes des clients et d'augmenter les ventes [2]. Une autre technique qui est employée dans le domaine du marketing est le *targeting* (recommandation de produits), pour proposer des produits à un client en fonction de ses achats précédents et lui faire des propositions de produits qui sont le plus souvent achetés ensemble. Cette technique est utilisée dans plusieurs site Web pour promouvoir les ventes. Les règles d'associations ont d'autres domaines d'utilisations comme la médecine, afin d'analyser les symptômes et d'établir des relations entre des symptômes et des maladies [3] ainsi que la détection de fraudes, en analysant les comportements douteux et les transactions suspectes.

Nous rencontrons cependant plusieurs problèmes avec les règles d'associations. D'abord le temps d'exécution peut être très important, surtout quand il s'agit de bases de données de tailles importantes [4]. De plus, cette technique peut générer plusieurs règles redondantes et parfois des règles non-importantes [5]. C'est pourquoi, de nombreux défis restent à relever avec les ARM, afin de pourvoir remédier à ces problèmes.

Pour les besoins du mémoire, nous expliquons, dans ce chapitre, les notions permettant la compréhension de la notion de règles d'associations.

2.2. Définition

Les règles d'association

Les règles d'associations forment l'une des techniques les plus importante d'exploration de données (*data mining*) dont l'objectif principal est d'établir une relation entre deux ou plusieurs données [6].

L'exploration des règles d'associations est utilisée pour trouver des modèles dans un ensemble de données, en examinant les connexions entre les différents éléments de données. Ces modèles peuvent être utilisés pour prévoir des résultats futurs et faire d'autres prédictions à partir de sous-ensembles particuliers de données, tels les clients susceptibles d'acheter un produit précis.

Le format standard d'une règle d'associations est (Si - Alors) où :

A(antécédent) →B(conséquent) [7]. L'antécédent représente l'événement déclencheur, dont la conséquence est le résultat, et $\sum (A, B) \geq 1$ [8].

Considérons un client qui achète les articles A, B et C. Chaque enregistrement de vente correspond à un ItemSet (jeu d'éléments). En d'autres termes, c'est l'ensemble des éléments qui apparaissent dans la même transaction, qui est représentée par la règle d'association suivante : A, B →C. Chaque transaction est une collection de données brutes. Selon le contexte et le domaine d'application, chaque donnée est constituée d'éléments, qui peuvent prendre différentes formes (textes, images, vidéos, sons, etc.).

Soit une transaction T = {t1, t2, ..., ti, tn}, et les éléments ti de T sont des items. Un item (article) est un élément d'une transaction.

Tous les ItemSet sont des éléments de I, qui contient de deux jusqu'à n éléments de la même transaction, et un ensemble vide n'est pas considéré comme un ItemSet.

L'objectif du processus d'extraction de règles est de découvrir des relations cachées entre divers objets. Il n'existe pas de méthode particulière pour identifier les éléments à prendre en compte, mais il existe des mesures qui permettent d'identifier les éléments à ignorer [9], à savoir :

- Support : C'est la fréquence d'un élément dans l'ensemble des données.

Le support d'une règle d'association $A \rightarrow B$ est défini par [9] :

Sup (A) = $\frac{(\ nombre\ d\imath occurences(\ A)}{(\ nombre\ total\ de\ transactions)}$

- Confiance : C'est le pourcentage pour qu'une situation soit vraie, la confiance d'une règle d'association $A \rightarrow B$ est: $\mathrm{Conf(A\ \rightarrow B\ )} = \frac{\mathrm{sup(A \cup B)}}{\mathrm{sup(A)}}$

- Lift : C'est la relation entre la confiance d'une règle et la confiance attendue. Elle est donnée par la formule: $\mathrm{Lift(A\ \rightarrow B\ )} = \frac{\mathrm{Conf(A \rightarrow B)}}{\mathrm{sup(B)}}$
- Conviction : C'est la proportion de la possibilité que A soit vrai sans B.
- Intérêt :Cette mesure est utilisée pour extraire les règles les plus utiles pour l'utilisateur.
- Compréhensibilité : C'est le rapport entre la longueur de l'antécédent et le conséquent.
- Fitness fonction (fonction d'ajustement) : Utilisée pour évaluer la qualité des règles générées. Chaque approche d'ARM peut avoir sa propre fonction.

Toutes ces mesures sont interdépendantes les unes des autres, ce qui rend l'utilisation de plusieurs mesures nécessaire pour pouvoir évaluer la qualité des règles d'associations. Par exemple, une règle d'associations peut avoir un grand support mais une faible confiance, ce qui peut réduire la fiabilité de la règle et n'est pas très fiable.

2.2.1 La représentation des données

Les données sont généralement présentées sous la forme d'un tableau qui regroupe un ensemble de transactions identifiées chacune par un TID (*Transaction identifier*).

TABLEAU 1 EXEMPLE DE TRANSACTIONS

TID	Transactions
1	Pain, Lait, Café
2	Pain, Sucre
3	Sel, Sucre
4	Pain, Lait, Sucre
5	Pain, Lait, Fromage, Sucre, Café
6	Lait, Café, Fromage

Le tableau 1 représente six transactions dans un magasin d'épicerie dont chaque transaction montre un ensemble d'articles (*item*) achetés ensemble.

L'ensemble des items est représenté comme suit : I= {Pain, Lait, Café, Sucre, Sel, Fromage}, qu'on peut représenter d'une façon plus générale par : I= $\{i_1, i_2, \ldots, i_n\}$.

Chaque transaction est représentée de la façon suivante : T= $\{t_1, t_2, \ldots, t_k\}$.

{Pain, Fromage} de la transaction 5 est un ItemSet.

Par exemple, la transaction 1 dans notre tableau est : t1= {Pain, Lait, Café}.

Un ItemSet est fréquent, si et seulement si son support est supérieur ou égal à un support minimum appelé « MinSup », prédéfini par l'utilisateur.

2.2.2 Les caractéristiques d'une règle d'association

Soit la règle d'association A → B, l'importance de cette règle est mesurée à l'aide de différentes métriques qu'on a définies brièvement à la section précédente. Les métriques qui nous intéressent le plus sont : le support, la confiance et le lift.

2.2.3.1 Le support

Le support d'une règle d'association A→B est donné par le nombre de transactions où A et B apparaissent ensemble divisées par le nombre total de transactions.

$$\sup(A \rightarrow B) = \frac{nombre\ d'occurences(A \cup B)}{Nombre\ total\ de\ transactions}$$

Le support d'un ItemSet peut être considéré comme la fréquence avec laquelle l'ensemble de ses éléments sont apparus.

Exemple :

1) Soit la règle : $\text{Fromage, Café} \rightarrow \text{Lait}$, le support de cette règle est :

On a : $\sup(\text{Fromage, Café} \rightarrow \text{Lait}) = \frac{nombre\ d'occurences(\ (\text{Fromage,Café}) \cup \text{Lait})}{Nombre\ total\ de\ transactions}$

$Donc$: $\sup(\ \text{Fromage, Café} \rightarrow \text{Lait}) = \frac{2}{6} = \frac{1}{3} = 33\ \%$.

2) Soit l'ItemSet : $\{\ \text{Pain, Lait, Sucre}\ \}$, son support est donné par :

$$\text{sup}(\text{Pain, Lait, Sucre}) = \frac{nombre\ d'occurences(\text{Pain, Lait, Sucre})}{Nombre\ total\ de\ transactions}$$

On a : (Pain, Lait, Sucre) apparaissent une fois ensemble dans le tableau 1 :

$Nombre\ d'occurences(\text{Pain, Lait, Sucre}) = 2$

Nombre total de transactions = 6.

$Donc$: $\text{sup}(\text{Pain, Lait, Sucre}) = 2/6$ = 33 %.

3.2.3.2 La confiance

La confiance d'une règle d'association est définie comme la fréquence relative conditionnelle de B sachant A, c'est-à-dire la probabilité que B soit acheté (en restant sur l'exemple d'un panier d'épicerie) si A l'est également. Elle est calculée comme suit :

$$\text{Conf}(A \rightarrow B) = sup(A \cup B)/\text{sup}\ (A)$$

Exemple :

La confiance de la règle {Fromage, Café $\rightarrow$ Lait} est égale à :

$$\text{Conf}(\text{Fromage, Café} \rightarrow \text{Lait}) = sup(\ (\text{Fromage, Café} \rightarrow \text{Lait}) \cup \text{Lait})/\text{sup}\ (\text{Fromage, Café})$$

$Nous\ avons\ sup((\text{Fromage, Café} \rightarrow \text{Lait}) \cup \text{Lait}) = \frac{2}{6}$.

$Et\ le\ sup(\text{Fromage, Café}) = \frac{2}{6} = 0.5$.

$Donc$: $\text{Conf}(\text{Fromage, Café} \rightarrow \text{Lait}\ ) = 1$.

2.2.3.3 Le lift

Le lift est égal à la confiance de la règle divisée par le support de son conséquent [9].

$$\text{Lift}(A \rightarrow B) = Conf\ (A \rightarrow B)/\text{sup}\ (B)$$

Exemple :

Le lift de la règle {Fromage, Café $\rightarrow$ Lait}est égal à :

$$\text{Lift}\ (\ \{\text{Fromage, Café} \rightarrow \text{Lait}\}) = Conf\ (\ \{\text{Fromage, Café} \rightarrow \text{Lait}\}\ )/\text{sup}\ (Lait)$$

$Nous\ avons\ Conf\ (\{\text{Fromage, Café} \rightarrow \text{Lait}\}) = 1$.

$Nous\ avons\ sup(Lait) = \frac{4}{6} \simeq 0.66.$

$Donc : \text{lift (Fromage, Café } \rightarrow \text{Lait)} = \frac{1}{0.66} = 1.51.$

2.3 Extraction des règles d'associations

L'exploration des règles d'association est un processus fondamental dans le domaine de l'analyse de données. Son objectif principal est de découvrir les relations significatives entre les éléments les plus fréquents dans une base de données. Pour atteindre cet objectif, plusieurs algorithmes ont été développés, et parmi eux, deux des plus notables sont l'algorithme APRIORI et le FP-Growth [10]. Ces algorithmes sont des outils importants pour extraire des règles d'association à partir de données brutes. Ils fonctionnent en identifiant des combinaisons d'éléments qui se produisent fréquemment dans la base de données afin de générer les règles d'association. La recherche de ces règles passe principalement par trois étapes :

- La préparation des données : cette étape consiste à ne conserver que les données pertinentes ;
- La recherche des ItemSets fréquents : le but est de trouver les items qui reviennent le plus souvent. Cette étape est coûteuse en termes de temps d'exécution, surtout avec un grand nombre de données ;
- La production des règles d'associations, qui consiste à trouver les règles ayant un support≥ minSup, confiance ≥ minConf et un Lift ≥ minLift où minSup, minConf et minLift qui sont fixées par l'utilisateur et représentent les valeurs minimales qu'une règle doit satisfaire pour qu'elle soit acceptée.

2.3.1 L'algorithme Apriori

C'est l'algorithme le plus utilisé des ARM. Il est considéré comme la base des autres algorithmes d'extractions. Il a été proposé par Agrawal et Srikant en1994 [11]. Il permet de trouver les éléments les plus fréquents et de générer des règles.

Cet algorithme utilise une approche itérative où k fréquent ItemSets sont utilisés pour trouver k +1 ItemSets.

Pour améliorer l'efficacité de la génération d'ensembles fréquents d'objets, une propriété importante est employée, appelée « propriété d'Apriori », qui réduit l'espace de recherche. Elle est

définie comme suit : « Tous les sous-ensembles d'un ensemble d'éléments fréquents doivent être fréquents et si un ensemble d'objets est peu fréquent, tous ses SuperSets seront peu fréquents. »

Le fonctionnement de cet algorithme est résumé dans la section suivante (Algorithme 1) :

Input : un support minimum et une base de données de transaction.

Output : Génération des ItemSets fréquents :

1. Mi = Ø, i = 0 ;
2. C1= tous les ItemSets dans la base de données ;
3. Tous les ItemSets fréquents de C1;
   Tant que (Mi est non vide) **faire**
   1. Ci+1= Candidate-gen (Li) ;
   2. Fi+1= tous les ItemSets fréquents de ci+1;
   3. I++ ;
   4. Retourner l'union des Mi.

   Fin tant que

Algorithme 1. Fonctionnement de l'algorithme Apriori

2.3.1.1 Recherche des items fréquents

La première étape de l'algorithme Apriori consiste à examiner toutes les transactions pour extraire les items les plus fréquents dans la base de données et cela en calculant le support de chacun d'entre eux et en le comparant au seuil défini initialement par l'utilisateur. Si le support de l'item est supérieur au supMin, il sera considéré comme un item fréquent.

L'algorithme Apriori [12] commence par trouver les items fréquents dans la banque de données. Cette tâche est réalisée en parcourant chaque item présent dans une transaction et en calculant son support (si le support de l'item n'a pas déjà été calculé). Si le support de l'item est supérieur ou égal au seuil (déterminé par l'utilisateur), l'item en question est retenu comme fréquent. Le principe Apriori démontre son utilité ici pour minimiser l'espace de données, c'est-à-dire qu'aucun sur-ensemble d'éléments non fréquents ne doit être testé.

2.3.1.2 RECHERCHE DES ITEM-SETS FRÉQUENTS

Après avoir trouvé les items fréquents, tous ceux qui ne le sont pas doivent être éliminés. Cette deuxième étape de l'algorithme a pour but de trouver les Item-sets fréquents. Ensuite, l'algorithme génère des sous-ensembles de deux éléments minimum (seuls les ItemSets qui respectent le seuil minimum sont retenus). Le même processus est appliqué jusqu'à ce que la liste de candidats soit nulle (ne respecte pas le seuil minimum).

2.3.1.3 GÉNÉRATION DES RÈGLES D'ASSOCIATIONS

Afin de bien comprendre le fonctionnement de cet algorithme, nous donnerons l'exemple qui suit.

Les grands magasins utilisent des méthodes pour augmenter les achats des clients et aider les entreprises à faire des profits. Essayons d'appliquer l'algorithme Apriori dans le tableau suivant où chaque lettre correspond à un article d'un panier d'épicerie :

TABLEAU 2 TABLEAU DE TRANSACTIONS

TID	Transactions
1	A, C, D
2	B, C, E
3	A, B, C, E
4	B, E
5	A, C, E

Supposons que le support minimum (minSup) requis est égal à 2.

- **Itération 1 :**

  On calcule l'occurrence de chaque élément de $C1$, et on compare chaque valeur avec la valeur minSup, afin d'éliminer les items qui ne satisfont pas le minSup. Le résultat obtenu détermine l'ensemble fréquent, F1.

C1

Itemset	Support
A	3
B	3
C	4
D	1
E	4

F1

Itemset	Support
A	3
B	3
C	4
E	4

On élimine l'item D, car son support est inférieur au minSup.

- **Itération 2 :**

Ici, on l'applique sur un groupe de deux éléments (uniquement les éléments présents dans F1).

C2

Itemset	Support
{A, B}	1
{A, C}	3
{A, E}	2
{B, C}	2
{B, E}	3
{C, E}	3

F2

Itemset	Support
{A, C}	3
{A, E}	2
{B, C}	2
{B, E}	3
{C, E}	3

- **Itération 3 :**

ItemSet de taille 3

C3

Itemset	Support
{A, B, C}	Non (priorité d'Apriori)
{A, B, E}	Non (priorité d'Apriori)
{A, C, E}	2
{B, C, E}	2

F3

Itemset	Support
{A, C, E}	2
{B, C, E}	2

- **Itération 4 :**

On calcule l'ensemble $C4$, qui est constitué d'un seul candidat, {A, B, C, D, E}, qui ne satisfait pas le minSup, et l'algorithme se termine.

C4

Itemset	Support
{A, B, C, E}	2

2/Génération des règles d'associations :

Une règle d'associations forte doit satisfaire le minSup et le minConf.

Pour chaque ItemSet non vide S de I, on retournera la règle :

$s \Rightarrow$ (I− s) si support (I − s) / support (s).

En appliquant ces règles sur les items de F3, on obtiendra :

S1= {A, C, E} et les sous-ensembles non-vides de S sont : {A, C}, {A, E}, {C, E,{A},{C},{E}

Mettant minConf = 60

- Règle 1 : {A, C} ⇒ ({A, C, E} - {A, C})

Confiance =support (A, C, E) /support (A, C) = 2/3 = 66.66 % >minConf

Règle 1 sélectionnée.

- Règle 2 : {A, E} ⇒ ({A, C, E} - {A, E})

Confiance = Support (A, C, E) /Support (A, E) = 2/2 = 100 % >minConf

Règle 2 sélectionnée.

- Règle 3 : {C, E} ⇒ ({A, C, E} - {C, E})

Confiance = support (A, C, E) /support (C, E) = 2/3 = 66.66 % >minConf

Règle 3 sélectionnée.

-Règle 4 : {A} ⇒ ({A, C, E} - {A})

Confiance =support (A, C, E) /support (A) = 2/3 = 66.66 % >minConf

Règle 4 sélectionnée.

- Règle 5 : {C} ⇒ ({A, C, E} - {C})

Confiance = support (B, C, E) /support (C) = 2/4 = 50 % <minConf

Règle 5 rejetée.

- Règle 6 : {E} ⇒ ({A, C, E} - {E})

Confiance = support (, C, E) /support (E) = 2/4 = 50 % <minConf

Règle 6 rejetée.

S2 = {B, C, E} et les sous-ensembles non-vides de S sont : {B, C}, {B, E}, {C,E},{B},{C},{E}

- Règle 1 : {B, C} ⇒ ({B, C, E} - {B, C})

Confidence = Support (B, C, E) / Support (B, C) = 2/2 = 100 % >minConf

Règle 1 sélectionnée.

- Règle 2 : {B, E} ⇒ ({B, C, E} - {B, E})

Confiance = Support (B, C, E) / Support (B, E) = 2/3 = 66,66 % >minConf

Règle 2 sélectionnée.

- Règle 3 : {C, E} ⇒ ({B, C, E} - {C, E})

Confiance = Support (B, C, E) / Support (C, E) = 2/3 = 66,66 % >minConf

Règle 3 sélectionnée.

-Règle 4 : {B} ⇒ ({B, C, E} - {B})

Confiance = Support (B, C, E) / Support (B) = 2/3 = 66,66 % >minConf

Règle 4 sélectionnée.

- Règle 5 : {C} ⇒ ({B, C, E} - {C})

Confiance = Support (B, C, E) / Support (C)b= 2/4 = 50 % <minConf

Règle 5 rejetée.

- Règle 6 : {E} ⇒ ({B, C, E} - {E})

Confiance = Support (B, C, E) / Support (E) = 2/4 = 50 % <minConf

Règle 6 rejetée.

Les résultats de S2 peuvent être résumés dans le tableau 3:

TABLEAU 3 RÉSULTATS DE LA GÉNÉRATION DE RÈGLES D'ASSOCIATIONS

Règle	Forme	Confiance	Résultat
R1	{B, C} → ({B, C, E} − {B, C})	66.66 %	Retenue
R2	{B, E} → ({B, C, E} − {B, E})	66.66 %	Retenue
R3	{C, E} → ({B, C, E} − {C, E})	66.66 %	Retenue
R4	{B} → ({B, C, E} − {B})	66.66 %	Retenue
R5	{C} → ({B, C, E} − {C})	50 %	Rejetée
R6	{E} → ({B, C, E} − {E})	50 %	Rejetée

L'algorithme Apriori est considéré comme une base pour d'autre algorithmes d'extraction de règles d'associations, mais il présente quelques limitations. Premièrement, il génère un grand nombre de règles d'associations (comme tous les algorithmes liés au support/confiance), où plusieurs règles peuvent ne pas être intéressantes, ce qui va occuper un espace mémoire très important. Deuxièmement, il nécessite beaucoup d'analyses, ce qui rend le temps d'exécution très long ; par conséquent, l'un des critères les plus importants pour l'évaluation d'un algorithme n'est pas vérifié.

2.3.2 Algorithme Apriori TID

L'algorithme Apriori TID est une variante de l'algorithme Apriori proposée par Agrawal et Srikant en 1997 comme alternative à l'algorithme Apriori [13]. Cet algorithme permet la découverte des éléments les plus fréquents dans une base de données de transactions et produit exactement les mêmes résultats qu'Apriori, mais il fonctionne d'une façon différente.

L'algorithme Apriori TID reçoit la base de transactions en Input et retourne les ItemSets fréquents (un ItemSet fréquent est un ItemSet qui apparaît au moins minSup fois).

L'algorithme Apriori TID transforme les bases de données transactionnelles en transactions basées sur des éléments pour des recherches plus rapides. Cette transformation élimine le besoin d'analyser l'ensemble de données plusieurs fois. Dans l'algorithme Apriori TID, seule la première lecture de la base de données d'origine est effectuée. De plus, les lectures sont acheminées vers les bases de données candidates. Il s'agit de l'ensemble des éléments candidats élagués en fonction du support minimum. Par conséquent, l'algorithme d'Apriori TID devient très efficace à la fin du processus, lorsque la taille de C_k diminue [14].

2.3.3 Algorithme FP-Growth

L'algorithme FP-Growth [15] se sert d'une approche permettant d'extraire des règles d'associations à partir de thèmes communs. La méthode condense la banque de données en un arbre de motifs fréquents appelée "FP-Tree" (Frequent Pattern Tree). L'algorithme commence par analyser la banque de données pour produire cet arbre.

Afin de générer l'arbre FP-Tree, l'algorithme compte le nombre d'occurrences de chaque élément de la base de données individuellement, pour ensuite ne laisser que les items ayant le support minimal (qui était fixé initialement par l'utilisateur), puis il ordonne les items retenus dans un ordre croissant selon leurs nombres d'occurrences. Après avoir ordonné les items, l'algorithme trie le tableau de transactions d'après le nombre d'apparitions d'items, afin de pouvoir par la suite traiter toutes les transactions une par une.

La deuxième phase de l'algorithme est l'exploration de l'arbre FP-Tree dans lequel on va pouvoir générer l'arbre à partir de notre tableau de transactions générées, pour enfin pouvoir construire l'arbre FP-Tree dans lequel on visualisera les règles d'associations.

Afin de bien comprendre le fonctionnement de cet algorithme, nous présenterons un exemple simple.

Exemple :

Appliquons l'algorithme FP-Growth à l'exemple précédent.

Soit le tableau de transactions suivant :

EXEMPLE DE TRANSACTIONS

TID	Transactions
1	Pain, Lait, Café
2	Pain, Sucre
3	Sel, Sucre
4	Pain, Lait, Sucre
5	Pain, Lait, Fromage, Sucre, Café
6	Lait, Café, Fromage

- Phase I : Construction du FP-Tree

- Étape 1 : Tableau trié
  Cette étape consiste à calculer le support de chaque élément dans notre base de données.
  Soit le Minsup = 2, on obtient le résultat suivant (Tableau 4) :

TABLEAU 4 TABLEAU DE SUPPORT

Item	Support
Pain	4
Lait	4
Café	3
Sucre	3
Sel	1 (Sup< MinSup)
Fromage	2

- Étape 2 : Construction du FP-Tree
  Dans cette étape, nous allons extraire les ItemSets de taille (1) ; il s'agit du premier scan de notre base de données.
  - Matrice de données formelles : cette matrice représente le nombre d'apparitions de chaque item dans chaque transaction.

TABLEAU 5 MATRICE DE DONNÉES FORMELLES

TID	Pain	Lait	Café	Sucre	Sel	Fromage
T1	1	1	1	0	0	0
T2	1	0	0	1	0	0
T3	0	0	0	1	1	0
T4	1	1	0	1	0	0
T5	1	1	1	1	0	1
T6	0	1	1	0	0	1

TABLEAU 6 TABLEAU DES SUPPORT TRIÉS

Item	Support
Pain	4
Lait	4
Café	3
Sucre	3
Fromage	2

L'item sel est éliminé, car son support est < 2.
À cette étape, il faut ordonner les transactions selon les supports d'items.

TABLEAU 7 TABLEAU DE TRANSACTIONS TRIÉES

Transaction	Produit (Item)	Item trié
T1	Pain, Lait, Café	Pain, Lait, Café
T2	Pain, Sucre	Pain, Sucre
T3	Sel, Sucre	Sucre, Sel (Éliminé)
T4	Pain, Lait, Sucre	Pain, Lait, Sucre
T5	Paint, Lait, Fromage, Sucre, Café	Pain, Lait, sucre, Café, Fromage
T6	Lait, café, Fromage	Lait, café, Fromage

Le principe de la création de l'arbre FP-Tree consiste à traiter les transactions une par une (T1→T2→T3→T4→T5→T6) ; on obtient le résultat suivant :

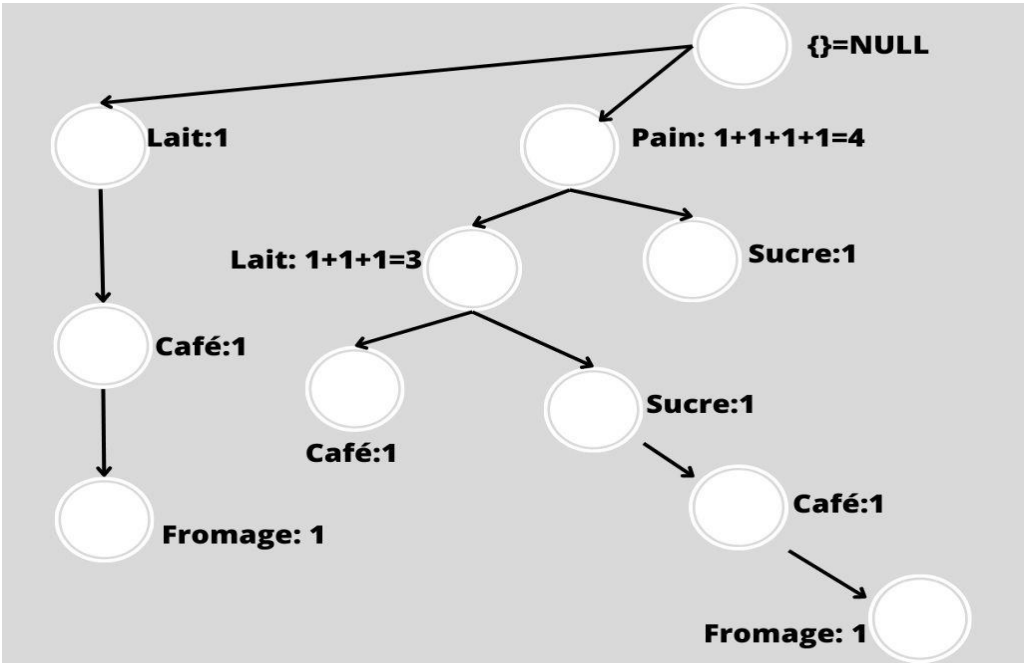


FIGURE 1 ARBRE FINAL

- Phase II : Exploration de FP-Tree

Commençons par l'item ayant le support le plus faible (fromage) :

- Pour l'item fromage, nous avons deux chemins possibles.
  - Pain, Lait, Sucre, Café, Fromage
  - Lait, Café, Fromage
- Pour l'item sucre, nous avons deux chemins possibles.
  - Pain, Sucre
  - Pain, Lait, Sucre
- Pour l'item café, nous avons trois chemins possibles.
  - Lait, Café
  - Pain, Lait, Café
  - Pain, Lait, Sucre, Café
- Pour l'item lait, nous avons deux chemins possibles.
  - Pain, Lait
  - Lait
- Pour l'item pain, nous avons un seul chemin possible.
  - Pain

L'ensemble des items fréquents correspond à l'ensemble des ItemSets fréquents de taille 1 du tableau trié ainsi que des items trouvés dans la phase II Cad : {Pain}, {Lait},{Café},{Sucre},{Fromage},{Lait, Fromage},{Café, Fromage},{Lait, Café, Fromage}, {Pain, Sucre},{Lait, Café},{Pain, Café},{Pain, Lait, Café},{Pain, Lait}.

Après l'application des algorithmes Apriori et FP-Growth, nous constatons que les deux ont le même objectif, mais avec une approche différente. L'algorithme Apriori est fondé sur la génération de règles, tandis que l'algorithme FP-Growth utilise une structure arborescente pour l'extraction des items fréquents sans aucune génération de règle.

La structure arborescente que le FP-Growth utilise le rend plus efficace en termes de complexité et de mémoire que l'algorithme Apriori pour les grandes bases de données, mais les deux peuvent présenter des limites au niveau de la redondance.

L’utilisation de règles d’associations présente quelques limites qui doivent être prises en considération. Par exemple, si la quantité de données est importante, cela peut prendre beaucoup de temps et beaucoup de mémoire pour l’analyser. De plus, des données inexactes ou incomplètes pourraient fausser les résultats et conduire à la découverte de règles d’associations inutiles.

Afin de mieux comprendre leurs limites, nous présentons un exemple comparatif des différents algorithmes présentés précédemment (Apriori, Apriori-TID et le FP-Growth) en termes de temps d’exécution et de redondance des règles d’associations générées.

Pour cette comparaison, nous utilisons le data-set « Groceries », qui est constitué d'un mois de données réelles de transactions effectuées dans une épicerie sur une période de 30 jours. Cet ensemble de données comporte un total de 9835 transactions. La source de cet ensemble de données "Groceries" est attribuée à Michael Hahsler, Kurt Hornik et Thomas Reutterer, qui l'ont fourni pour être utilisé avec le package "arules." En langage de programmation R[14*] .

Lien : R : Ensemble de données sur les transactions d’épicerie (r-project.org)

Nous fixons le MinSupport à 2 et la confiance à 60 % pour les trois algorithmes : Apriori, Apriori-TID et FP-Growth. Nous avons exécuté les trois algorithmes et mesuré leurs temps d'exécution et la redondance des règles générées.

Nous avons obtenu les résultats suivants (Tableau 8) :

TABLEAU 8 RÉSULTATS DE COMPARISON

Algorithme	Le taux de redondance	Temps d’exécution
Apriori	0,87	3,68 s
Apriori-TID	0,87	1,77 s
FP-Growth	0,98	0,18 s

Nous avons les taux de redondance des algorithmes Apriori et Apriori-TID, qui sont pareils (0,87), ce qui signifie que ces algorithmes fournissent des règles redondantes dans 87 % des cas. D’un autre côté, l’algorithme FP-Growth a produit des règles encore plus redondantes, avec un taux de redondance d’environ 0,98.

Le temps d’exécution le plus rapide a été obtenu en utilisant l’algorithme FP-Growth, qui a généré les règles d’associations après environ 0,18 seconde. Les règles ont été générées par les algorithmes Apriori et Apriori-TID après environ 3,68 et 1,77 secondes respectivement.

Les résultats que nous avons obtenus pour le temps d’exécution et le taux de redondance dépendent des paramètres que nous avons choisis (support et confiance), mais des métriques différentes peuvent conduire à des résultats dissemblables, d’où vient l’importance du choix des métriques pour obtenir les meilleurs résultats possibles.

Les algorithmes d'exploration des règles d'associations peuvent prendre beaucoup de temps, en fonction de la taille de la base de données et des paramètres choisis. De plus, des données redondantes peuvent conduire à la découverte de règles d'associations inutiles ou incorrectes.

2.4. Conclusion

Dans ce chapitre, nous avons abordé la notion de règles d'associations ainsi que les différentes méthodes d'exploration des règles d'associations telles que : les algorithmes Apriori, Apriori TID, FP-Growth avec des exemples, ce qui nous a permis d'illustrer leur fonctionnement ainsi que les limites que présente chacun de ces algorithmes, surtout quand il s'agit d'un grand nombre de données.

Cette présentation des algorithmes nous a permis de constater l'importance de réduire le nombre de règles d'associations générées, tout en évitant la redondance de données.

Dans le chapitre suivant, nous présentons la méthode de Quine-McCluskey et ses principes fondamentaux.

Chapitre 3 : La méthode de Quine-McCluskey

Chapitre 3

Quine-McClusckey

3.1. Introduction

La méthode Quine-McCluskey est une technique d'optimisation de la logique numérique utilisée pour simplifier les expressions de l'algèbre booléenne. Elle est particulièrement utile pour les expressions booléennes complexes et de grande taille qui ne peuvent être simplifiées facilement par inspection ou par des techniques de manipulation algébrique standards.

3.2. Quine-McClusckey

3.2.1. Présentation de la méthode

La méthode Quine-McCluskey est une méthode d'optimisation des fonctions booléennes qui a été développée par Willard Van Orman Quine, ensuite étendue par Edward J. McCluskey en 1952, d'où provient son nom. C'est une méthode très utilisée dans la conception des circuits intégrés [15], qui sont des composants électroniques regroupant un grand nombre de fonctions sur une puce minuscule. Ces circuits intégrés sont utilisés dans une multitude de domaines tels que l'électronique, les systèmes embarqués, les ordinateurs, les véhicules automobiles, et bien d'autres applications. La méthode Quine-McCluskey contribue à optimiser les fonctions logiques de ces circuits intégrés, améliorant ainsi leur efficacité, réduisant les coûts et préservant leurs performances essentielles.

Le fonctionnement de la méthode Quine-McCluskey repose sur l'identification des termes logiques qui peuvent être combinés pour former des termes plus simples. Elle utilise une table de vérité qui décrit la sortie d'une fonction logique pour chaque valeur d'entrée possible. Ensuite, elle organise des groupes en fonction du nombre de bits partagés, en commençant par une combinaison unique de présence ou d'absence dans la table de vérité. Ces groupes sont ensuite combinés pour former des termes plus simples, permettant ainsi l'élimination des variables présentes dans des termes contradictoires. La méthode se répète jusqu'à ce que toutes les combinaisons possibles aient été explorées [16].

La méthode Quine-McCluskey (QM) est particulièrement utile pour simplifier les circuits logiques de grande taille. Elle peut être employée pour concevoir des systèmes de contrôle automatisés, des processeurs, des microcontrôleurs, des circuits d'horloge et autres applications électroniques [17].

3.2.2 Principe de base et fonctionnement

Une expression booléenne est constituée d'une ou de plusieurs variables, selon la complexité du système étudié. Chaque variable est soit vraie (1) ou fausse (0) [18]. Par exemple, pour l'expression booléenne suivante : E= ABCDE+ABDE.

Dans cette expression, nous avons deux termes « ABCDE » et « ABDE », et cinq éléments, à savoir : « A », « B », « C », « D » et « E ». Nous constatons que « A », « B », « D » et « E » sont des variables communes aux deux termes. L'élément « C » est présent dans le terme « ABCDE » et absent dans « ABDE ». Alors «ABDE» est considéré comme un facteur commun aux deux termes, et la différence entre les deux termes est noté « - ». On aura donc dans cet exemple : ABDE, dans lequel le « - » représente la variable ignorée, qui est l'élément « C » [19].

La méthode de QM utilise les annotations suivantes :

- « 1 » pour représenter le vrai ;
- « 0 » pour représenter le faux ou la négation ;
- « - » pour représenter les variables ignorées.

La méthode de QM utilise les termes pour identifier les « mintermes » (termes qui ont la valeur de sortie 1) ou les « maxtermes » (termes qui ont la valeur de sortie 0) d'une fonction booléenne [20].

Un groupe est un ensemble de termes qui ont le même nombre de bits à 1. Les termes sont généralement classés en groupes, afin de simplifier le processus de réduction des impliquants premiers. Prenons l'exemple de l'expression booléenne suivante :

E= AB'C' + ABC + AB'C + A'BC

Dans l'expression E nous avons quatre termes (AB'C', ABC, AB'C, A'BC), ou chaque terme est une combinaison de variables booléenne (A,B,C) et de leurs compléments ou le 1 représente la présence de la variable et le 0 représente son absence.

En utilisant cette notation binaire, l'expression E est résumé comme suit :

E= 100 + 111+ 101+011

Dans ce contexte, nous classons les termes en groupe comme le montre le tableau 9 :

TABLEAU 9 LA CLASSIFICATION EN GROUPE

Groupe 0	ABC
Groupe 1	AB’C, A’BC
Groupe 2	AB'C'

Dans le tableau 8 nous avons :

- Groupe 0 : Termes avec zéro bit à 1(ABC).
- Groupe 1 : Termes avec un bit à 1(A'B, AB'C, et A'BC).
- Groupe 2 : Termes avec deux bits à 1(AB'C').

Chaque ligne de la table représente un groupe, et la classification est effectuée en fonction du nombre de bits à 1 présents dans chaque terme. Les avantages de cette classification en groupes sont multiples :

- Elle simplifie le processus de réduction des impliquants premiers.
- Elle permet de regrouper les termes susceptibles d'être combinés pour former des mintermes ou des maxtermes.
- Elle facilite la recherche de combinaisons potentielles, réduisant ainsi le temps nécessaire pour simplifier l'expression booléenne.

Cette classification est une étape pour la simplification de l’expression booléenne, en regroupant les termes qui peuvent être combinés pour former des mintermes ou des maxtermes.

3.2.3. Fonctionnement

La méthode de Quine-McCluskey est une méthode qui permet de simplifier les expressions booléennes. Elle s’applique à des termes où chaque variable apparaît exactement une fois sous forme de 1 (vrai) ou de 0 (faux). Le but de cette méthode est de trouver les impliquants premiers, qui sont des éléments qui ne peuvent être combinés à d’autres éléments pour éliminer une variable. Si l’on prend l’exemple de l’expression : E = AB'CD'+ ABCD'+ ABC'D. En binaire, cette expression se présente comme suit : E = 1010 + 1110 + 1101 (le chiffre ‘1’ ou ‘0’ indiquent si la variable correspondante est présente ‘1’ ou absente ‘0’), AB'CD' ne peut être combiné à un autre terme de l’expression E.

Enfin, la méthode recherche les impliquants premiers essentiels, qui sont des impliquants premiers qui contiennent un minterme qui n’est compris dans aucun autre impliquant premier. Ces impliquants premiers essentiels sont importants, car ils sont nécessaires pour décrire toutes les sorties de la fonction booléenne.

Cette méthode permet de réduire considérablement la complexité des expressions booléennes, ce qui facilite leur analyse et leur utilisation dans la conception de circuits électroniques. Nous pouvons résumer les étapes de la méthode QM comme suit :

Prenons comme exemple d'application la fonction F = W'X'YZ '+ W'XY'Z '+ W'XY'Z + W'XYZ' +W'XYZ + WX'Y'Z +WXY'Z où (W'X'YZ ', W'XY'Z ', W'XY'Z, W'XYZ', W'XYZ, WX'Y'Z , WXY'Z) sont les termes et (W, X, Y, Z) sont les éléments de la fonction F.

- **Étape 1 :** La binarisation des termes et la formation de groupes. Chaque terme est représenté sous forme binaire, et les termes ayant le même nombre de bits à 1 sont regroupés ensemble.

TABLEAU 10 BINARISATION DES TERMES DE LA FONCTION F

TID	**Terme**	**Termes en format binaire**			
		w	**x**	**y**	**z**
1	W'X'YZ '	0	0	1	0
2	W'XY'Z '	0	1	0	0
3	W'XY'Z	0	1	0	1
4	W'XYZ'	0	1	1	0
5	W'XYZ	0	1	1	1
6	WX'Y'Z	1	0	0	1
7	WXY'Z	1	1	0	1

TID : identifiant des termes

Donc : F(WXYZ) = 0010 + 0100 + 0101+ 0110 + 0111 + 0110 + 1101.

Après la binarisation des termes de la fonction F, il faudra les classifier et les regrouper selon leur poids. Le principe de classification est le suivant : les termes de zéro bit à 1 sont placés dans le groupe 0.

- Les termes d'un bit à 1 sont placés dans le groupe 1 ;
- Les termes de deux bits à 1 sont placés dans le groupe 2 ;
- Les termes de trois bits à 1 sont placés dans le groupe 3, et ainsi de suite jusqu'au groupe n.

En appliquant ce principe de classification à notre exemple, nous obtenons la classification suivante :

TABLEAU 11 BINARISATION DES TERMES

Groupe	TID	Terme	Termes en format binaire			
			w	x	y	Z
1	1	W'X'YZ '	0	0	1	0
	2	W'XY'Z '	0	1	0	0
2	3	W'XY'Z	0	1	0	1
	4	W'XYZ'	0	1	1	0
	5	W'XYZ	0	1	1	1
3	6	WX'Y'Z	0	1	1	0
	7	WXY'Z	1	1	0	1

Cette classification permet de simplifier la fonction booléenne et de faciliter la comparaison des termes, grâce à l'identification des termes qui peuvent être combinés ensemble. C'est donc une étape cruciale de la méthode de QM.

- **Étape 2 :** la comparaison de chaque terme dans un groupe avec les termes du groupe suivant. Autrement dit, chaque terme du groupe G_i est comparé avec les termes de G_{i+1}. Les termes ayant une différence d'un seul bit sont combinés et marqués avec un "-" dans la position du bit différent, et les autres restent tels quels. Cette combinaison de termes va faire apparaître de nouveaux groupes (seuls les termes non combinés sont comparés). Par exemple, dans le tableau 11, la combinaison du groupe 1 et du groupe 2 forme le groupe 4, et la combinaison du groupe 2 et du groupe 3 constitue le groupe 5.

  La comparaison est effectuée de manière itérative. Par exemple, dans le tableau 12, si l'on prend l'exemple de comparaison entre TID (1 et 4), on a uniquement 1 bit qui change, ce qui nous donne 0-10 (W'- YZ' où W'- YZ') où le (-) représente le bit différent. Ce résultat est placé dans un nouveau groupe i+1, qui sera équivalent au groupe 4 de notre exemple.

TABLEAU 12CLASSIFICATION DES TERMES SELON LE POIDS

Groupe	TID	Termes	Termes en format binaire			
			w	x	y	Z
4	1,4	W'YZ '	0	-	1	0
	2,3	W'XY '	0	1	0	-
	2,4	W'XZ '	0	1	-	0
5	3,6	W'XZ	0	1	-	1
	3,7	XY'Z	-	1	0	1
	4,6	W'XY	0	1	1	-
	5,7	WY'Z	1	-	0	1

Ce processus est répété avec les nouveaux groupes, jusqu'à ce qu'aucune combinaison supplémentaire ne soit possible. Tout terme, qui n'a pas pu être combiné avec d'autres termes, est considéré en tant qu'impliquant premier.

TABLEAU 13 CLASSIFICATION DES TERMES SELON LE POIDS (ITÉRATION 2)

Groupe	TID	Termes	Termes en format binaire			
			w	x	y	Z
6	2,3, 4, 6	W'X	0	1	-	-
	2,4, 3, 6	W'X	0	1	-	-

Terme redondant

L'identifiant (2, 4, 4, 6) représente un terme redondant.

En éliminant la redondance, nous obtenons le tableau 14 :

TABLEAU 14 MINIMISATION DE LA FONCTION

TID	Termes	Termes en format binaire				Retenu
		w	x	y	Z	
2,3, 4, 6	W'X	0	1	-	-	Oui

Les impliquants premiers sont les éléments qui ne peuvent être combinés avec d'autres éléments pour éliminer une variable. Ils sont représentés dans le tableau 14.

TABLEAU 15 IMPLIQUANTS PREMIERS

TID	Termes	Termes en format binaire			
		w	x	y	Z
1,4	W'YZ '	0	-	1	0
3,7	XY'Z	-	1	0	1
5,7	WY'Z	1	-	0	1
2,3,4,6	W'X	0	1	-	-

Étape 3 : À cette étape, nous comparerons tous les TID non combinés dans l'étape, qui sont : (1, 4), (3, 7), (5, 7) et (2, 3, 4, 6), avec les séquences binaires de la fonction F. Le (-) peut prendre la valeur « 0 » ou « 1 », et chaque correspondance entre une séquence binaire et un terme est notée par (X). Par exemple, la séquence 0100 correspond au terme W'YZ '.

TABLEAU 16 TABLE DES IMPLIQUANTS PREMIERS

TID	Termes	Séquences binaires de la fonction F						
		0010	0100	0101	0110	0111	1001	1101
1,4	W'YZ ' (0-10)	X			X			
3,7	XY'Z (-101)			X				X
5,7	WY'Z (1-01)						X	X
2,3,4,6	W'X (01- -)		X	X	X	X		

- **Étape 4 :** Identifier les impliquants premiers essentiels, c'est-à-dire les impliquants premiers qui couvrent à eux seuls une sortie de la fonction, dont une colonne du tableau. Afin de trouver les impliquants premiers essentiels, nous prendrons en considération les colonnes avec une seule case cochée et nous établirons la correspondance avec la ligne correspondante.

TABLEAU 17 RECHERCHE DES IMPLIQUANTS PREMIERS ESSENTIELS, PREMIÈRE ITÉRATION

Impliquants premiers	TID						
	1	2	3	4	5	6	7
W'YZ '	X			X			
XY'Z			X				X
WY'Z						X	X
W'X		X	X	X	X		

Dans le tableau 17, nous remarquons que W'YZ ' couvre une colonne à lui seul, ce qui veut dire qu'il représente un impliquant premier essentiel.

Donc : S = W'YZ '.

À cette étape, nous supprimons la ligne correspondant à W'YZ ' et nous répétons exactement la même procédure dans le nouveau tableau d'impliquants premiers réduits.

TABLEAU 18 RECHERCHE DES IMPLIQUANTS PREMIERS ESSENTIELS, DEUXIÈME ITÉRATION

Impliquants premiers	TID						
	1	2	3	4	5	6	7
XY'Z			X				X
WY'Z						X	X
W'X		X	X	X	X		

W'YZ ' couvre à lui seul une colonne, ce qui veut dire qu'il représente un impliquant premier essentiel.

Donc : S = W'YZ ' + W'X.

La méthode s'arrête lorsque tous les « X » dans le tableau des impliquants premiers ont été supprimés.

TABLEAU 19 RECHERCHE DES IMPLIQUANTS PREMIERS ESSENTIELS, ITÉRATION 3

Impliquants premiers	TID						
	1	2	3	4	5	6	7
XY'Z			X				X
WY'Z						X	X

W'YZ ' couvre à lui seul une colonne, ce qui veut dire qu'il représente un impliquant premier essentiel.

Donc : S = W'YZ ' + W'X + XY'Z.

Il n'existe aucune colonne où les conditions précédentes sont vérifiées. Alors, l'algorithme s'arrête.

La méthode de Quine-McCluskey permet de minimiser une expression booléenne en utilisant des techniques combinatoires pour trouver les termes les plus simples possibles. Elle est utilisée en conception de circuits logiques et dans d'autres domaines liés à la logique booléenne.

3.3. Algorithme de Quine-McClusckey

La méthode s'appuie sur la combinaison itérative de groupes de mots en fonction de leur nombre de bits. Chaque groupe i est combiné au groupe i+1.

Supposons que nous avons deux groupes G1= {T1, T2, T3, T4} et G2 = {T5, T6, T7}

La combinaison des deux groupes revient à la combinaison de leurs termes, c'est-à-dire que chaque terme du groupe G1 est comparé à tous les termes du groupe G2. Et nous marquons le bit différent par un (-). Les termes obtenus après cette combinaison forment un nouveau groupe G+1. Voici un algorithme qui décrit ce processus :

G1, G2, G3 : groupe de termes

G1 : T1, T2, T3, T4

G2 : T5, T6

Comparaison entre T1 et T5

Comparaison entre T1 et T6

Comparaison entre T2 et T5

Comparaison entre T2 et T6

Comparaison entre T3 et T5

Comparaison entre T3 et T6

Comparaison entre T4 et T5

Comparaison entre T4 et T6

Les combinaisons des termes forment le groupe G3.

Algorithme 2. Comparaison entre les termes

Les termes non combinés représentent les impliquants premiers.

La méthode de Quine-McCluskey s'appuie donc sur des comparaisons pour créer des mintermes, tout en éliminant les redondants, jusqu'à ce qu'une expression finale soit trouvée.

Afin de mieux comprendre le fonctionnement de cette méthode, nous présenterons un algorithme qui résume son processus (Algorithme 3):

Entrée : l'ensemble des mintermes M

Sortie : expression minimale pour M

1. Créer des groupes à partir de l'ensemble M selon le nombre de bits à 1, Gi ;

2. Tant qu' (il existe des groupes à combiner) ;

2.1 . Entrer chaque terme du groupe Gi avec les termes du groupe Gi +1 ;

2.2. Chaque bit différent est marqué par (-) ;

2.3. Placer tous les termes combinés dans un groupe Gi +1 en éliminant tous les termes redondants ;

3. Si (Gi +1 est non vide), alors :

Répéter 2 avec le groupe Gi +1 ;

4. Trouver l'expression « simplifier ».

Algorithme 3. Algorithme de Quine-McCluskey

3.4. Conclusion

Dans ce chapitre, nous avons présenté la méthode de Quine-McCluskey et son rôle dans la minimisation des fonctions booléennes ainsi que son fonctionnement par un exemple pratique. Cette méthode permet la minimisation des fonctions d'une manière rapide et efficace, et elle a démontré son utilité dans l'optimisation des systèmes et des circuits logiques.

Dans le prochain chapitre nous présenterons notre méthodologie pour la réduction du nombre de règles d'associations.

Chapitre 4 : Réduction du nombre de règles d'associations

Chapitre 4

Réduction du nombre de règles d'associations

4.1. Introduction

Dans ce chapitre, nous présentons notre méthodologie pour la réduction du nombre de règles d'associations et l'architecture globale de notre travail, tout en effectuant le lien entre les règles d'associations et la méthode de Quine-McCluskey.

4.2. Méthodologie proposée

Dans cette section, nous nous pencherons d'abord sur l'architecture globale de notre système. Nous détaillerons chaque étape, tout en montrant son objectif.

Il existe des principes à prendre en considération afin de pouvoir appliquer la méthode de Quine-McCluskey aux règles d'associations. Chaque antécédent ou conséquent doit être transformé en mode en binaire dans lequel le 1 représente la présence d'un élément et le 0 figure son absence.

4.2.1. Architecture du système

Le fonctionnement de notre système est constitué principalement de trois étapes :

(a) Le système reçoit en entrée un ensemble « E » de règles d'associations ;

(b) Le nombre de règles d'associations est réduit en appliquant la méthode de Quine-McCluskey, pour obtenir un nouvel ensemble de règles « E' »;

(c) Le système utilise l'ensemble « E » comme base de connaissances pour classer un ensemble de données de test T et enregistre les résultats. Il prend ensuite l'ensemble « E' » comme base de connaissances pour classer le même ensemble de données de test T et enregistre les nouveaux résultats. Les deux résultats sont comparés, afin de trouver si une perte d'informations importantes a eu lieu.

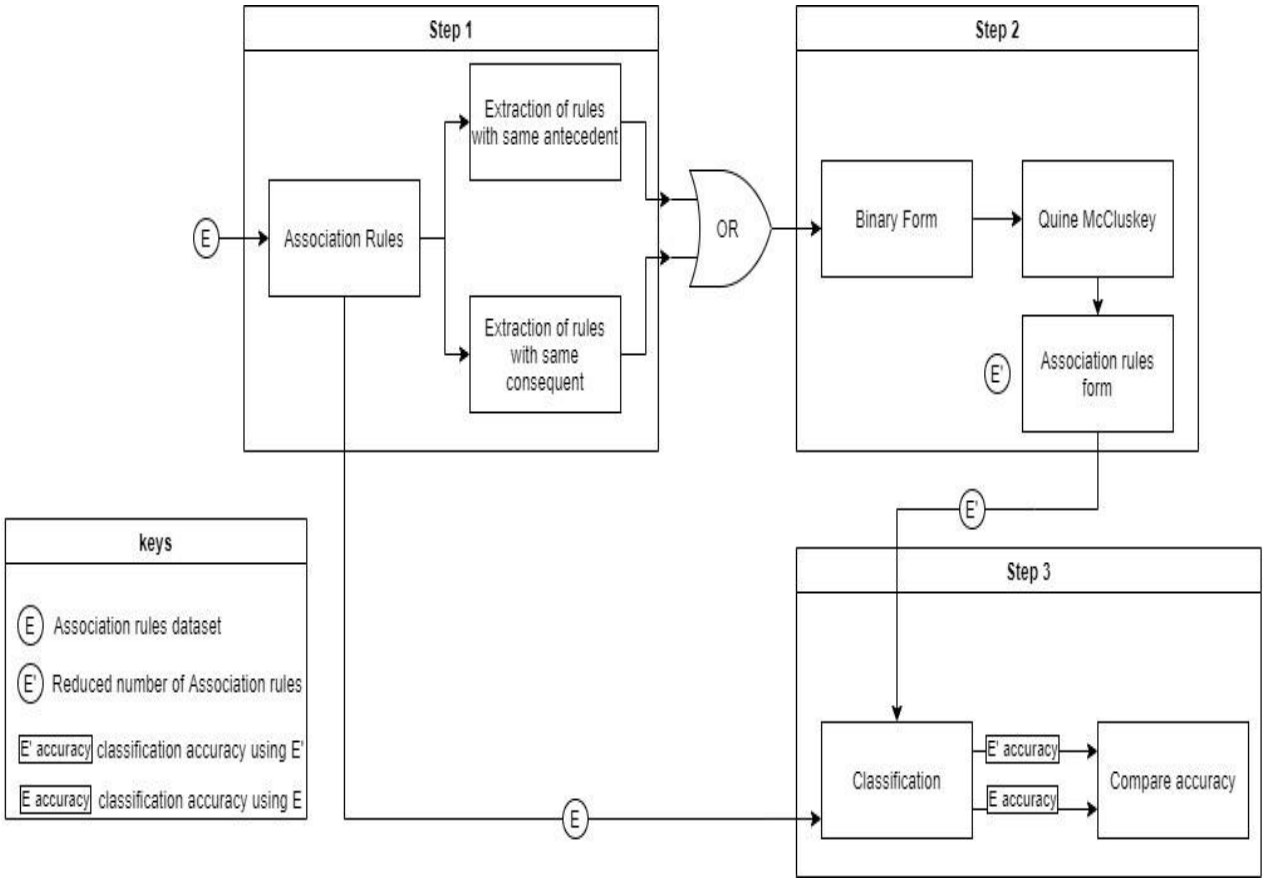


FIGURE 2 ARCHITECTURE DU SYSTÈME PROPOSÉ

La Figure 2 résume l'architecture et le fonctionnement de notre système. Dans la première étape d'extraction, l'ensemble "E" est formé en regroupant les règles ayant le même antécédent ou le même conséquent. Ensuite, pour améliorer l'efficacité et réduire la complexité, le système applique la méthode de Quine-McCluskey pour réduire le nombre de règles d'association, générant ainsi un nouvel ensemble que nous notons "E'". Par la suite, les deux ensembles "E" et "E'" sont utilisés comme base de connaissances pour classifier l'ensemble de données. Finalement, pour évaluer l'impact de la réduction des règles d'association, les deux ensembles de résultats obtenus à partir de "E" et "E'" (notés respectivement "E accuracy" et "E' accuracy") sont comparés. Cette comparaison vise à déterminer si la simplification des règles d'association a entraîné une perte d'informations.

4.2.2. Méthode pour l'application de Quine-McCluskey aux règles d'associations

Avant d'utiliser une méthode de minimisation de la fonction booléenne telle que la méthode Quine-McCluskey sur les règles d'associations, nous devons définir une relation entre la forme binaire et les règles d'associations. Cette relation est un écart entre la forme des règles d'associations et la forme binaire. L'entrée de Quine-McCluskey se trouvant sous forme booléenne, nous nous inspirons de règles d'associations négatives [21] où la négation d'un élément représente son absence. Exemple : AB'→C est égal à A→C, puisque B' représente l'absence de B. Nous avons

maintenant une relation entre la forme des règles d'associations et la forme booléenne. Une fois que les règles se trouvent sous la forme booléenne, nous les écrivons sous forme binaire, dans laquelle la négation représente 0. Nous avons maintenant notre deuxième relation entre la forme binaire et la forme booléenne. Ensuite, nous minimisons les règles avec les mêmes ensembles conséquents. Et les règles avec les mêmes antécédents sont minimisées ensemble pour garder le sens des règles. Enfin, nous devons revenir à la forme des règles d'associations. La figure 3 ci-dessous montre une séquence qui nous permet de changer la forme de n'importe quelle règle dans laquelle les règles d'associations négatives de (forme booléenne) représentent un écart ou une relation entre les règles d'associations et les formes binaires.

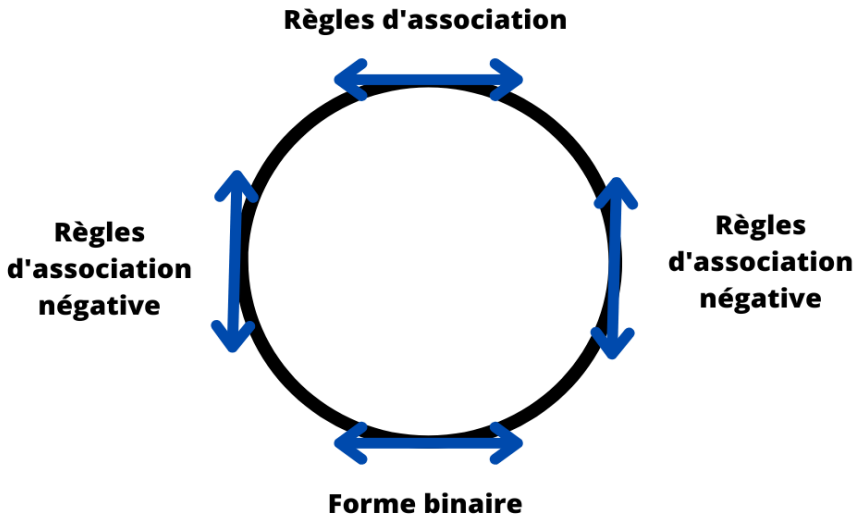


FIGURE 3 SÉQUENCE DE CHANGEMENT D'UNE RÈGLE D'ASSOCIATION

4.2.2.1 Binarisation des règles d'associations

Dans cette étape, les règles sont converties en binaires. Prenons les règles d'association suivantes pour comprendre le processus de binarisation des règles (tableau 19) :

TABLEAU 20 ENSEMBLE DE RÈGLES D'ASSOCIATIONS

Antécédent	Conséquent
A, C	E
A, B	E
A, D	E
B, C, D	E
A, B, C, D	E
A, C, B	E

Dans les règles précédentes, nous observons qu'elles ont toutes le même conséquent, à savoir « E ». Pour la binarisation, l'existence d'un item sera représentée par 1 et son absence par 0. Les résultats de la binarisation sont présentés dans le tableau suivant (Tableau 21):

TABLEAU 21 BINARISATION DE RÈGLES D'ASSOCIATIONS

Antécédents				Conséquents
A	**B**	**C**	**D**	**E**
1	0	1	0	1
1	1	0	0	1
1	0	0	1	1
0	1	1	1	1
1	1	1	1	1
1	1	1	0	1

Une règle d'associations est constituée de deux parties : l'antécédant et le conséquent. Afin d'obtenir une meilleure réduction, il est important d'appliquer cette règle sur des règles ayant le même antécédent ou le même conséquent.

Comme le montre la figure 3, nous pouvons changer la forme d'une règle dans n'importe quel sens, sans perdre la règle. Pour ce faire, nous avons développé quatre fonctions permettant de transformer la présentation des règles d'association.

Il s'agit d'utiliser nos méthodes ci-dessous où le paramètre X a deux valeurs : « a » indiquant que T/S/Sbin est un antécédent de Y ou « c » montrant que T/S/Sbin est un conséquent de Y.

Méthode de passage de règles d'associations vers les règles d'associations négatives (ARtoNAR : Association rules to negative Associationn rules)

Cette fonction permet de convertir les règles d'association en règles d'association négatives.

[X : T : Y] $\xrightarrow{\text{ARtoNAR}}$ [Y→ S] ou [S → Y]

Exemple

ARtoNAR ([ab→ c, a→c]) = [ab→ c, ab' → c]

Méthode de passage des règles d'associations négatives vers la forme binaire (NARtoBinary : negative association rules to binary)

Dans le cas des règles d'association négatives, cette fonction les transforme en forme binaire.

[X : S : Y] $\xrightarrow{\text{NARtoBinary}}$ [Y→ S_{bin}] ou [S_{bin} → Y]

Exemple :

NARtoBinary ([ab→ c, ab' →c])= [11 , 10] → c

Méthode de passage du binaire vers règles d'associations négatives(BinarytoNAR :binary to negative association rules)

Cette fonction permet de passer des règles binaires aux règles d'association négatives.

[X : S_{bin} : Y] $\xrightarrow{\text{BinaryToNAR}}$ [Y→ S] ou [S → Y]

Exemple :

BinarytoNAR ([11, 10] → c) = [ab→ c, ab' → c]

Méthode de passage des règles d'associations négatives vers les règles d'associations (NARtoAR : negative association rules)

Si nous avons des règles d'association négatives, cette fonction les ramène à la forme d'association normale.

[X : S : Y] $\xrightarrow{\text{NARtoAR}}$ [Y→ T] ou [T → Y]

Exemple :

NARtoAR ([ab→ c, ab'→c]) = [ab→ c , a → c]

En utilisant nos méthodes de changement de forme des règles d'associations, il est possible de minimiser les règles, comme s'il s'agissait de termes booléens. Prenons comme exemple de minimisation les règles ci-dessous (tableau 22):

TABLEAU 22 EXEMPLE DE RÈGLE (ANTÉCÉDENT ET CONSÉQUENT)

Antécédents	**Conséquents**
a, b	m
a, b	m, n
a, b	e
b	e
d, f	E
f	E

En commençant par les règles partageant le même conséquent e, nous pouvons inscrire :

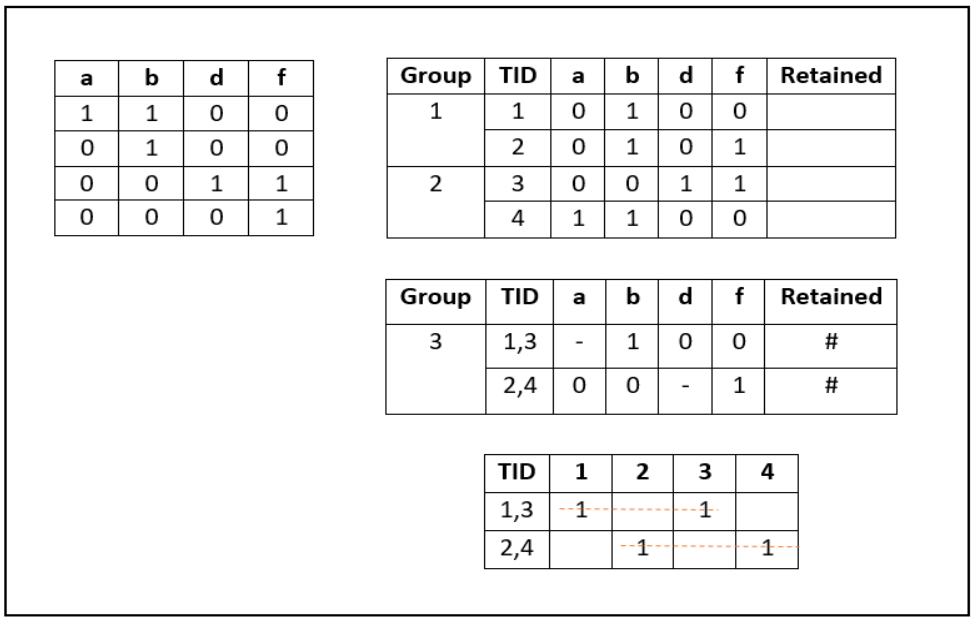
Antécédents de e

S1= ab + b + df + f = S2= abd'f' + 'abd'f' + a'b'df + a'b'd'f.

Nous pouvons maintenant écrire S2 sous forme binaire :

[**a** : abd'f' + a'bd'f' + a'b'df + a'b'd'f : e] ⟶ [**a** : 1100 + 0100 + 0011 + 0001: e].

L'application de la méthode QM sur S2 est illustrée à la figure suivante (figure 6).



a	b	d	f
1	1	0	0
0	1	0	0
0	0	1	1
0	0	0	1

Group	TID	a	b	d	f	Retained
1	1	0	1	0	0	
	2	0	1	0	1	
2	3	0	0	1	1	
	4	1	1	0	0	

Group	TID	a	b	d	f	Retained
3	1,3	-	1	0	0	#
	2,4	0	0	-	1	#

TID	1	2	3	4
1,3	1		1	
2,4		1		1

FIGURE 4 APPLICATION DE LA MÉTHODE DE QUINE-MCCLUSKEY À S2

On a : S'= -100 + 00-1. Nous pouvons inscrire le résultat sous forme de règles d'associations négatives comme suit :

[a: -100 + 00-1: e] ⟶ [a: bd'f' + a'b'f : e]

Finalement, la forme en règles d'associations est donc :

b →e, f →e

En appliquant le même processus sur les règles avec le même antécédent nous obtenons :

ab→m, ab→mn, ab→e. on obtient :

Consequents of ab

S1= m + mn + e = S2 = mn'e' + mne' + m'n'e.

Le résultat de l'application de la QM sur S' est le suivant :

me' + m'n'e (nous pouvons l'écrire aussi : ab → m, ab → e **)**

4.2.2.2 Élimination de la redondance

Après avoir réduit le nombre de règles, la base de connaissances peut être submergée par des règles redondantes.

Exemple :

TABLEAU 23 EXEMPLE DE L'ÉLIMINATION DE LA REDONDANCE

L'ensemble des règles		**Règle obtenue**	**Équivalent de la règle**
1	a, b →F	a- →F	a→ F.
	a → F		
2	a, c → F	a-→ F	
	a→ F		

Dans l'exemple illustré dans le tableau, nous avons : {ab →F, a → F} donnerait a- →F, qui se traduit par a→ F, et {a, c → F, a→ F} donnerait a-→ F se traduisant par a→ F, nous avons donc une redondance. Dès lors, toute redondance doit être éliminée. On ne garde qu'une seule règle représentative : a→ F.

4.3. Validation de l'approche proposée

L'objectif principal de notre méthode consiste à réduire le nombre de règles d'associations, tout en gardant les règles les plus importantes. Le défi dans la réalisation de cette tâche est d'éviter la perte d'informations.

La réduction du nombre de règles d'associations peut engendrer des pertes d'informations. D'où l'importance d'une vérification, afin de savoir si une perte d'informations a eu lieu.

Prenons, par exemple, l'ensemble de règles d'associations « E' », obtenu par réduction de « E ». Afin de vérifier la perte d'informations, il faudra utiliser « E » et « E' » dans une tâche commune F, et par la suite, effectuer une comparaison des résultats. C'est ce qui nous permettra de voir si la réduction effectuée par notre approche a entraîné une perte d'informations importantes.

On utilise la classification comme tâche commune et on compare les performances des classifications entre les deux états du système. Le premier est l'état avant la réduction et le second après la réduction. Si les performances des classifications (taux de précision) diminuent, cela indique clairement que des informations importantes ont été perdues (Figure 5).

Voici un schéma qui résume le processus de vérification :

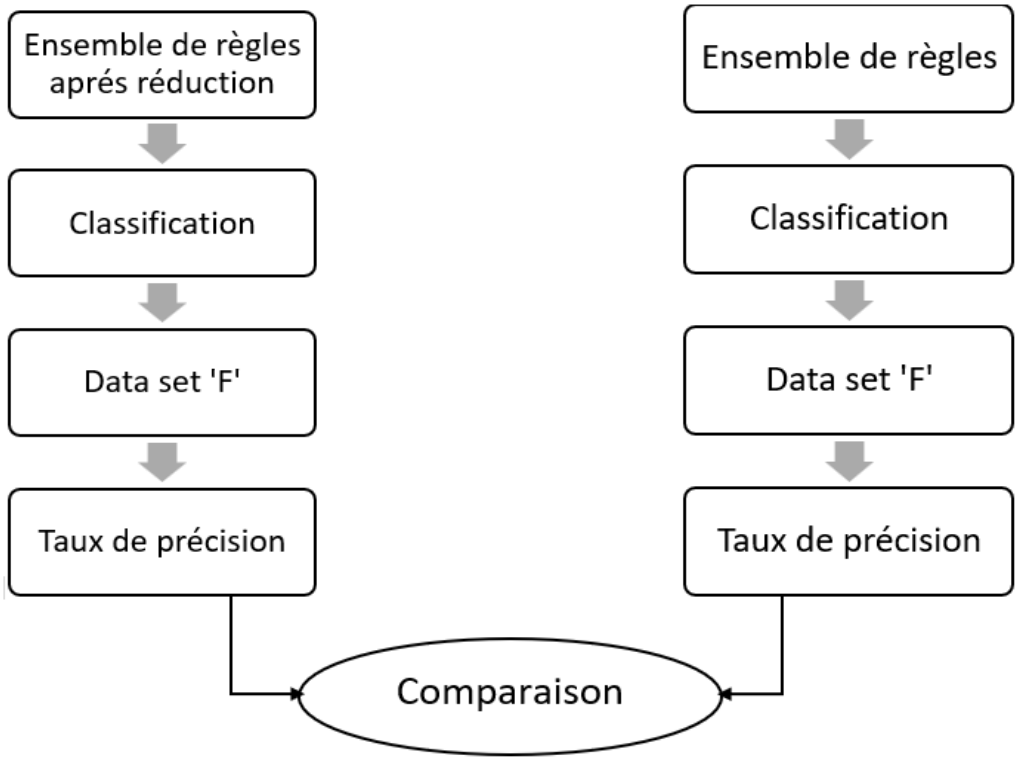


FIGURE 5 PROCESSUS DE VÉRIFICATION DES PERTES

4.4. Conclusion

Dans ce chapitre, nous avons présenté les différentes étapes de notre méthodologie pour appliquer la méthode de Quine-McCluskey ainsi que ses avantages en termes de précision.

Dans le chapitre suivant, nous présenterons l'implémentation et les résultats de l'expérimentation, afin de pouvoir évaluer l'efficacité de notre méthodologie.

Chapitre 5 : implémentation et expérimentation

Chapitre 5

Implémentation et expérimentation

5.1. Introduction

Dans ce chapitre, nous présentons la mise en œuvre de notre système ainsi que les résultats obtenus après l'application de notre approche sur un dataset (base de données).

5.2. Implémentation

5.2.1 Langage utilisé

Nous avons mis en œuvre notre système en utilisant le langage de programmation Python. Le choix de ce langage nous a donné plus de flexibilité et de performance dans le développement de notre système.

Notre système est conçu de manière à fonctionner sur différentes plateformes, cela signifie que la caractéristique de portabilité est bien présente, ce qui offrira une flexibilité à l'utilisateur.

Nous avons utilisé plusieurs scripts pour l'implémentation de notre système, à savoir :

-Extract_By_Concequent.py : est utilisé pour l'extraction des règles ayant les mêmes conséquents. Comme illustré dans l'algorithme 4 ci-dessous. Ce script permet d'analyser des ensembles de règles et d'identifier les éléments récurrents en se basant sur les résultats attendus.

D'abord le script rassemble les conséquents qui se répètent au moins deux fois dans l'ensemble de règles, afin d'en tirer les conséquents les plus significatifs et récurrents.

Une fois les conséquents identifiés, vient l'étape de la transformation des antécédents correspondant en termes, pour représenter les règles de manière plus compréhensible.

Le script utilise une fonction de binarisation avec trois paramètres clés (X, T, Y) où :

- X = c : représente le type d'extraction effectué (par conséquent ou antécédent) ;
- T : représente les termes extraits eux-mêmes ;
- Y : représente le conséquent en commun dans les règles.

Prenons l'exemple suivant pour illustrer le fonctionnement de ce script. Supposons que nous avons l'ensemble de règles suivant (Tableau 24):

TABLEAU 24 EXEMPLE DE RÈGLES(ANTÉCÉDENTS, CONSÉQUENTS)

Règles	Antécédents	Conséquents
1	A	D
2	A, B	D
3	F	E
4	E, F	E

Dans cet exemple, nous avons quatre règles avec différents antécédents et conséquents. L'objectif est d'extraire les règles qui partagent les mêmes conséquents et de procéder à la binarisation à l'aide du script « Exctract_By_Consequents.py ».

Après l'exécution du script, nous obtenons les résultats suivants :

[c: A + AB :D], [c: F + EF: E] 

[c : 10 + 11 : D], [c : 01+ 11 : E]

Dans cet exemple, les règles initiales sont extraites en regroupant les conséquents communs et en les représentant de manière binaire. Par exemple, les règles 1 et 2 sont regroupées en une seule règle [c: A + AB :D]. De même, les règles 3 et 4 sont regroupées en une règle unique [c: F + EF: E].

L'utilisation du script « Extract_By_Consequents.py » dans cet exemple nous a permis de simplifier l'analyse des règles, en mettant l'accent sur les conséquents communs.

```
Fonction Extract_by_Conséquents (x : liste) :
 Liste des variables :
- x : liste des règles d'associations
- S : fonction booléenne
- Ea : liste des items constituant les antécédents
- Ec : liste des items constituant les conséquents
- Résultat : liste résultant des règles extraites
Début
        Si réduction par même conséquents, alors
                Ea = liste_Antécédents (x)
                Pour chaque règle R dans x Faire
                        Y = Conséquents (R) // Extraire les conséquents de la règle R
                        Terme = ""  // Initialiser le terme binaire
                        Pour chaque item I dans Ea Faire
                                Si I est présent dans les antécédents de la règle R, alors
                                        Terme = Terme + I// Ajouter l'item
                                Sinon
                                Terme = Terme + I ' // Ajouter le complément I'

                        S = S + terme // Mettre à jour la fonction booléenne S avec le
                terme construit
                Résultat = Résultat + [c, S, Y]  // Ajouter la règle extraite à la liste
        résultante
Retourner la liste résultante Résultat
Fin
```

Algorithme 4. Fonction « Extract_by_Conséquents »

Extract_By_Antécédents.py : Le but de ce script est d'extraire les règles qui partagent les mêmes antécédents. Il prend un paramètre X = a (a : antécédent commun), afin de trouver l'extraction, en s'appuyant sur les mêmes antécédents. Le fonctionnement de ce script est représenté dans l'algorithme 5 suivant.

Voici un autre exemple pour illustrer le fonctionnement du script (Tableau 25) :

TABLEAU 25 EXEMPLE POUR ILLUSTRER LE FONCTIONNEMENT DU SCRIPT EXTRACT_BY_ANTÉCÉDENTS.PY

Règles	Antécédents	Conséquents
1	Z	B
2	Z	D
3	B	E, F
4	B	E

Après l'exécution du script, nous obtenons les règles extraites suivantes :

[a: B + D: Z], [a: E, F + E: B] **Binarisation** [a :10 + 01 : Z], [a : 11 + 10 : B]

Dans cet exemple, les règles qui partagent les mêmes antécédents sont regroupées. Par exemple, les règles 1 et 2 sont regroupées en une seule règle [a: BD' +B' D: Z], où « Z » est l'antécédent commun. De même, les règles 3 et 4 sont regroupées en [a : E F + EF' : B], où « B » est l'antécédent commun.

```
Fonction Extract_by_Antécédents (x : liste) : liste
   Variables
      x : règles d'associations
      S : fonction booléenne
      Ea : liste des items formant les antécédents
      Ec : liste des items formant les conséquents
Début
Résultat = liste_vide
      Si réduction par même antécédent, alors
         Ec = liste_Conséquents (x)
         Pour chaque règle R dans x Faire
                  Y = Antécédent(R) ) // Extraire les antécédents de la règle R
                  Terme = ""// Initialiser le terme binaire
                        Pour chaque item I dans Ec Faire
                        Si I est présent dans conséquents (R), alors
                              Terme = Terme + I // Ajouter l'item
                           Sinon
                              Terme = Terme + I'// Ajouter le complément I'
                        S = S + Terme // Mettre à jour la fonction booléenne S avec le
terme construit
                        Résultat = Résultat + [a, S, Y] // Ajouter la règle extraite à la liste
résultante
 Retourner Résultat
 Fin
```

Algorithme 5. Fonction « Extract_by Antécédents »

Le processus des deux scripts Extract_By_Consequent et Extract_By_Antecedent peut être résumé dans le schémas suivant :

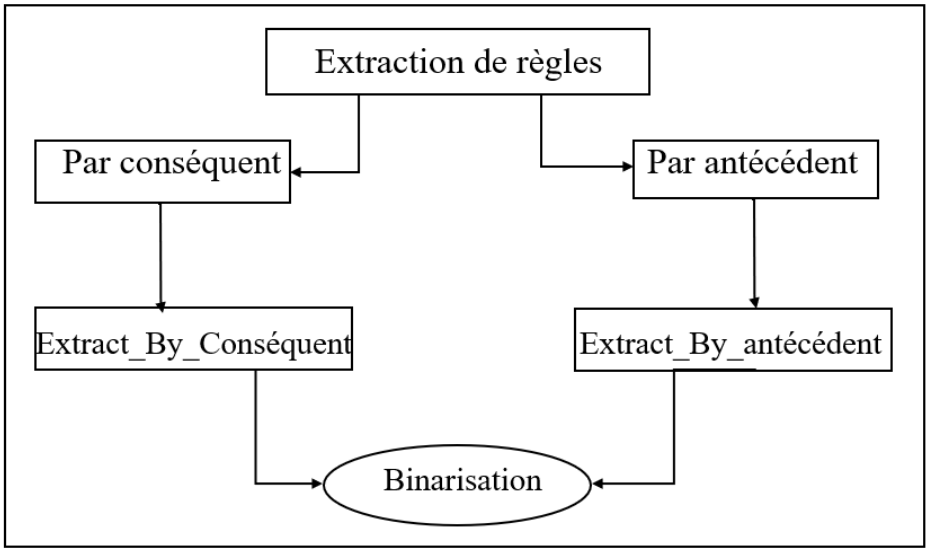


FIGURE 6 PROCESSUS DES SCRIPTS D'EXTRACTION

Le script « Quine-McCluskey.py » : Ce script met en œuvre la méthode de QM (Algorithme 6). Il prend en entrée un ensemble de termes binaires et utilise un deuxième paramètre pour spécifier la conservation du conséquent ou de l'antécédent.

Le fonctionnement du script peut être détaillé de la manière suivante :

1. La réception du premier paramètre, qui est un ensemble de termes sous la forme binaire ;

2. Le deuxième paramètre est utilisé pour spécifier si le conséquent ou l'antécédent doit être conservé dans la simplification ;

3. L'application de l'algorithme de Quine-McCluskey pour simplifier les termes binaires ;

4. Il produit une représentation simplifiée des termes sous la forme [X : S'bin : Y], (où X : les termes simplifiés, S'bin : la forme binaire simplifiée, Y : le conséquent ou l'antécédent conservé).

Le fonctionnement du Script peut être représenté comme suit :

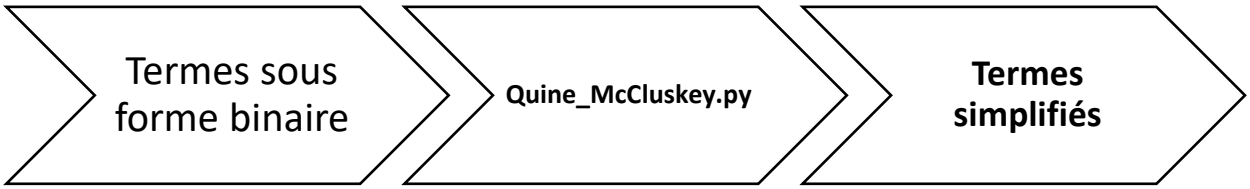


FIGURE 7 PROCESSUS DU SCRIPT QUINE-MCCLUSKEY.PY

Voici un autre exemple pour illustrer le fonctionnement du script :

[c : 1100 + 1000 : d] 

[c : 1-00 :d]

Le script « Quine-McCluskey.py » met en pratique la méthode de Quine-McCluskey et permet ainsi de simplifier efficacement des termes binaires. Présentons l'algorithme qui représente son déroulement (Algorithme 6):

Fonction QUINE-MCCLUSKEY(X : caractère, Sbin : liste, Y : chaîne de caractères) : liste

Début

Si X = 'a' alors

S'bin = QUINE-MCCLUSKEY(Sbin) // Application de l'algorithme de Quine-McCluskey sur Sbin

Retourner (a, S'bin, Y)

Sinon si X = 'c' alors

S'bin = QUINE-MCCLUSKEY(Sbin) // Application de l'algorithme de Quine-McCluskey sur Sbin

Retourner (c, S'bin, Y)

Fin Si

Fin

Algorithme 6. Fonction Quine-McCluskey

Cet algorithme définit la fonction QUINE-MCCLUSKEY avec les paramètres X, Sbin et Y. La fonction vérifie la valeur de X pour déterminer si l'opération doit être effectuée sur le conséquent (X = 'c') ou sur l'antécédent (X = 'a'). Ensuite, elle applique l'algorithme de Quine-McCluskey sur la liste Sbin pour simplifier les termes. Enfin, le script retourne la liste des termes simplifiés.

Le script_GetRules.py : Dont le fonctionnement est illustré dans l'algorithme 7, permet le passage des règles binaires en règles régulières. Le paramètre X peut prendre la valeur « a » pour indiquer que Y représente un antécédent, ou « c » pour énoncer que Y représente un conséquent. Son fonctionnement peut être illustré comme suit :

FIGURE 8 FONCTIONNEMENT DU SCRIPT GETRULES.PY

L'exemple suivant illustre le fonctionnement du script :

[a: 1101 + 1010: E] se transforme en [E→ABC'D, E →AB'CD']

[c: 10-0 + 1-01: E] se transforme en [AB'D' → E, AC'D → E]

Dans cet exemple, les règles initiales sont présentées sous la forme binaire [X : Sbin : Y]. Le script GetRules.py les transforme en règles régulières, où Y est associé à S. Pour les paramètres X = 'a', les règles sont présentées sous la forme « Y → S », tandis que pour X = 'c', elles sont affichées sous la forme « S → Y ».

Fonction GetRules (X : caractère, Sbin : liste, Y : chaîne de caractères) :

Liste variable :

Antécédent : Antécédent d'une règle d'associations

Conséquent : Conséquent d'une règle d'associations

Début

Si X = 'c' alors // Si X est égal à 'c', on construit une règle avec Y comme conséquent

Pour chaque élément t dans Sbin faire

Si e == 0 alors //On ajoute son complément négatif à l'antécédent

Antécédent = Antécédent + t'

Si e == 1 alors //On l'ajoute directement à l'antécédent

Antécédent = Antécédent + t

Retourner [Antécédent → Y] // Retourne la règle complète sous forme [Antécédent → Y]

Si X = 'a' alors // Si X est égal à 'a', on construit une règle avec Y comme antécédent

Pour chaque élément e dans Sbin faire

Si e == 0 alors //On ajoute son complément négatif au conséquent

Conséquent = Conséquent + t'

Si e == 1 alors alors //On l'ajoute directement au conséquent

Conséquent = Conséquent + t

Retourner [Y → Conséquent] // Retourne la règle complète sous forme [Y → Conséquent]

Fin

Algorithme7. Fonction GetRules

5.2.2 La suppression de la redondance des règles d'associations

Après la réduction du nombre de règles d'associations, il peut y avoir des règles redondantes, par exemple, les règles {A, B → C, A → C}. Après réduction, nous obtenons la règle A- → C (équivalente à : A → C). De même, les règles {A→ C, AD → C} se réduisent à A- → C,

qui est équivalent à : A → C. Nous avons donc une redondance où la même règle est exprimée de différentes manières.

Il est alors nécessaire de ne conserver qu'une seule règle. Dans notre exemple, la règle

A → c serait conservée, car elle regroupe les informations fournies par les règles redondantes.

La suppression de la redondance des règles d'associations permet de simplifier la base de connaissances, d'éviter la redondance et de ne conserver que les règles les plus pertinentes.

5.3 Expérimentations et résultats

Dans cette section, nous présentons les résultats obtenus et notre approche de validation. Nous utilisons une base de données comprenant des règles d'associations et des règles de classification dans lesquelles le résultat est une classe.

Nous avons appliqué notre méthode sur le *Dataset* de la langue, il contient plusieurs articles de journaux en langue arabe.

Nous définissons les règles d'associations régulières comme suit : A (antécédents) → B (conséquents), où A et B sont supérieurs ou égaux à 1. Par exemple : Lait, Café → Pain, tandis que les règles d'associations de classe ne peuvent comporter qu'un seul élément dans la partie conséquente, qui représente une classe. Elles se présentent sous la forme : n (antécédents) → Classe, où n est supérieur ou égal à 1. Par exemple : Film, Acteur → Culture.

5.3.1 Banque de données

Pour réaliser nos expérimentations, nous avons utilisé la banque de données SANAD (Single-Label Arabic News Articles Dataset for Automatic Text Categorization) [22], qui contient une vaste collection d'articles en langue arabe (plus de 190 000 articles) qui peuvent être utilisés pour le traitement automatique du langage. Cette banque de données contient sept différentes catégories, à savoir : Culture, Finance, Médical, Politique, Religion, Sport et Technologie [23].

Pour l'extraction des règles d'associations, nous avons mis en œuvre des méthodes éprouvées dans l'extraction des règles d'associations dans des textes en langue arabe. Nous avons utilisé plusieurs méthodes, à savoir :

- **Le stemming :** Cette technique permet la réduction de mots. Nous l'avons utilisée afin de regrouper différentes variables du même mot, ce qui permet la simplification du processus d'extraction des règles d'associations [24] ;
- **La méthode TF-IDF (Term Frequency-Inverse Document Frequency) :** Cette méthode est utilisée pour l'évaluation de l'importance des termes dans un texte. L'utilisation de cette méthode nous a permis de déterminer les termes les plus récurrents [25].

Nous avons aussi utilisé la binarisation et l'algorithme Apriori. Grâce à toutes ces techniques, nous avons extrait deux types de règles d'associations : les règles d'association régulières et les règles d'associations de classe.

Pour réaliser notre travail, nous avons utilisés le langage de programmation Python avec la librairie XLRD pour la lecture de fichier Excel et la librairie de traitement de texte Rosette .

Nous avons appliqué notre méthode sur le Data set de la langue arabe qui a été cité dans 29 articles, il contient plusieurs articles de journaux en langue arabe.

TABLEAU 26 EXEMPLE DE RÈGLES D'ASSOCIATIONS

Classes	**Antécédents**	**Conséquents**
Médical	مرض، ممرضة، طبيب	الربو
Politique	الدستور، الحكومة	الدولة
Culture	الحضارة، الأدب	التعليم، التنوع

Les résultats de nos expérimentations sont représentés dans le tableau suivant (Tableau 27):

TABLEAU 27 RÉSULTATS DES EXPÉRIMENTATIONS

Classes	Précision de classification		Réduction des règles	
	Avant QM	Après QM	Avant QM	Après QM
Culture	66 %	73 %	193	33
Finance	93 %	90 %	242	85
Médical	96 %	93 %	168	86
Religion	90 %	68 %	132	44
Sport	96 %	93 %	492	164
Politique	90 %	86 %	120	44
Technologie	96 %	93 %	402	159
Moyenne	88 %	90 %	1749	615

QM : Quine-McCluskey

Conformément aux données présentées dans le tableau 24 ci-dessus, nous avons réussi à réduire de manière significative le nombre de règles à 615, au lieu des 1749 initiales, ce qui représente un taux de réduction de 64,8 %.

Les données présentées dans le tableau 24 montrent les résultats de notre approche de réduction des règles. Pour la première banque de données (Culture), nous avons réussi à réduire le nombre de règles de manière significative, passant de 193 à seulement 33 règles, soit une réduction de 82,93 %. En ce qui concerne la deuxième banque de données (Finance), nous avons observé les réductions suivantes : de 242 règles à 85 (taux de réduction de 64,88 %). Pour la cinquième base de données (Sport), nous avons pu réduire le nombre de règles de 492 à seulement 164 (taux de réduction de 66,67 %). Enfin, de manière générale nous avons pu réduire de manière significative le nombre de règles à 615 au lieu de 1749 (taux de réduction de 64,8 %). Ces résultats démontrent clairement que notre approche peut réduire de manière significative le nombre de règles d'associations sans utiliser de seuil fixe.L'un des points forts de notre approche réside dans le fait qu'il n'est pas nécessaire de définir un seuil .

Les résultats obtenus indiquent clairement que notre approche est capable de réduire de manière significative le nombre de règles d'association sans avoir recours à un seuil fixe.

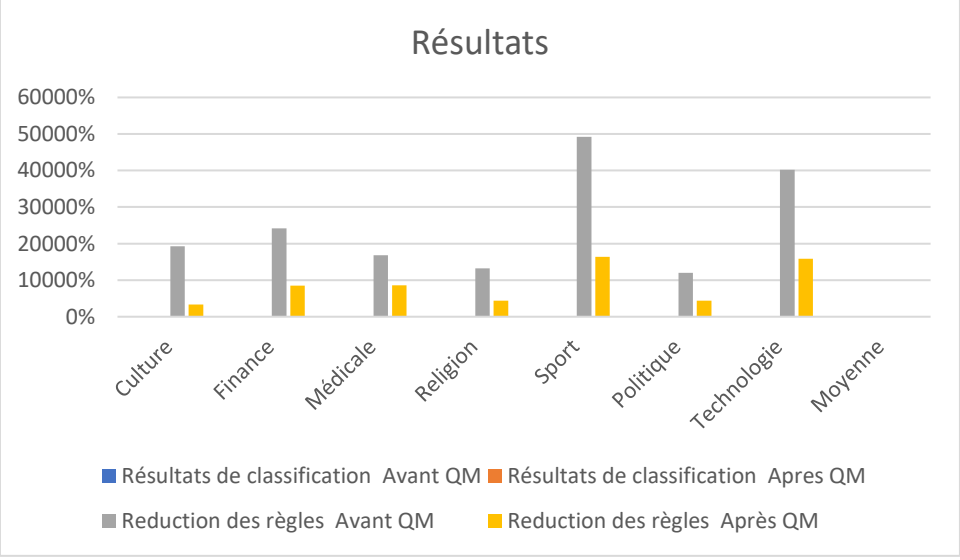


FIGURE 9 RÉSULTATS DE L'APPLICATION DE NOTRE MÉTHODOLOGIE

Le graphique ci-dessous illustre la réduction du nombre de règles d'associations pour chaque base de données.

5.3.2 Étude de la perte d'informations importantes

Dans le tableau 22 ci-dessus, le taux de précision de classification reste sensiblement le même pour la plupart des catégories après le processus de réduction des règles d'association. Dans certains cas, la précision s'est même considérablement améliorée, notamment pour la catégorie Culture, où le taux de précision moyen de classification est passé de 88 % à 90 %. Cette amélioration de la précision du système s'explique par le fait que notre approche permet d'améliorer la qualité de nos données brutes en éliminant les informations redondantes.

Le taux de précision de la classification est calculé comme suit [26] :

$$\text{Taux d'exactitude} = \frac{TP}{(TP+FP)}$$

VP (*True Positive*) : est le nombre d'objets correctement classés. FP (*False Positive*) : est le nombre d'objets classés incorrectement.

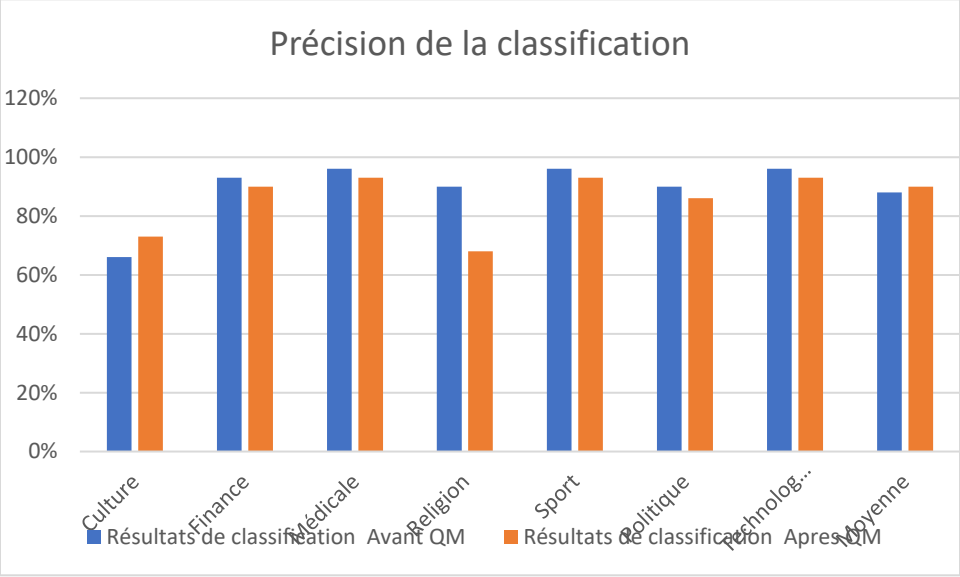


FIGURE 10 TAUX DE PRÉCISION AVANT ET APRÈS QM

Les taux de précision de la classification sont demeurés remarquablement similaires, ce qui démontre que notre approche a réussi à réduire le nombre de règles d'association tout en préservant l'essentiel des informations nécessaires pour obtenir des résultats satisfaisants. Il est important de

noter que notre principal objectif n'était pas d'améliorer la classification en soi, mais plutôt d'utiliser la classification comme un moyen de vérifier si notre approche entraîne une perte d'informations importante.Cette approche a donc permis la réduction du nombre de règles d'une manière importante, ce qui fait gagner de l'espace mémoire, ainsi qu'une amélioration du temps mémoire.

5.4 Résumé des résultats

En examinant les résultats produits par notre méthode de réduction du nombre de règles d'associations, nous avons effectué les observations suivantes :

- Réduction du nombre de règles : Grâce à notre approche, le nombre de règles d'associations a été considérablement réduit, ce qui a entraîné une réduction des besoins en mémoire pour les jeux de données. Nous avons donc réussi à atteindre notre premier objectif de réduction du nombre de règles ;

- Conservation des informations importantes : Nous avons étudié le taux des pertes et des conservations des règles en fonction de leurs scores de précision et nous constatons que notre approche conserve les informations pertinentes pour l'analyse des données. Cette conservation s'est clairement manifestée lors de la comparaison des taux de classification avant et après les réductions, montrant des niveaux de précision similaires.

En résumé, notre approche a permis de réduire considérablement le nombre de règles d'associations tout en préservant les informations nécessaires pour l'analyse des données. Les résultats ont montré que notre méthode maintien des niveaux de précision remarquablement similaires sans fixer de seuil préalable, démontrant ainsi son efficacité dans la réduction du volume de données tout en conservant les informations pertinentes.

5.5 Conclusion

Dans ce chapitre, nous avons présenté nos expérimentations ainsi que les résultats obtenus. Au cours de nos expériences, notre principal objectif était de réduire de manière significative le nombre de règles d'associations. Pour évaluer l'efficacité de nos réductions, nous avons comparé les taux de précision avant et après l'application de la méthode proposée. Cette analyse visait à déterminer si les informations essentielles pour l'analyse des données étaient préservées.

Les résultats ont montré que les taux de précision obtenus après les réductions étaient très similaires à ceux observés avant les réductions, ce qui démontre l'efficacité de notre approche.

En somme, notre approche a réussi à réduire considérablement le nombre de règles d'associations tout en préservant les règles d'association sans avoir à fixer de seuil spécifique.

Chapitre 6 : Conclusion générale

Chapitre 6

Conclusion générale

Les règles d'associations sont de plus en plus utilisées dans plusieurs domaines tels que : le marketing, la classification des documents, la prédiction d'action, la détection des utilisateurs influents dans les réseaux sociaux [27], l'interprétation de l'évaluation du risque de défaillance dans les processus de production continue [28], etc. L'utilisation de règles d'associations a pour but de trouver le plus d'informations pertinentes possible.

Dans ce travail, nous avons présenté la forme des règles d'associations ainsi que leur structure qui est fondée sur un ensemble de conditions qui mènent à un ensemble de conséquents. Plusieurs mécanismes ont été mis en place afin de filtrer et de déduire des règles d'associations pertinentes telles que le support, la confiance et le lift. Nous avons aussi présenté différents algorithmes d'extraction de règles d'associations, à savoir : les algorithmes Apriori, Apriori-TID et FP- Growth, pour lesquels nous avons donné des exemples d'applications ainsi qu'un exemple comparatif entre ces algorithmes, afin de pouvoir déterminer les limites de chacun.

Ce travail nous a permis de mieux comprendre le problème des règles d'associations. En effet, un nombre élevé de règles d'associations peut engendrer un très grand nombre de règles qui sont parfois redondantes ou non pertinentes. C'est une problématique qui a fait l'objet de plusieurs recherches dont l'objectif principal consistait à réduire le nombre de règles d'associations.

Dans ce mémoire, nous avons proposé une approche pour réduire le nombre de règles, tout en conservant les informations importantes et en évitant le problème de la redondance. Nous avons utilisé une méthode de minimisation des fonctions booléennes, qui est la méthode de Quine-McCluskey. Cependant, l'utilisation de cette méthode nécessite la transformation de règles d'associations en fonctions booléennes et donc un modèle de passage des règles d'associations → fonction booléenne. Les annotations utilisées dans ce passage sont : 0 et 1, qui représentent respectivement l'absence et la présence. Nous avons utilisé (-) pour indiquer que l'item était ignoré. Deux approches de réduction ont été proposées, une réduction par les mêmes antécédents et une autre par les mêmes conséquents.

Après avoir appliqué notre méthode, les expérimentations ont démontré que l'utilisation de la méthode de Quine-McClusky réduit considérablement le nombre de règles d'associations, tout en conservant les règles les plus importantes et cela avec des taux de précision intéressants.

L'utilisation de cette méthode nous a permis d'éviter tous les problèmes posés par les seuils de certaines méthodes (telles que minSup, minConf et minLift), qui peuvent exclure des règles d'associations importantes.

Les règles d'associations jouent un rôle crucial en fournissant des détails et des informations importantes. Il est donc nécessaire d'éviter toute perte possible, car une perte importante d'informations peut entraîner une baisse considérable de précision.

Les expérimentations ont montré que la méthode Quine-McCluskey est à la fois efficace et simple à utiliser. Elle réduit le nombre de règles, tout en préservant les règles essentielles, sans nécessiter de paramètres auxiliaires particuliers. En raison de son fonctionnement binaire, elle peut être utilisée pour n'importe quel type de données ; tout ce qu'il faut faire est de traduire les données et laisser la méthode s'occuper de la minimisation.

Bibliographie :

[1]: Ashrafi, M. Z., Taniar, D., & Smith, K. (2007). Redundant association rules reduction techniques. International Journal of Business Intelligence and Data Mining, 2, 29-63.

[2]: Liu, B., Hsu, W., & Ma, Y. (1998). Integrating classification and association rule mining. In Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining (KDD-98), 80-86. AAAI Press.

[3]: Turčínek, P., & Turčínková, J. (2015). Exploring consumer behavior: Use of association rules. Acta Universitatis Agriculturae et Silviculturae Mendelianae Brunensis, 63, 1031-1042.

[4]: Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules. In Proc. 20th int. conf. very large data bases, VLDB, 487-499. Citeseer.

[5]: Han, J., Pei, J., & Kamber, M. (2011). Data mining: Concepts and techniques (Elsevier).

[6]: Bokhabrine, A., Biskri, I., & Ghazzali, N. (2020). Textual Clustering: Towards a More Efficient Descriptors of Texts. In International Conference on Computational Collective Intelligence, 801-810. Springer.

[7]: Zaki, M. (2000). Generating non-redundant association rules. In Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining 2000, 34-43.

[8]: Turčínek, P., & Turčínková, J. (2015). Exploring consumer behavior: Use of association rules. Acta Universitatis Agriculturae et Silviculturae Mendelianae Brunensis, 63, 1031-1042.

[9]: Lin, W.-Y., Tseng, M.-C., & Su, J.-H. (2002). A confidence-lift support specification for interesting associations mining. In Pacific-Asia Conference on Knowledge Discovery and Data Mining, 148-158. Springer.

[10]: Zeng, Y., Yin, S., Liu, J., & Zhang, M. (2015). Research of improved FP-Growth algorithm in association rules mining. Scientific Programming.

[11]: Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules. In Proc. 20th int. conf. very large data bases, VLDB, 487-499. Citeseer.

[12]: Wu, X., Kumar, V., Quinlan, J. R., Ghosh, J., Yang, Q., Motoda, H., ... & Steinberg, D. (2008). Top 10 algorithms in data mining. Knowledge and information systems, 14(1), 1-37.

[13]: Agrawal, R., Srikant, R., & Vu, Q. H. (1997). Mining association rules with item constraints. In Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data, 67-73. ACM.

[14]: Kaur, P., & Singh, R. (2019). An efficient Apriori-TID algorithm for mining frequent itemsets in transactional databases. International Journal of Grid and High Performance Computing, 11(1), 40-53.

[15]: Han, J., Pei, J., & Kamber, M. (2011). Data mining: Concepts and techniques. Elsevier.

[16]: McCluskey, E. J. (1956). Minimization of Boolean functions. The Bell System Technical Journal, 35(6), 1417-1444.

[17]: Malvino, A. P., & Bates, J. A. (2006). Electronic principles (7th ed.). McGraw-Hill.

[18]: Duşa, A. (2010). A mathematical approach to the boolean minimization problem. Quality & Quantity, 44, 99-113.

[19]: Majumder, A., Chowdhury, B., Mondai, A. J., & Jain, K. (2015). Investigation on Quine McCluskey method: A decimal manipulation based novel approach for the minimization of Boolean function. In 2015 International Conference on Electronic Design, Computer Networks & Automated Verification (EDCAV), 18-22. IEEE.

[20]: Duşa, A. (2010). A mathematical approach to the boolean minimization problem. Quality & Quantity, 44, 99-113.

[21]: Navabi, Z. (2019). Digital design and computer architecture (2nd ed.). Academic Press.

[22]: Dong, X., Hao, F., Zhao, L., & Xu, T. (2020). An efficient method for pruning redundant negative and positive association rules. Neurocomputing, 393, 245-258.

[23]: SANAD: Single-Label Arabic News Articles Dataset for Automatic Text Categorization - Mendeley Data [**SANAD: Single-Label Arabic News Articles Dataset for Automatic Text Categorization - Mendeley Data**].

[24]: https://data.mendeley.com/datasets/57zpx667y9/2

[25]: Lovins, J. B. (1968). Development of a stemming algorithm. Mechanical Translation and Computational Linguistics, 11(1-2), 22-31.

[26]: Manning, C. D., Raghavan, P., & Schütze, H. (2008). Introduction to Information Retrieval. Cambridge University Press.

[27]:Agouti, T. (2022). Graph-based modeling using association rule mining to detect influential users in social networks. Expert Systems with Applications, 202(117436).

[28]: Pohlmeyer, F., Kins, R., Cloppenburg, F., & Gries, T. (2022). Interpretable failure risk assessment for continuous production processes based on association rule mining. Advances in Industrial and Manufacturing Engineering, Volume 5.

ANNEXE : ARTICLE

Bedhouche, L., Koalal, M. A., Ben Ayed, A., & Biskri, I. (2023). Efficient Association Rules Minimization Using a Double-Stage Quine-McCluskey-Based Approach. In Computational Collective Intelligence: 15th International Conference, ICCCI 2023, Budapest, Hungary, September 27–29, 2023, Proceedings (pp. 245-255). https://doi.org/10.1007/978-3-031-41456-5_19 .

Efficient Association Rules Minimization Using a Double-Stage Quine-McCluskey-Based Approach

Lidia Bedhouche, Mohamed Amir Koalal, Alaidine Ben Ayed[0000-0003-0877-1929] and Ismail Biskri[0000-0002-7644-5810]

Université du Québec à Trois-Rivières, Trois-Rivières, Canada
{Bedhouche ; Koalal ; Benayed ; Biskri}@uqtr.ca

Abstract. Association rules are used to identify relationships between variables in large datasets. They help us explain obtained results in a transaction database by uncovering associations between items. One major limitation of association rules is the combinatorial explosion that occurs when the number of constructed rules is enormous. Several state-of-the-art techniques try to define a given threshold to overcome this issue; Rules with scores below previously established thresholds are deleted. Nevertheless, those threshold-based strategies have often implied the loss of salient association rules, which have yet to be scored higher than the established threshold because their context of extraction is not recurrent. Ditto, thresholds are usually established in a quasi-arbitrary way.

This paper proposes a novel Quine-McCluskey-based double-stage approach that effectively reduces the number of association rules. The Quine-McCluskey technique is widely used in simplifying Boolean logic expressions. First, the number of initial association rules is reduced through the Quine-McCluskey technique. Next, we optimize the salient information loss rate. Conducted experiments show that the proposed approach significantly reduces the number of rules while keeping salient information

Keywords: Data mining. Association rules minimization. Quine-McCluskey approach.

1 Introduction

The computing power available on computers nowadays and the massive increase in data volumes urged the emergence of data mining techniques to ensure better decision-making. Data mining has remarkably evolved thanks to artificial intelligence and big-data developments. Wherever there is data, data mining is applicable. Indeed, all transactions generate data in different formats (text, images, etc.). Professionals use data mining techniques to automatically identify and extract the most crucial information and analyze it later.

Association rules, first introduced by Agrawal et Al. in 1933 [2], are a subset of information retrieval techniques. They are usually used to study and analyze databases to discover relationships between elements often used together. The use of association

rules allows extracting the essential information with an explicability effect that provides the information in a more accessible format while avoiding the black box effect of abstract representation and increasing data volume. Nevertheless, this method can generate a large number of association rules, some of which are redundant and irrelevant [1]. The evaluation of association rule extraction algorithms is mainly based on optimality, execution time, completeness, and memory consumption. The main challenge for those algorithms is to reduce the number of association rules while maintaining valuable and essential ones.

In this work, we propose a novel double-stage approach that effectively reduces the number of association rules. The proposed technique uses a minimization of Boolean functions technique, namely the Quine-McCluskey (QM) method [13,14,16]. The latter can be applied to any data type and only requires binarized data. Moreover, no parameters or metrics are required from the user. Our experiments show that the proposed approach significantly reduces the number of rules while keeping the most salient information.

The rest of this paper is broken down as follows: the second section describes related work on association rules. The third one details the QM-based association rules reduction. The fourth section reports and discusses experimental results and the fifth section puts forth conclusions.

2 Related work

2.1 Association rules

Association rules are a data mining technique used to discover patterns in large data sets by examining the relationships between different data elements. These patterns can be used to predict future outcomes and other predictions about specific subsets of the data, such as which customers are likely to buy a particular product.

An association rule is usually presented in the form (If-Then) where:
A(Antecedent) $\rightarrow$ B(Consequent) [3]. The antecedent presents the triggering event whose outcome is the consequent and $\sum(A, B) \geq 1$ [4]. If we take the example of customers buying items A, B, and also item C, the corresponding association rule takes the following form: A, B $\rightarrow$ C. Association rules are extracted from transactions and can have several applications in different domains. Each transaction is a piece of data collected in raw form. Depending on the context, each piece of data consists of elements called items, which can be actions, texts, images, sounds, etc.

Let the transaction T = {$t1$, $t2$, …, ti, …, tn}. The elements $ti \in T$ are the items of the transaction T. An item is an element of a transaction. The association rule mining process aims to reveal hidden relationships between different objects. There is no particular method to target the items to be considered, but there are measures to target the items to be ignored, like Support, Trust, and Lift.

The data is usually presented in a table that groups all the transactions, each one identified with a TID (Transaction Identifier) as shown in the following table:

Table 1. Transactions in a grocery store.

TID	Transactions
1	*A, B, C*
2	*A, D*
3	*E, D*
4	*A, B, D*
5	*A, B, F, D, C*
6	*B, C, F*

To extract an association rule, we first need to identify the frequent items, which are items that, according to a threshold established by the user, appear with a particular frequency. A rule must have at least two elements: an antecedent and a consequent, i.e., at least two items. Therefore, the next step is to look for item-sets. The latter comprises frequent items that appear together in the database. Like frequent items, item-sets must also be frequent according to the threshold given by the user.

The item-set is given by: $I = \{A, B, C, D, E, F\}$, which is more generally represented by: $I = \{i1, i2, \ , in\}$. Therefore, *Transaction1* in our table is $T1 = \{A, B, C\}$. All the item-sets are elements of I. An item set contains at least two elements, and any empty set or set containing only one item is not considered an item set. An item set is said to be frequent, if and only if: Support $\geq$ minSup. minSup is the minimum support predefined by the user.

An A $\rightarrow$ B rule is considered a quality rule according to many measures, such as Support, Confidence, and Lift. The number of generated rules and their relevance depend on the measures and the minimum thresholds initially set. The support represents the frequency of an item, an item-set, or a given rule in the database and is given by the number of transactions where A and B co-occur compared to the number of transactions. The support of a given rule is defined as described in Equation 1 [5]:

$$Sup(A \rightarrow B) = NH(A \cup B) / TNT \quad (1)$$

$NH(A \cup B)$ and TNT in Equation 1 are respectively the *Number of Hits of* $(A \cup B)$ and the *Total Number of Transactions*.

The Support for a given item A is given as follows:

$$Sup(A) = NH(A) / TNT \quad (2)$$

The support of an item-set is given as follows:

$$Sup(A, B) = NH(A \cup B) / TNT \quad (3)$$

If we take the example of the rule $\{F, C \rightarrow B\}$, the support of this rule is:

$$Sup(F, C \rightarrow B) = NH(F, C \cup B) / TNT = 2/6 = 33\% \quad (4)$$

NH and *TNT* in Equations 2, 3, and 4 are the same as in Equation 1.

The confidence of an association rule represents the frequency of an item in the database. In other words, it represents the conditional relative frequency of B given A, i.e., the probability that B will be purchased if A is also purchased [5]:

$$Conf(A \rightarrow B) = Sup(A \cup B) / Sup(A) \quad (5)$$

The Confidence of the rule: $\{F, C \rightarrow B\}$ is:

$$Conf(F, C \rightarrow B) = Sup((F, C \rightarrow B) \cup B) / Sup(F, C) \quad (6)$$

For the below example; $Sup((F, C \rightarrow B) \cup B) = 2/6$ and $Sup(F, C) = 2/6$. So, $Conf(F, C \rightarrow B) = 1$.

The lift of an association rule stands for the relationship between the confidence of a rule and the expected confidence, so it is the ratio between the confidence of the rule ($A \rightarrow B$) and its consequent (B) [5]:

$$Lift(A \rightarrow B) = Conf(A \rightarrow B) / Sup(B) \quad (7)$$

Therefore, the Lift of the rule: $\{F, C \rightarrow B\}$ is:

$$Lift(F, C \rightarrow B) = Conf(F, C \rightarrow B) / Sup(B) = 1/(4/6) = 1.5 \quad (8)$$

Many extraction approaches of association rules have been proposed since the early thirteenth. The most standard ones are the FP-Growth [2] and the Apriori algorithm [6]. The previously proposed association rules approaches are greedy regarding memory/processor consumption and execution time. These algorithms can generate many redundant and irrelevant rules.

- Data preparation: First, the data is preprocessed. Only relevant data for the coming two steps is kept.
- The search for frequent items: The goal is to find the most recurrent items. This step is costly in terms of execution time, mainly when we deal with a large amount of data.
- Association rules generation: The last step consists in finding rules with $support \geq minSup$, $Confidence \geq minConf$ and $Lift \geq minLift$ where *min-*

Sup, *minConf*, and *minLift* are set by the user and represent the minimum values that a rule must satisfy for it to be accepted.

To further understand the third step, which aims to generate a set of association rules, we give the following example using the Apriori algorithm:

```
Input : a minimum support and a transaction data set.
Output: generation of frequent item sets.
1. Mi=∅, i = 0;
2. C1= all 1-Item sets of the data set
3. 1=all frequent item sets of the data set
   While ( Mi is non-empty)
1. Ci+1= Candidate-gen(Li);
2. Fi+1=all frequent item sets of Ci+1;
3. i++;
4. Return ⋃ Mi;
   End
```

Fig. 1. Apriori algorithm.

Let $E = \{A, B, C, D\}$ be a set of elements and $T = \{T1, T2, T3, T4, T5\}$ a set of transactions. We represent the elements and the transactions as follows:

Table 2. Transactions in a grocery store.

TID	Transactions
t1	*A, B, C*
t2	*A, C, D*
t3	*B, C, D*
t4	*A, D, E*
t5	*B, C, E*

By applying the Apriori algorithm to the transactions in Table 2, we obtain the following results: Notice that the confidence scores for rules R1 and R5 are computed, respectively, as follows: *Conf*(R1) = *Sup*(B,C,E) / Sup(B,C) = 2/3 = 66.66% and *Conf*(R5) = *Sup*(B,C,E) / Supp(E) = 2/4 = 50%.

Table 3. Transactions in a grocery store.

Rule name	Mathemathical formalism	Confidence	result
R1	{B, C} → ({B, C, E} − {B, C})	66.66%	Retained
R2	{B, E} → ({B, C, E} − {B, E})	66.66%	Retained
R3	{C, E} → ({B, C, E} − {C, E})	66.66%	Retained
R4	{B} → ({B, C, E} − {B})	66.66%	Retained
R5	{C} → ({B, C, E} − {C})	50%	Rejected
R6	{E} → ({B, C, E} − {E})	50%	Rejected

As we can see, the algorithm generates many association rules (like all algorithms related to support and trust), or several rules can be redundant and irrelevant, occupying a huge memory space, especially when we have a large dataset, which makes the execution time very long.

2.2 The QM approach

The QM method is widely used for the minimization of Boolean functions. It allows us to determine the minimal form of the functions in a deterministic way. The method was developed by [13,14] and extended by [15]. It starts with a unique presence/absence combination in the truth table. The method minimizes pairs of product combinations whose variables appear only once in the True or False form (Example: xy′ z(101)), called minterms, which must have a different bit [8].

QM uses three annotations: '1', representing the true value, '0' for false, and '-' for ignored variables. The main objective is to find the non-combinable elements that represent the prime implicants. If we consider the following expression: E = W X'Y Z' + W XY 'Z ' + W XY 'Z (in binary: E = 1010 + 1100 + 1101), W X'Y Z' cannot be combined with any other term of the expression E. Finally, the method looks for the essential prime implicants. A prime implicant is essential if it contains a minterm that is not included in any other prime implicant [9].

Some association rules may share the same consequents or antecedents. Example: abc → D, ab → D, ac → D. Since the rules share the same consequent D, their antecedents can be rewritten in Boolean algebra: abc+abc′ +ab'c where (') represents negation. Thus, we have our antecedents in minterms abc → D, abc′ → D, ab'c → D.

QM phases are:

1. The switch from the canonical form to the binary form.
2. The classification and comparison of the terms according to their weights (the number of 1's), i.e., comparison of the terms of the group 'i' with the terms of the group 'i+1'. Each different bit is replaced by a '-'.
3. The search for the prime implicants (the non-combined terms)
4. The search for the essential prime implicants.

3 Methodology

3.1 System Architecture

The proposed approach works according to the following steps:

5. The system receives a set "E" of association rules as input.
6. The number of association rules is reduced by applying QM to obtain a new "E" set.
7. The system uses the set "E" as a knowledge base to classify a set of test data T and records the results. It then uses set "E'" as a knowledge base to classify the same test data set T and records the new results. The two results are compared to deduce whether a loss of important information has occurred.

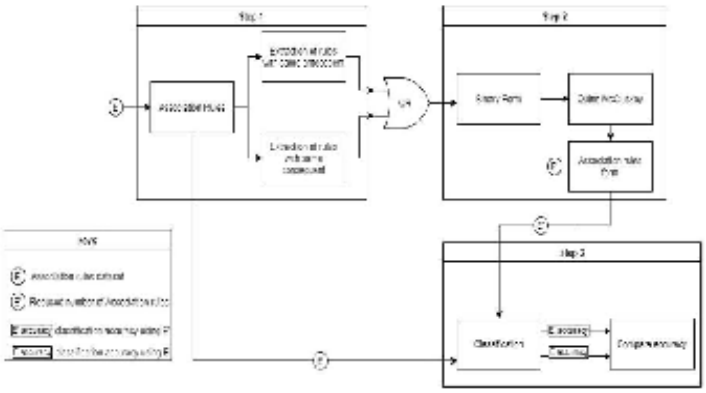


Fig. 2. System architecture.

3.2 Our method for applying the QM on Association Rules

Before using a Boolean function minimization method, such as the QM method, on association rules, we need to define a relationship between the binary form and the association rules. This relationship is a gap between the form of the association rules and the binary form. Since the input to QM is in boolean form, we take inspiration from negative association rules [10], where the negation of an element represents its absence. Example: $AB' \rightarrow C$ is equal to $A \rightarrow C$ since B' represents the absence of B. Now we have a relationship between the form of the association rules and the Boolean form. Once the rules are in Boolean form, we write them in the binary form where negation represents 0. We now have our second relationship between the binary form and the Boolean form. Next, we minimize the rules with the same consequents together, and the rules with the same antecedents are minimized together to keep the meaning of the rules. Finally, we will have to return to the form of the association rules. We can change the form of a rule as needed in any direction without losing it,

e.g., ab → d, ab → d, ef → h, ef → hg. For this purpose, we have four functions for changing the presentation of association rules, namely: *ARtoNAR* for switching association rules to negative association rules. *NARtoBinary*, for the passage of negative association rules to the binary form. Finally, respectively *NARtoAR* and *BinaryToNar* do the opposite operations. All functions apply to rules with the same antecedent or consequent. The element in common between the rules (consequent or antecedent) is not affected by the previous functions, as shown in the following examples:

Function for switching from Association rules to negative Association rules ;

ARtoNAR([abc, ac]) = [abc, ab′ → c] (S2)

Function to switch from negative association rules to binary ;

NARtoBinary([ab → c, ab′ → c]) = [11, 10] → c

Function for switching from binary to negative association rules ;

BinarytoNAR([11, 10] → c) = [ab → c, ab′ → c]

Function of switching from negative to Method of association rules ;

NARtoAR([ab → c, ab′ → c]) = [ab → c, a → c]

By using our approach to change the form of association rules, it is possible to minimize the rules as if they were Boolean terms. As an example of minimization, consider the rules below:

ab → m, ab → mn, ab → e, b → e, df → e, f → e. (9)

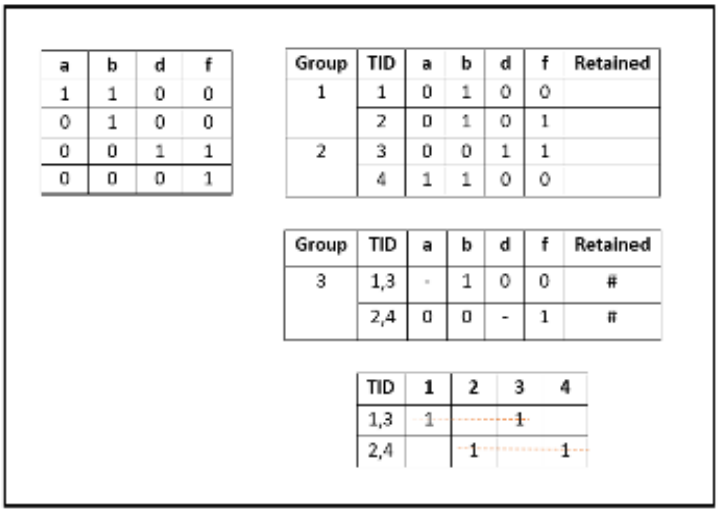
If we start with the rules sharing the same consequent e, we can write:

S1 = ab + b + df + f = S2 = abd'f' + a'bd'f' + a'b'df + a'b'd'f (10)

We can now write S2 in binary form:

abd'f' + a'bd'f' + a'b'df + a'b'd'f] ⇒ [1100 + 0100 + 0011 + 0001] (11)

The application of the QM method on S2 is presented in Figure 3.



a	b	d	f
1	1	0	0
0	1	0	0
0	0	1	1
0	0	0	1

Group	TID	a	b	d	f	Retained
1	1	0	1	0	0	
	2	0	1	0	1	
2	3	0	0	1	1	
	4	1	1	0	0	

Group	TID	a	b	d	f	Retained
3	1,3	-	1	0	0	#
	2,4	0	0	-	1	#

TID	1	2	3	4
1,3	1		1	
2,4		1		1

Fig. 3. Application of the QM method on S2.

The result S ′ = −100 + 00−1. We can rewrite the result in the form of negative association rules as follows:

$$[-100 + 00-1] \Rightarrow [bd'f' + a'b'f] \quad (12)$$

Finally, we can rewrite as association rules: b → e, f → e.

We can, also apply the same process on the rules sharing the same antecedent ab :

$$ab \rightarrow m,\ ab \rightarrow mn,\ ab \rightarrow e \quad (13)$$

We obtain:

$$S1 = m + mn + e = S2 = mn'e\ ' + mne' + m'n'e \quad (14)$$

The result of the QM on S2 will be me′ + m′n ′ e. We can furthermore rewrite it as: ab → m, ab → e.

3.3 Redundancy elimination after system reduction

After reducing the number of rules, the knowledge base may be overwhelmed with redundant rules. For example: $\{ab \rightarrow F, a \rightarrow F\}$ would infer $a \rightarrow F$, the latter induces $a \rightarrow F$ (i). Symmetrically, $ac \rightarrow F$, $a \rightarrow F$ would infer $a \rightarrow F$. The latter induces $a \rightarrow F$ (ii). Rules (i) and (ii) are redundant. Next, only one representative rule $(a \rightarrow F)$ is kept for each duplicate set of rules.

4 Experimental results

Processing data with classic data reduction techniques usually results in salient information loss. To validate the proposed system, especially its resilience to information loss, we use E', a set of association rules obtained by reducing E (an initial set of rules) in a common classification task F. Furthermore, we compare obtained results in both scenarios (before and after association rules reduction). The intuition behind this validation approach is that if the performance of the classifications (accuracy rate) decreases, we admit that important information has been lost. Otherwise, we can admit that the association-rules reduction process was successfully performed without salient information loss.

Table 4. A sample of association rules extracted from articles in [11].

Class	Antecedent	Consequent
Medical	مرض، ممرضة، طبيب	الربو
Politics	الستور، الحكومة	الدولة
Culture	الحضارة، الأدب	الفلوجة، الفرع

Table 5. Classification precision and number of association rules before and after the application of our QM-based association rules reduction.

Class	Classification precision		Number of association rules	
	Before QM	After QM	Before QM	After QM
Culture	66%	73%	193	33
Finance	93%	90%	242	85
Medical	96%	93%	168	86
Religion	90%	68%	132	44
Sport	96%	93%	492	164
Politics	90%	86%	120	44
Technology	96%	93%	402	159
Average	88%	90%	1749	615

An Arabic news articles dataset named SANAD (Single-Label Arabic News Articles Dataset for Automatic Text Categorization) [11] is experimented to assess the proposed approach's performance. SANAD includes 190.000 Arabic news articles belonging to seven categories: Culture, Finance, Medical, Politics, Religion, Sports,

and Tech. All the news articles were collected from three popular news websites: AlKhaleej, AlArabiya, and Akhbarona. Table 4 illustrates a sample of extracted association rules from news articles belonging to the Medical, Sport, and Culture categories.

The results illustrated in Table 5 show that we could significantly reduce the number of rules from 1749 to 615. The achieved reduction rate is 64.8%. A strong point of our approach is the no need to set an empirical threshold. Moreover, the classification accuracy rate remains approximately the same for most classes after the association-rules reduction process. Even there are some scenarios where the precision rate was remarkably improved. It is the case for the culture class. The average classification accuracy rate is improved from 88% to 90%. This improvement in system accuracy can be explained by the fact that our approach improves the quality of our crud data by getting rid of redundant information.

5 Conclusion

Association rules have widely demonstrated their usefulness in different domains, such as detecting influential users in social networks [16], interpreting failure risk assessment in continuous production processes [17], etc. Indeed, its formalism of antecedent(s)/consequent(s) is easy to understand for humans. Also, it can be processed by machines. Several researchers have introduced association rules-based information extraction techniques. In this work, we have addressed this problem while trying to keep the most salient information. We used a technique of Boolean function minimization, namely the QM method. In order to use the QM method, we had to determine new annotations, allowing us to keep the readability of the association rules and to allow the minimization of the rules. If the QM method has as input a boolean function, then our rules had to transform this form. We were inspired by negative association rules, where the negation "¬" of a variable means its absence. We had to figure out how to express the association rules in a binary format. We annotated absence by 0 and presence by 1. For minimization, we considered the dash "-" to say that the element is ignored, as it has no direct effect on the result. Conducted experiments showed that the proposed method could significantly reduce the number of association rules without any information loss. Quite the opposite, the overall precision rate of classification was improved by 2% due to eliminating unnecessary, redundant information.

Acknowledgements. The authors would like to thank Natural Sciences and Engineering Research Council of Canada for financing this work.

References

1. Ashrafi, Mafruz Zaman, David Taniar, and Kate Smith.: 'Redundant association rules reduction techniques. International Journal of Business Intelligence and Data Mining, 29-63 (2007).
2. Agrawal, Rakesh, and Ramakrishnan Srikant.: Fast algorithms for mining association rules. In Proc. 20th int. conf. very large data bases, VLDB, 487-99. Citeseer (1994).
3. Zaki, Mohammed.: Generating non-redundant association rules. In Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining 2000, 34–43.
4. Turčínek, Pavel, and Jana Turčínková.: 'Exploring consumer behavior: Use of association rules'. Acta Universitatis Agriculturae et Silviculturae Mendelianae Brunensis, 63: 1031-42.
5. Lin, Wen-Yang, Ming-Cheng Tseng, and Ja-Hwung Su.:"A confidence-lift support specification for interesting associations mining." In Pacific-Asia Conference on Knowledge Discovery and Data Mining, 148-58. Springer (2002).
6. Han, Jiawei, Jian Pei, and Micheline Kamber.: Data mining: concepts and techniques (Elsevier). (2011)
7. Zeng, Yi, Shiqun Yin, Jiangyue Liu, and Miao Zhang.: 'Research of improved FPGrowth algorithm in association rules mining'. Scientific Programming (2015).
8. Staneva, Liliya Anestieva.: 'Minimising using the Method of Quine-McCluskey with Generalised Nets'.In: 29th Annual Conference of the European Association for Education in Electrical and Information Engineering (EAEEIE), 1-3. IEEE.
9. Majumder, Alak, Barnali Chowdhury, Abir J Mondai, and Kunj Jain.: ' Investigation on Quine McCluskey method: A decimal manipulation based novel approach for the minimization of Boolean function. '. In: International Conference on Electronic Design, Computer Networks Automated Verification (EDCAV), 18-22. IEEE. (2015)
10. Dong, Xiangjun, Feng Hao, Long Zhao, and Tiantian Xu.: 'An efficient method for pruning redundant negative and positive association rules '. Neurocomputing, 393: 245-58. (2020)
11. SANAD, https://data.mendeley.com/datasets/57zpx667y9/2
12. El Bazzi, Mohamed Salim, Taher Zaki, Driss Mammass, and Abdelatif Ennaji.: 'Automatic Indexing of Arabic Texts: State of the Art'. pp. 1–2. Electronic Journal of Information Technology. (2016).
13. Willard V. Quine.: 'The Problem of Simplifying Truth Functions'. The American Mathematical Monthly, 59(8), 521–531.(1952).
14. Willard V. Quine.: 'A Way to Simplify Truth Functions'. The American Mathematical Monthly, 62(9), 627–631. (1955).
15. McCluskey, Edward Joseph Jr.: 'Minimization of Boolean Functions'. Bell System Technical Journal. 35 (6): 1417–1444. (1956).
16. Tarik Agouti.: 'Graph-based modeling using association rule mining to detect influential users in social networks'. Expert Systems with Applications, 202(117436). (2022).
17. Florian Pohlmeyer, Ruben Kins, Frederik Cloppenburg and Thomas Gries .: 'Interpretable failure risk assessment for continuous production processes based on association rule mining'. Advances in Industrial and Manufacturing Engineering, Volume 5. (2022).