جامعة السلطان مولاي سليمان
ⵜⴰⵙⴷⴰⵡⵉⵜ ⵙⵙⵓⵍⵟⴰⵏ ⵎⵓⵍⴰⵢ ⵙⵍⵉⵎⴰⵏ
Université Sultan Moulay Slimane

Centre d'Études Doctorales : Sciences et Techniques

Formation Doctorale : Mathématiques et Physiques Appliquées

THÈSE

Présentée par

Youssef FAKIR

Pour l'obtention du grade de
DOCTEUR

Discipline : Informatique

***Spécialité :* Informatique**

Extraction des Règles d'Association et Prédiction du Diabète par les Algorithmes de Machine Learning

Soutenue le Samedi 24 Juin 2023 devant la commission d'examen :

Mr Said SAFI	Professeur, Université Sultan Moulay Slimane, FP Béni-Mellal, Maroc	Président
Mr Mustapha OUJAOURA	Professeur, Université Cadi Ayyad, ENSA, Safi	Rapporteur
Mr Mohamed BASLAM	Professeur, Université Sultan Moulay Slimane, F.S.T. Béni-Mellal, Maroc	Rapporteur
Mr Yousef EL MOURABIT	Professeur, Université Sultan Moulay Slimane, F.S.T. Béni-Mellal, Maroc	Rapporteur
Mr Hicham ZOUGAGH	Professeur, Université Sultan Moulay Slimane, F.S.T. Béni-Mellal, Maroc	Examinateur
Mr. Rachid EL AYACHI	Professeur, Université Sultan Moulay Slimane, F.S.T. Béni-Mellal, Maroc	Directeur de Thèse

Dédicace

Je dédie ce mémoire à mes parents pour leur grand amour, leur sacrifice et leur tendresse. Je le dédie également à ma sœur, mes amis et mes proches.
Qu'ils trouvent ici le témoignage de ma gratitude et de mon respect.

Youssef FAKIR.

Remerciements

Tout d'abord, je remercie Dieu le tout puissant pour ses faveurs et sa gratitude. Ensuite, je tiens à exprimer mes remerciements pour tous ceux qui m'ont aidé et encouragé à accomplir ce travail. En particulier, j'adresse mes remerciements à:

Mr. **Said SAFI**, Professeur à la Faculté Polydisciplinaire de Béni Mellal pour avoir accepté de présider le jury.

Mr. **Mustapha OUJAOURA**, Professeur à l'Ecole Nationale des Sciences Appliquées de Safi, Université Cadi Ayyad pour avoir accepté de rapporter ce travail. Je le remercie pour l'attention avec laquelle il a lu et évalué ce travail.

Mr. **Mohamed BASLAM**, Professeur à la Faculté des Sciences et Techniques, Université Sultan Moulay Slimane, pour avoir accepté de rapporter ce travail. Je le remercie pour l'attention avec laquelle il a lu et évalué ce travail.

Mr. **Yousef EL MOURABIT**, Professeur à la Faculté des Sciences et Techniques, Université Sultan Moulay Slimane, pour avoir accepté de rapporter ce travail. Je le remercie pour l'attention avec laquelle il a lu et évalué ce travail.

Mr. **Hicham ZOUGAGH**, Professeur à la Faculté des Sciences et Techniques, Université Sultan Moulay Slimane, pour avoir accepté d'examiner ce travail. Je le remercie pour l'attention avec laquelle il a lu et évalué ce travail.

Mr. **Rachid EL AYACHI**, Professeur à la Faculté des Sciences et Techniques, Directeur de ce mémoire, pour m'avoir accepté au sein du laboratoire de Traitement de l'Information et Aide à la Décision (TIAD). J'ai spécialement apprécié ses qualités humaines indéniables, sa disponibilité et ses précieux conseils. Qu'il trouve ici l'expression de mon respect.

Enfin, j'adresse un grand merci pour tous les membres du département d'Informatique de la Faculté des Sciences et Techniques de Béni Mellal. Je remercie également tous mes ami(e)s et tous ceux qui m'ont aidé tout au long de mon parcours.

Publications

1. Youssef FAKIR, Abdelfatah Maarouf and Rachid ELAYACHI, Mining frequents itemset and association rules in diabetic dataset, Lecture Notes in Business Information Processing, Volume 449, 2022. https://www.springer.com/series/7911
2. Youssef FAKIR, Abdelmotalib NAOUM, Analysis of Decision Tree Algorithms for Diabetes Prediction, Lecture Notes in Business Information Processing, Volume 449, 2022. https://www.springer.com/series/7911.
3. Youssef FAKIR, Chaima AHLE TOUATE, Rachid ELAYACHI & Mohamed FAKIR, Closed Frequent Itemsets Mining Based on It-Tree, Journal Of Medical Informatics And Decision Making, Vol-1 Issue 2, 2021.
4. Youssef FAKIR, Chaima ESAILI, Mohamed FAKIR & Rachid ELAYACHI, Extraction of itemsets frequents, International Journal of Mathematics Research. ISSN 0976-5840 Volume 12, Number 1 (2020), pp. 23-32
5. Youssef FAKIR, Meryem ELMOUHI, Rachid ELAYACHI & Mohamed FAKIR, A Comparative Study between Relim and SaM Algorithms, International Journal of Computer Science and Information Security (IJCSIS), Vol. 18, No. 5, May 2020.
6. Youssef FAKIR, Rachid El AYACHI & Mohamed FAKIR, Mining Frequent Pattern by Titanic and FP-Tree algorithms, International Journal of Scientific Research in Computer Science, Engineering and Information Technology, Volume 6, Issue 5, pp : 208-215, September-October-2020
7. Youssef FAKIR, Rachid El AYACHI & Mohamed FAKIR, Frequent Pattern mining, International Journal of Scientific Research in Computer Science, Engineering and Information Technology, Volume 6, Issue 4, 2020

Communications

1. Youssef FAKIR and Rachid ELAYACHI, Association Rules Mining in Diabetic Dataset Using Multiple Item Support Tree and FP-Growth Based Algorithms, ICBASET 2022: International Conference on Basic Sciences, Engineering and Technology, August 25 - 28, 2022 in İstanbul, Turkey.
2. Youssef FAKIR, Rachid ELAYACHI, Extraction of Frequent itemset mining on spark, ICBASET 2022 : International Conference on Basic Sciences, Engineering and Technology, August 25 - 28, 2022 in İstanbul, Turkey.
3. Youssef FAKIR, Abdelfatah Maarouf and Rachid ELAYACHI, Mining frequents itemset and association rules in diabetic dataset, the 7th international conference on business intelligence, Khouribga, 26-28 May 2022.
4. Youssef FAKIR, Abdelmotalib NAOUM, Analysis of Decision Tree Algorithms for Diabetes Prediction, the 7th international conference on business intelligence, Khouribga, 26-28 May 2022.
5. Youssef FAKIR & Rachid ELAYACHI, Algorithms for Association Rule Mining : a comparative study ; International Conference on Mathematics & Data Science (ICMDS-2021) Khouribga, Morocco, 28-30 October, 2021.

Résumé

La découverte de connaissances à partir de bases de données médicales est importante pour établir un diagnostic médical efficace. Le but de l'exploration de données est d'extraire des informations de la base de données et de générer une description claire et compréhensible des modèles.

Le travail présenté avait un double objectif. Le premier est d'appliquer les outils de clustering et de classification à notre base de données. Le second objectif était d'identifier les variables ayant le plus d'impact sur les patients diabétiques à partir des règles d'association extraites, suite à l'extraction des motifs fréquents par un ensemble d'algorithmes que nous comparons les uns aux autres. Nous avons d'abord converti les attributs numériques en une forme catégorique. Des algorithmes basés sur l'algorithme Apriori ont été utilisés pour générer des règles sur les données relatives au diabète des Indiens Pima, afin de selectionner les algorithmes les plus performants. L'ensemble de données a été extrait du référentiel d'apprentissage automatique UCI contenant un total d'instances 768 et 8 attributs numériques. Nous avons découvert que les étapes de prétraitement, souvent négligées dans la découverte des connaissances sont les éléments les plus critiques pour déterminer le succès d'une application d'exploration de données. Enfin, nous avons généré les règles d'association qui sont utiles pour identifier des associations générales dans les données, afin de comprendre les relations entre les champs mesurés et le fait que le patient développe ou non un diabète. Pour remedier aux limites des algorithmes sur les données massives, nous avons adopté la plateform Spark afin de réduire le temps d'execution. Nous avons appliqué trois algorithmes (YAFIM, DFPS et R-APRIORI) sur une base de données des patients diabétiques de 253680 instances.

En utilisant les motifs extraits de la base de données PIMA, la prédiction du diabète est effectuée à l'aide des algorithmes d'apprentissage automatique tels que Forêts aléatoires, Decision Tree, les réseaux de neuronnes, Naive Bayes et K-plus proche voisins.

Les performances de chaque algorithme sont analysées pour déterminer celui qui présente la meilleure exactitude, précision, sensibilité et spécificité. Nous avons constaté que l'algorithme Forêts aléatoires fonctionne bien que les autres algorithmes avec une exactitude de 88.31%.

Mots clés : Dataminng, motifs, Apriori, FP-Growth, Charm, Eclat, CPF-Growth, Yafim, R-Apriori, DFPS, Régles d'association, Algorithmes de Machine learning, Prédiction du Diabète.

Abstract

Knowledge discovery from medical databases is important for effective medical diagnosis. The purpose of data mining is to extract information from the database and generate a clear and understandable description of patterns.

The work presented had a double objective. The first was to apply the clustering and classification methods on our database. The second objective was to identify the variables with the greatest impact on diabetic patients on the basis of the association rules extracted, following the extraction of frequent patterns by a set of algorithms, which we compare with each other. We first converted the numeric attributes to a categorical form. Algorithms based on the Apriori algorithm were used to generate rules on Pima Indian diabetes data in order to select the best performing algorithms. The dataset was extracted from the UCI machine learning repository containing a total of 768 instances and 8 numeric attributes. We've found that the pre-processing stages, often overlooked in knowledge discovery, are the most critical elements in determining the success of a data mining application. Finally, we generated the association rules which are useful to identify general associations in the data, to understand the relationship between the measured fields whether or not the patient develops diabetes.

To remedy the weaknesses of the algorithms on massive data, we used the Spark platform to reduce the execution time. We proposed three algorithms: YAFIM, DFPS and R-APRIORI by using a database of diabetic patients of 253680 instances.

By using the itemsets extracted in the dataset PIMA, diabetic prediction is done using Machine Learning algorithms such as Random Forest, Decision Tree, Neural Network, Naive Bayes and KNN.

The performance of each algorithm is analyzed to determine the one with the best accuracy, precision, sensitivity, and specificity. It can be conducted that the Random Forest model works well than the others algorithms with a curracy of 88.31%.

Key Words: Dataminng, itemsets, Apriori, FP-Growth, Charm, Eclat, CPF-Growth, Yafim, R-Apriori, DFPS, Association rules, Machine learning algorithms, Diabetic prediction.

Table des matières

Liste des figures

Liste des tableaux

Liste des acronymes

BD : Base de Données

Bd+ : Bordure positive

Bd- : Bordure négative

CHARM : Close Association Rule Mining avec un H sans correspondance

CFI-Tree : Closed Frequent Item-Tree

DIC : Dynamic Itemset

DEclat : Diffset Eclat

DFPS : Distributed FP-Growth based Spark

DT : Arbre de décision

ECD : Extraction des Connaissances dans les bases de Données

Eclat : Equivalence CLAss Transformation

FP-Growth : Frequent Pattern Growth

HDFS : Hadoop Distributed File System

Hmine : Hyper-Structure Mining of Frequent Patterns in Large Databases

K-NN : K-plus proches voisins

LCM : Linear time Closed itemset Miner

MAFIA : Merging Adaptive Finite Intervals And is more than a clique

MF : Motif Frequent

MRM : Motifs Rares Minimaux

NB : Naive Bayes

PFP : Parallel FP-Growth

ReLim : Elimination Recursive

RDD : Resilent Distributed Dataset

Rapriori : Reduced Apriori

RF : Random Forest

SaM : Split and Merge

SVM : Support à vecteur machine

TID : Tuple IDentifier

YAFIM : Yet Another Frequent Itemset Mining

Introduction générale

1. Contexte

Data mining est l'un des plus importants domaines de nos jours dû à la vélocité des données et leurs exploitations pour des fins d'optimisations de gain d'informations cachés. L'idée de base de l'exploration des données est d'extraire les connaissances utiles, cachées à partir de données disponibles. Ces connaissances extraites peuvent être sous forme de modèles, règles de décision et concepts qui sont utiles et compréhensibles.

Les règles d'associations sont l'une des meilleures méthodes d'extraction de connaissances à partir des données. Elles se représentent sous une forme de motif = Antécédent → Conséquent où l'antécédent et le conséquent sont des items. L'extraction des règles d'association procède d'abord par l'extraction des itemsets fréquents et ensuite la génération des règles d'association à partir des itemsets.

En revue de la littérature, la majorité des algorithmes utilisés pour l'extraction des règles d'association génèrent un temps de réponse très élevé et un grand nombre de règles d'associations du coup des mesures de qualité de règles d'associations interviennent pour réduire leurs nombres. De plus que ces algorithmes ont un grand temps de réponse, les données massives exigent un temps de réponse aussi élevé ce qui valorisent l'utilisation des outils de traitement des données massives qui permettent une implémentation distribuée des algorithmes d'extraction de règles d'association.

En effet, le Big Data est défini comme étant un volume élevé, une vitesse élevée et une grande variété de données qui exigent des formes d'information innovantes et rentables. L'objectif de l'analyse des données massives est d'obtenir des informations précieuses qui peuvent aider à la prise de décision. De nombreux outils ont été développés pour le traitement et l'analyse des données.

2. Objectifs

La plupart des services de santé, tels que les services ambulatoires, les hôpitaux et les cliniques utilisent des systèmes d'information pour stocker et gérer les données de leurs patients. Bien que ces systèmes d'information accumulent d'énormes quantités de données sous différentes formes (chiffres, textes, images, etc.), les transformer en informations utiles permettant de prendre des décisions médicales importantes n'est pas une tâche facile pour un praticien de la santé. Dans ce contexte, l'utilisation de techniques d'exploration de données devient inévitable pour les chercheurs d'extraire des informations utiles à partir des bases de données médicales.

Par ailleurs, ce présent travail abordera dans une première étape l'extraction des motifs fréquents dans le domaine médical, plus spécifiquement les données des patients diabétiques, Il s'agit d'extraire les itemsets fréquents qui mènent à la génération des règles d'association. En fait, certains algorithmes qui servent à l'extraction des règles d'association génèrent des règles redondantes et nombreuses ce qui déroute la prise de décision. En faisant référence à cette problématique, on peut citer les objectifs de cette recherche comme suit :

- Comparaison et implémentation distribuée de FP-Growth et Apriori afin d'avoir le plus optimal au niveau de temps d'exécution.
- Conception d'une approche de prétraitement de données afin d'éviter les redondances dans les règles extraites et améliorer leurs pertinences.
- Intégration des algorithmes d'extractions des règles d'association Apriori et FP- Growth dans le contexte de Big Data pour améliorer le temps de réponse en se basant sur les outils de calculs distribués.
- Evaluer et diminuer le temps de réponse d'extraction des règles d'associations dans une large dataset en se basant sur des méthodes de clustering et de visualisation.
- Sélection des règles d'association fiables en se basant sur les métriques de précision des règles extraites en total.

Dans la deuxième étape des algorithmes de machine learning ont été proposés afin de prédire le diabète en l'occrence les arbres de décision (DT), les forêts aléatoires, les réseaux de neurones, le K-plus proche vosin et les vecteurs à support machine (SVM).

3. Structure de la mémoire

En plus de l'introduction et dela conclusion, ce mémoire est structuré en quatre chapitres :

- Dans le premier chapitre, nous présentons la définition, les tâches et les techniques de Data Mining. Nous présentons par la suite les concepts et les notions de base des techniques des règles d'association et le processus complet de leurs extractions et les différents algorithmes d'extraction des règles d'association. Les algorithmes les plus pratiques sont décrits.
- Le deuxième chapitre est consacré à la première contribution. Après un prétraitement, un certain nombre d'algorithmes d'extraction de règles d'association parmi les plus performants sont comparés : Apriori, FP-Growth, CFP-Growth, Charm et Eclat.

  Le dataset utilisé provient d'une enquête de la population indienne des femmes (768 enregistrements) en Arisona, USA (Pima Indian Diabetes).

Le but de cette comparison est de voir l'algorithme le plus performant afin de l'utiliser pour l'extraction des règles d'association.

- Dans le troisième chapitre, pour réduire le problème du temps d'exécution et de l'espace mémoire dans les bases de données massives, nous proposons trois algorithmes (DFPS, RAPRIORI, YAFIM) de l'extraction des motifs fréquents en utilisant le framework de traitement de données Spark. Nous avons utilisé des visualisations de données pour déterminer et comprendre les distributions de données, afin de pouvoir les exploiter dans la phase d'extraction des règles d'association. De cette manière, un modèle de clustering utilisant l'algorithme K-modes a été mis en œuvre pour réduire les données et obtenir une cible plus précise.
- Le quatrième chapitre est consacré à la prédiction du diabète en utilisant les algorithmes de machine leaning, à savoir : les réseaux de neurones, les forêts aleatoires, l'arbre de décision, les supports vecteurs machines et les k-plus proches voisins. Ces algorithmes ont été appliqués en se concentrant sur les facteurs de risque (motifs les plus fréquents) du diabète extraits de la base de données PIMA décrite dans le deuxième chapitre.

Chapitre 1 - Etat de l'art sur l'extraction des motifs et règles d'association

1. Introduction

Ce chapitre commence par une description du processus d'extraction de connaissances, suivie d'une présentation des concepts de base impliqués dans l'extraction de motifs et des mesures de qualité pour les règles d'association, et enfin de la citation d'algorithmes d'extraction d'ensembles d'éléments.

2. Processus d'extraction de connaissances

L'extraction de connaissances [1,2] à partir de bases de données est un processus qui consiste à passer de données brutes à des connaissances. Fayyad [3] définit ce concept comme « un processus non trivial qui construit un modèle valide, nouveau, potentiellement utile et au final compréhensible, à partir de données ». Il décompose ce processus en un ensemble d'étapes complexes à savoir :

a. La **sélection** ou la création d'un ensemble de données à étudier ;
b. Le **prétraitement** qui permet d'éliminer le bruit et traiter les données manquantes ;
c. La **transformation** ou la définition des structures optimales de représentation des données ;
d. La **fouille de données** et la détermination de la tâche (classification, recherche de modèles, etc.) en définissant les paramètres appropriés ;
e. L'**interprétation** et l'**évaluation** durant laquelle les éléments extraits sont analysés pour aboutir à des connaissances stockées dans une base de connaissances.

Les différentes étapes de ce processus sont présentées dans la Figure 1.1 et sont décrits comme suit :

a. **Sélection** : Une fois l'objectif de l'extraction est fixé, l'étape de base est la sélection des échantillons significatifs de données. Toutes les données brutes ne sont pas nécessairement pertinentes pour une application des algorithmes de la fouille de données, il est nécessaire de sélectionner un sous-ensemble adapté à l'étude à mener et déterminer la structure générale des données ainsi que les règles utilisées en identifiant les informations exploitables et vérifier leur qualité et leur facilité d'accès.

b. **Prétraitement** : Les erreurs de saisie, les champs nuls, et les valeurs manquantes, impose généralement une phase de nettoyage de données, celle-ci a pour objectif de corriger ou de contourner l'inexactitude et les erreurs de données comme suit :

- Exclure les enregistrements incomplets ;
- Identifier et traiter les valeurs manquantes, les valeurs erronées ou incertaines et les

inconsistances ;

- Remplacer les données manquantes ;
- Prédire les valeurs (valeur moyenne des objets similaires, la régression) ;
- Utiliser l'absence de valeur comme une information ;

c. **Transformation:** En vue d'appliquer un traitement spécifique aux données précédemment sélectionnées, il est nécessaire d'adapter leur structure dans un format approprié à la tâche de la fouille de données choisies dans la troisième étape.

d. **Fouille de données:** Dans cette phase, des méthodes intelligentes sont utilisées afin d'extraire les connaissances utiles à partir de données et les présenter sous une forme synthétique. Lors de cette étape plusieurs techniques peuvent être utilisées à savoir, le clustering, la classification, la régression, les règles d'association, etc.

e. **Interprétation:** Cette étape identifie les modèles intéressants représentant les connaissances, en se basant non-seulement sur des mesures d'intérêt, mais aussi sur l'avis de l'expert. Cette évaluation prend généralement une forme graphique ou textuelle et contribue fortement à améliorer la lisibilité et la compréhension des résultats et facilite le partage de la connaissance. Les résultats produits par les algorithmes de fouille de données ne sont pas toujours exploitables directement. En effet, il est utile de définir des nouvelles mesures de qualité afin d'assister le décideur à utiliser les règles d'association les plus pertinentes.

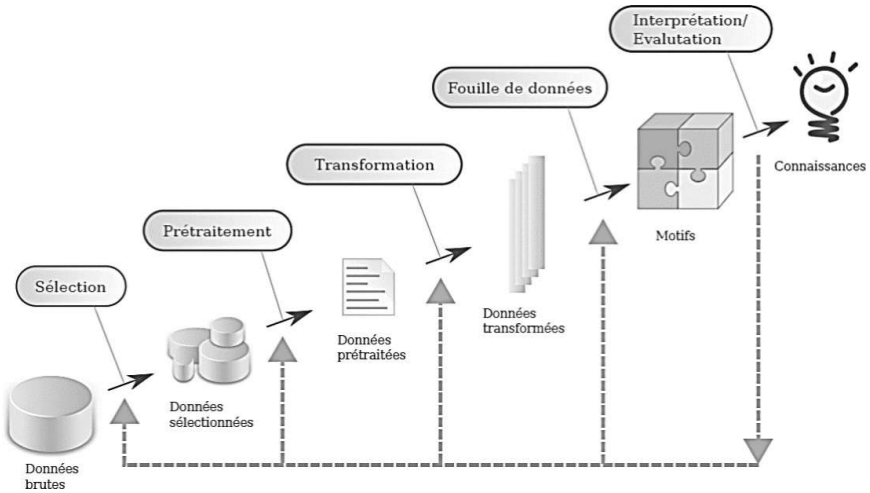

Figure 1.1. Processus de l'extraction de connaissance dans les bases de données

Il existe une distinction précise entre le concept de l'ECD et celui de la fouille de données. En effet, ce dernier n'est qu'une des étapes de découverte de connaissances correspondant à

l'extraction de connaissances à partir de données. Les méthodes de la fouille de données peuvent être divisées en deux grandes familles : les méthodes descriptives et les méthodes prédictives. En ce qui concerne la première famille, elle est caractérisée par son mode de découverte et d'analyse descriptive de données. La deuxième famille cherche à prédire une donnée particulière. Parmi les techniques prédictives utilisées en data mining, on peut citer les règles d'association, qui constituent sans doute la tâche phare qui a attiré l'attention des chercheurs et sur laquelle de nombreux travaux ont été réalisés. Cette technique a pour vocation d'extraire les règles intelligible et exploitable à partir des grands volumes de données. Ces règles d'association sont des implications de la forme : $a \Rightarrow b$ ou a et b sont des items (variables booléennes de la forme attribut = valeur). Une telle règle décrit une corrélation entre l'ensemble d'items dans une base de transactions. Autrement dit, étant donné un ensemble d'items, l'objectif est de découvrir si l'occurrence de cet ensemble est associée à une autre occurrence d'un autre ensemble d'items. Par exemple, « 90% des clients qui achètent un smartphone achètent aussi une pochette et un abonnement à Internet » est une règle d'association associant l'item smartphone aux items pochette et abonnement à Internet.

3. Extraction des motifs

Nous commençons par définir l'ensemble des notions de base relatives à la technique d'extraction des motifs, qui seront utilisés tout au long de ce travail. Définissons d'abord la version de base d'extraction de motifs qui permet de faire la fouille dans une base des transactions.

3.1. Base de transactions

Une base de transactions (appelée aussi contexte d'extraction) est défini par un triplet

$D = (T,I,R)$ où :

- T est un ensemble fini de transactions (ou objets).
- I est un ensemble fini d'items (ou attributs).
- R est une relation binaire $R \subseteq T * I$ entre les transactions et les items.

Un couple (t, i) $\in R$ dénote le fait que la transaction t $\in T$ contient l'item i $\in I$.

Exemple : Le contexte D représenté par la tableau 1.1 représente une base transactionnelle de 6 transactions (T = {1, 2, 3, 4, 5, 6}) et de 5 items (I = {A, B, C, D, E}). Le couple (2, B) $\in R$ car la transaction 2 $\in T$ contient l'item B $\in I$.

D= {1, ACD), (2, BCE), (3, ABCE), (4, BE), (5, ABCE), (6, BCE)}. Cette base peut être représentée sous forme horizontal (Tableau 1.2(a)), vertical (Tableau 1.2 (b)) ou booléenne (Tableau 1.2 (c)).

Tableau 1.1 : (Base de Transactions) D, T = {1, 2, 3, 4,5} et I = {A, B, C,D, E}

TID	Items
1	ACD
2	BCE
3	ABCE
4	BE
5	ABCE
6	BCE

Tableau 1.2 : a) Représentation horizontale de D, b) Représentation Verticale de D, c) Représentation en mode binaire

1	A	C	D	
2	B	C	E	
3	A	B	C	E
4	B	E		
5	A	B	C	E
6	B	C	E	

a)

A	B	C	D	E
1	2	1	1	2
3	3	2		3
5	4	3		4
	5	5		5
	6	6		6

b)

	A	B	C	D	E
1	1	0	1	1	0
2	0	1	1	0	1
3	1	1	1	0	1
4	0	1	0	0	1
5	1	1	1	0	1
6	0	1	1	0	1

c)

3.2. Motif ou Itemset

Un motif, aussi appelé itemset, est un sous-ensemble non vide de I où I représente l'ensemble des items. Une transaction t ∈ *T*, avec un identificateur communément noté TID (Tuple IDentifier), contient un ensemble, non vide, d'items de *I*. Un sous-ensemble I de *I* où k =|I| est

appelé un k-motif ou simplement un motif. k représente la cardinalité de I. La cardinalité d'un motif varie de 0(∅) à m (I) ou m est le nombre d'items de *I*.

3.3. Support d'un motif

Le support d'un motif I⊆*I* (support relative) est donc le rapport de la cardinalité de l'ensemble des transactions qui contiennent tous les items de *I* par la cardinalité de l'ensemble de toutes les transactions. Il capture la portée du motif, en mesurant sa fréquence d'occurrence.

$$\sup(I) = \frac{|\{\psi(\mathrm{I})\}|}{|(O)|}$$

Notons que $|\psi(I)|$ est appelé support absolu de I.

Exemple: Dans la base des transactions D, on trouve, support(A)=3/6 et support(CE)=4/6. Le support des motifs est anti-monotone par rapport à l'inclusion ensembliste c'est à dire, si I_1 est un sous motif I ($I_1 \supseteq$ I) alors Support (I_1) ≥ Support (I).

Le support mesure la fréquence d'un motif : plus il est élevé, plus le motif est fréquent. On distinguera les motifs fréquents des motifs in-fréquents à l'aide d'un seuil minimal de support.

3.4. Treillis des motifs

Un treillis des motifs [4] est un regroupement conceptuel et hiérarchique des motifs. Il est aussi dit treillis d'inclusion ensembliste. Toutefois l'ensemble des parties de I est ordonné par inclusion ensembliste dans le treillis des motifs. Le treillis des motifs associé à la base de données présenté au Tableau 1.1 est représenté par la figure 1.2. Dans la base des transactions illustrée dans le Tableau 1.2, t_1 possède les motifs : ϕ , a, c, d, ac, ad, cd et acd. Parmi l'ensemble global de 2^P motifs (avec p le nombre d'items dans la base de données, ici p=5), on va chercher ceux qui apparaissent fréquemment. Pour cela, on introduira les notions de connexion de Galois et de support d'un motif. Pour cette base on a:

- Un motif de taille 0 = ∅ (C_0 = 0 motifs).
- Deux motifs de taille 1 = {a}, {b}, {c}, {d} et {e}, qu'on notera, pour simplifier, a, b, c, d et e. (C1 =5 motifs).
- Dix motifs de taille 2 = ab, ac, ad, ae, bc, bd, be, cd, ce, de (C_2 =10 motifs)
- Dix motifs de taille 3 = abc, abd, abe, acd, ace, ade, bcd, bce, bde, cde (C_3 = 10 motifs).
- Cinq motifs de taille 4 = abcd, abce, abde, acde, bcde (C_4 = 5 motifs).

- Un motif de taille 5 = abcde (C_5 = 1 motifs).

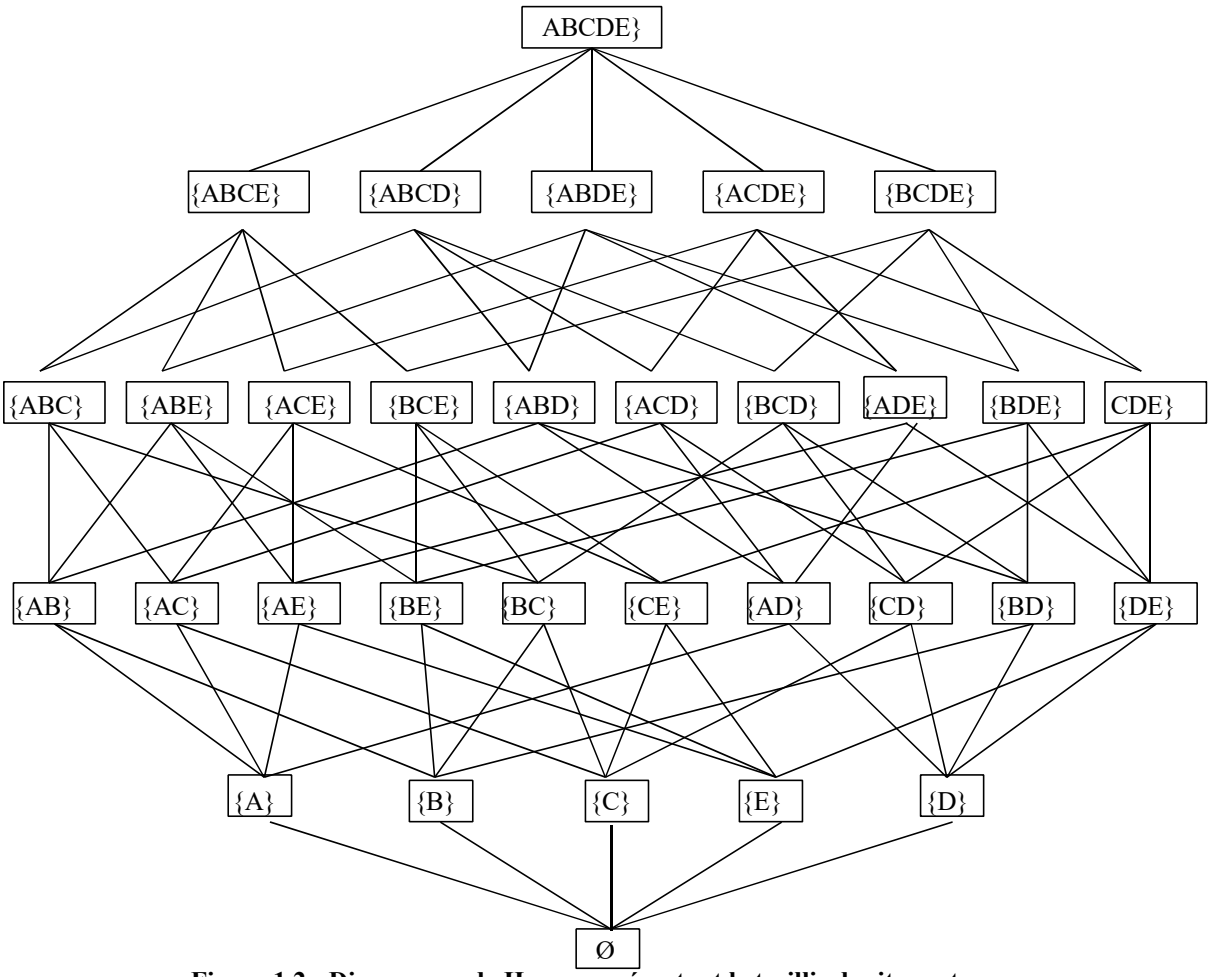

Figure 1.2 : Diagramme de Hasse représentant le treillis des itemsets

3.5. **Correspondance de Galois** [4]

Soit le contexte d'extraction D = ({O, *I*, *R*}). Soit l'application Φ de l'ensemble des parties de O (c'est-à-dire l'ensemble de tous les sous-ensembles de *O*), noté par P(*O*), dans l'ensemble des parties de *I*, noté par P(*I*). L'application φ associe à un ensemble d'objets O⊆ *O*, l'ensemble des items i ∈ *I* communs à tous les objets o ∈ O. l'ensemble des parties d'un ensemble d'éléments O, constitué de tous les sous-ensembles de O noté 2^O

$$2^O \rightarrow 2^I$$

$$\Phi: P(O) \rightarrow P(I)$$

$$\Phi(O) = \{i \in I \mid \forall o \in O, (o, i) \in R\}$$

$$2^I \rightarrow 2^O$$

$$\Psi: P(I) \rightarrow P(O)$$

$$\Psi(I) = \{o \in O \mid \forall i \in I, (o, i) \in R\}$$

Φ(O) dénote l'ensemble de tous les items communs à un groupe de transactions T (intension), et ψ(I) l'ensemble de toutes les transactions partageant les mêmes items de I (extension). Le couple (ψ, φ) définit une correspondance de Galois entre l'ensemble des parties de O et l'ensemble des parties de I.

Les applications $\gamma = \Phi o \Psi$ et $\omega = \Psi o \Phi$ sont appelées les opérations de fermeture de la correspondance de Galois.

Par exemple, dans la base de données de la tableau 1.2, nous avons Φ ({4, 5}) = {B, E} et ψ({A, C}) = {1, 3, 5}. Ce qui signifie que l'ensemble de transactions {4, 5} possède en commun l'ensemble d'attributs {B, E}. De la même manière, l'ensemble d'attributs {A, C} possède en commun l'ensemble de transactions {1, 3, 5}.

La définition suivante présente le statut de fréquence d'un motif, fréquent ou in-fréquent, étant donné un seuil minimal de support.

3.6. Opérateur de fermeture

Les applications $\gamma = \Phi \circ \Psi$ et $\omega = \Psi \circ \Phi$ sont appelées les opérateurs de fermeture [4] de la correspondance de Galois [4]. Par exemple, dans le contexte D du Tableau 1.2, on a si T = {3, 5}, alors Φ (T) = {ABCE} et donc ω ={3, 5}. Et si T = {1, 2, 3}, alors Φ (T) = {C} et donc ω = {1, 2, 3, 5}. Si O = {AC}, alors ψ(O) = {1, 3, 5} et donc γ = {AC}. Dans ces exemples, les ensembles {3, 5} et {AC} sont fermés.

L'opérateur de fermeture γ, tout comme ω, est caractérisé par le fait qu'il est :

- **Isotonie** : $I_1 \subseteq I_2 \Rightarrow \gamma(I_1) \subseteq \gamma(I_2)$
- **Expansivité** : $I \subseteq \gamma(I)$
- **Idempotence** : $\gamma(\gamma(I)) = \gamma(I)$

3.7. Motif fréquent, Motif rare [5,6]

Soit une base de transactions D = (O, I , R), un seuil minimal de support conjonctif minsupp, un motif I ⊆ I est dit fréquent si son support relative (supp) dépasse un seuil minimum fixé par l'utilisateur noté minsupp (supp≥ minsupp) sinon I est dit infréquent ou rare.

Exemple Dans le Tableau 1.2, en fixant une valeur minsupp = 3/5 on obtient que {A} et {BC} sont fréquents, alors que le motif {ABC} est in-fréquent ou rare.

- **Motif fréquent maximal:** Un motif est dit motif fréquent maximal s'il est fréquent et si tous ces sur-motifs ne le sont pas.

- **Motif fermé:** Un motif est dit fermé lorsqu'aucun des motifs qui le contiennent ne possède le même support. Un motif X est dit fermé si aucun de sur-motifs n'a pas exactement le même support que X.
- **Motif fréquent maximal:** Un motif fréquent est dit Maximal si aucun de ses sur-motifs immédiats n'est fréquent.

Exemple: Le schéma de la figure 1.3 illustre la relation entre le motifs fréquents, fréquents fermés et fréquents maximaux:

- Les motifs encadrés par les lignes minces ne sont pas fréquents, les autres le sont.
- Les motifs encadrés par des lignes plus épais sont fermés.
- Les motifs encadrés par des lignes plus épais et colorés sont maximaux

Les motifs maximaux ⊏ Les motifs fermés ⊏ Les motifs frequents.

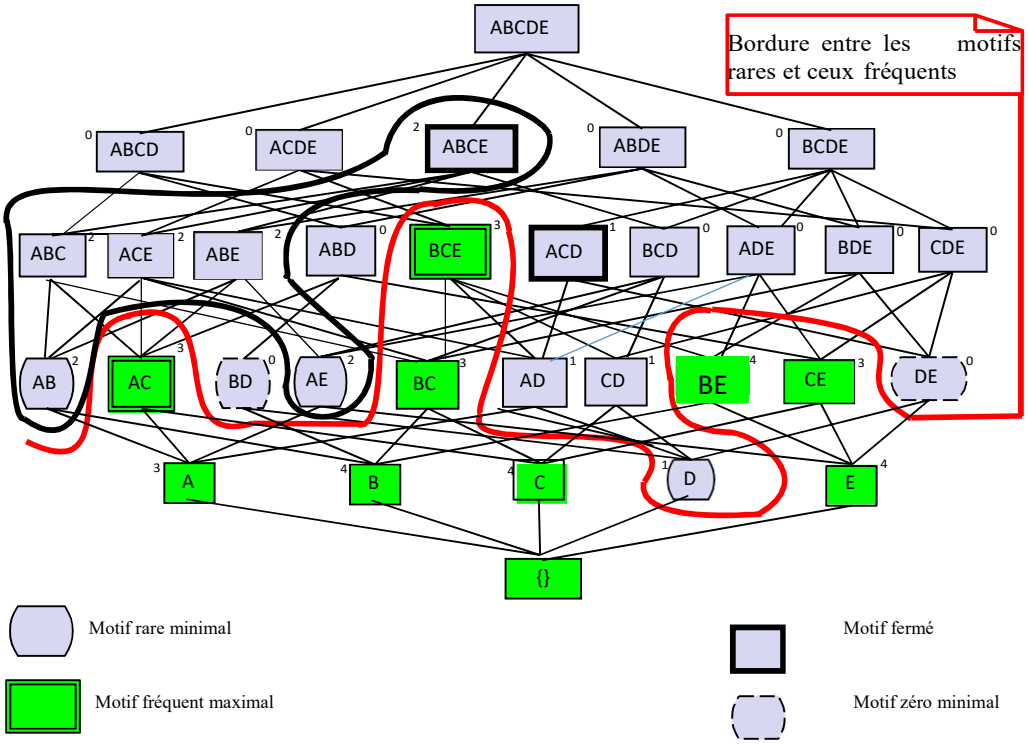

Figure.1.3 : Treillis des itemsets associé au contexte D pour minsupp = 3

3.8. Bordure positive et négative

Compte tenu d'un seuil minimum de support minsupp. La bordure positive [6] Bd^+ représentée par les motifs fréquents maximaux. Bd^+, est l'ensemble des plus grands itemsets fréquents (au sens de l'inclusion) dont tous les sur-ensembles sont fréquents, et est définie comme suit:

$$Bd^+ = \{I \in I \mid supp(I) \geq minsupp, \forall I_1 \supseteq I, supp(I_1) \geq minsupp\}$$

La bordure négative [6] Bd− est représentée par les motifs in-fréquents minimaux. Bd^-, est l'ensemble des plus petits itemsets qui ne sont pas fréquents dont tous les sous-ensembles sont fréquents, et est définie comme suit:

$$Bd^- = \{I \in I \mid supp(I) \leq minsupp, \forall I_1 \subseteq I, supp(I_1) \geq minsupp\}$$

Exemple: Soit le treillis donné par la figure 1.3. Pour minsupp=2, les bordures positive et négative pour la contrainte fréquence sont comme suit:

$$BD^+(MF) = \{(ABCE, 2)\}, BD^-(MF) = \{(D, 1)\}.$$

L'ensemble des motifs rares et l'ensemble des motifs fréquents ont tous les deux un sous-ensemble minimal générateur.

Dans le cas des motifs rares, l'ensemble générateur minimal est l'ensemble des motifs rares minimaux *MRM*. Un motif rare minimal est un motif rare ayant tous ces sous-ensembles fréquents. L'ensemble des motifs rares minimaux forme un ensemble générateur minimal à partir duquel tous les motifs rares peuvent être retrouvés. D'une manière duale, l'ensemble des fréquents maximaux forme un ensemble générateur à partir duquel tous les motifs fréquents peuvent être retrouvés.

4. Mesure de qualité des règles d'association

Dans cette section on présente les notions utilisées pour mesurer la qualité des règles d'association.

Définition 1 (Règle d'association). Une règle d'association est un couple (A, B), où A et B sont des itemsets non vides disjoints, i.e. $A \neq \emptyset$,$B \neq \emptyset$ et $A \cap B = \emptyset$. On note classiquement une telle règle sous la forme $A \rightarrow B$. A s'appelle l'antécédent de la règle et B le conséquent de la règle.

Un exemple de règle d'association extraite d'une base de données de ventes de supermarché est: céréales ∧ sucre → lait. Cette règle indique que les clients qui achètent des céréales et du sucre

ont également tendance à acheter du lait. La mesure de support définit la portée de la règle, c'est-à-dire la proportion de clients ayant acheté les trois articles, et la mesure de confiance définit la précision de la règle, c'est-à-dire la proportion de clients ayant acheté du lait parmi ceux ayant acheté des céréales et du sucre.

Définition 2 (Règle redondante). Une règle d'association est dite redondante si elle n'est pas porteuse de connaissances supplémentaires par rapport à l'ensemble des règles résultantes.

Ainsi, il est nécessaire de prendre en considération les mesures de qualité des règles d'association, les mesures les plus utilisées sont le support, la confiance et le lift. Cependant il existe d'autres mesures de qualité.

Définition 3 (Règle d'association solide**).** Une règle d'association A → C est dite solide si étant donné un support minimum γ, l'itemset AUC est fréquent et si sa confiance dépasse un seuil donné, fixé a priori appelé le seuil de confiance minimum appelé σ.

$$A \rightarrow C \text{ est solide ssi } sup(AUC) \geq \gamma \text{ et } conf(A \rightarrow C) \geq \sigma$$

Définition 4 (Support d'une règle). On définit le support d'une règle d'association comme étant le support de l'itemset A ∪ B (i.e. la proportion de transactions contenant à la fois A et B).

$$Supp(A \rightarrow B) = Sup(A \cup B) = \left| \frac{t(A \bigcup B}{|((A)|} \right| \qquad (1.1)$$

Définition 5 (Confiance d'une règle). La confiance d'une règle A → B, notée Conf (A → B), représente la probabilité conditionnelle qu'une transaction contenant B sachant qu'elle contient A

$$Conf(A \rightarrow B) = \frac{\sup(AUB)}{\sup(A)} \qquad (1.2)$$

Remarque: $0 \leq Conf(A \rightarrow B) \leq 1$ et $0 \leq Supp(A \rightarrow B) \leq 1$

Pour illustrer la notion de confiance, on considère deux ensembles de transactions t(A) et t(B). La confiance est une mesure permettant d'évaluer la solidité de la règle d'association, elle mesure le degré d'inclusion de A dans B, où t(A) représente le nombre de transaction contenant A et t(B) représente le nombre de transaction contenant B.

Définition 6 : Le Lift : Le Lift est une mesure statistique, symétrique, représente le rapport d'indépendance entre l'antécédent et le conséquent de la règle. Il prend ses valeurs dans l'intervalle [0, +∞ [. Elle est définie par la formule 3.

$$\text{Lift}(A \rightarrow B) = \frac{conf(AUB)}{\sup(B)} \qquad (1.3)$$

Définition 6 : La Conviction: la Conviction est une mesure non symétrique qui mesure le dégrée d'implication de la règle, elle est proposée pour pallier au limite de la confiance. La conviction varie dans l'intervalle [0, +∞ [, plus cette valeur est grande le conséquent apparait avec l'antécédent. Elle est donnée par la formule 4.

$$\text{Conv}(A \rightarrow B) = \frac{1-\sup(B)}{1-conf(A \rightarrow B)} \qquad (1.4)$$

Nous avons aussi les trois cas suivants:

- Si Conv $(A \rightarrow B) = 1$ alors A et B sont indépendants
- Si Conv $(A \rightarrow B) > 1$ alors A et B sont positivement dépendants
- Si Conv $(A \rightarrow B) < 1$ alors A et B sont négativement dépendants

Le nombre des règles extraites à partir d'une base de données est exponentiel en fonction d'items d qui composent le jeu de données est $R=3^d-2^{d+1}+1$. L'extraction des règles d'association consiste à extraire les règles dont le support et la confiance sont au moins égaux respectivement, à des seuils minimaux de support et de confiance définis par l'utilisateur. La plupart des approches proposées pour l'extraction des règles d'association reposent sur les quatre phases suivantes:

a. **Préparation des données**:

Cette phase consiste à sélectionner les données (attributs et objets), de la base de données, utiles à l'extraction des règles d'association et transformer ces données en un contexte d'extraction.

b. **Extraction des ensembles fréquents d'attributs:** Cette phase consiste à extraire du contexte tous les itemsets fréquents. Un itemset est fréquent si son support est supérieur ou égal au seuil minimal de support défini par l'utilisateur.

c. **Génération des règles d'association**: Cette phase consiste à utiliser les itemset fréquents extraits de la phase précédente pour générer les règles d'association ayant la confiance supérieure ou égale à la confiance minimale.

d. **Visualisation et interprétation :** Cette phase consiste à la visualisation des règles d'association extraites afin d'en déduire des connaissances utiles pour l'amélioration de l'activité concernée en utilisant certaines mesures à citer le support, la confiance, etc. [7,8].

5. Algorithmes d'extraction des itemsets

Dans cette section, nous présentons une vue panoramique sur quelques algorithmes d'extraction des itemsets fréquents les plus connus qui permettent d'extraire des règles d'association. De nombreux algorithmes ont été développés pour résoudre le problème de recherche des motifs fréquents [9,10,11]. Ces algorithmes peuvent être classés en trois grandes catégories: les algorithmes d'extraction des motifs fréquents, des motifs fréquents maximaux [12,13,14] et des motifs fréquents fermés [15,16,17]. Ici on s'interese seulement aux algorithmes d'extractions des items fréquents les plus utilisés.

5.1. Algorithmes d'extraction des itemsets fréquents

5.1.1. Algorithme Apriori

L'algorithme Apriori proposé par Agrawal [9] et ses co-auteurs est un algorithme de base qui permet d'extraire des motifs fréquents dans une base ayant plusieurs milliers d'attributs et plusieurs millions d'enregistrements. L'algorithme Apriori répond au critère d'anti-monotonie. C'est à dire, si un item/itemset n'est pas fréquent, alors tous ses sur-ensembles ne peuvent pas être fréquents. Apriori opère en deux étapes pour extraire les motifs fréquents:

a. Étape de combinaison des items.
b. Étape d'élagage.

Pour extraire les motifs fréquents, Apriori parcourt la base de données D et détermine une liste candidate C1 d'items fréquents de taille 1, puis l'algorithme filtre C1 et ne conserve que les items qui satisfont le support minimum MinSup et les stocke dans une liste des fréquents F1. À partir de F1, l'algorithme génère les motifs candidats de taille 2 dans une liste disons C2 et cela en combinant toutes les paires d'items fréquents de taille 1 en F1. Ensuite, Apriori analyse D et détermine tous les motifs dans C2 qui satisfont le support MinSup, le résultat est stocké dans une liste F2. Le processus d'exploration d'Apriori est exécuté jusqu'à ce qu'il n'y ait plus des motifs candidats dans D à vérifier.

5.1.2. Algorithme AprioriTID

Afin d'améliorer les performances de l'algorithme Apriori, les mêmes auteurs ont proposé dans [12] l'algorithme AprioriTID qui est basé sur le même principe d'Apriori, mais dans AprioriTID à partir de deuxième passage, la BD n'est plus utilisée pour calculer les supports des itemsets

candidats. Il utilise un ensemble C_k de la forme (Tid, C_k) ou C_k correspond à la liste des itemsets contenus dans la transaction identifiée par Tid. Pour K =1, C_1 correspond à la base de transaction D. L'amélioration qu'apporte cet algorithme par rapport au précédent est le fait de stocker à chaque itération les identificateurs des transactions contenant les sous-ensembles fréquents dans l'ensemble C_k par conséquent, on réduit le nombre de passages de la base de données.

5.1.3. Algorithme DIC

L'algorithme DIC (Dynamic Itemset Counting) proposé par Brin et al. [13] pour réduire le nombre de parcours de la base de transaction. L'idée de base est de partitionner la base de données en M blocs de transactions, après le parcours d'une partition de taille m on vérifie les k-itemsets candidates qui ont déjà atteint le support minimum pour générer les candidates de taille (k + 1). Par exemple, Apriori peut produire 3 parcours pour compter 3-itemsets tandis que DIC produit 1.5 parcours. Et dans le premier parcours avec 1-itemet, DIC peut compter certains itemsets qui sont 2-itemsets ou 3-itemsets.

5.1.4. Algoritme RELim

RElim est un algorithme d'extraction des motifs développé en 2005 par Christian Borgelt's [14] et inspiré de l'algorithme FP. L'algorithme RElim est basé sur l'élimination récursive des éléments de la base de données transactionnelle d'une façon semblable à plusieurs autres algorithmes de la recherche des éléments fréquents: dans un traitement initial, il détermine les fréquences des éléments. Tous les éléments rares sont rejetés des transactions car ils ne peuvent pas être parmi les éléments fréquents. En outre, les éléments dans le cadre de chaque opération sont triés d'une façon croissante leur fréquences dans la base de données. Bien que l'algorithme ne nécessite pas cet ordre spécifique, les expériences ont montré qu'il conduit à des temps d'exécution beaucoup plus courts qu'un ordre aléatoire.

5.1.5. Algorithme SAM

L'algorithme SaM (Split and Merge) [15,16] établi par Christian Borgelt. C'est la simplification de l'algorithme RElim [14] déjà assez simple. Alors que RElim représente une base de données (conditionnelle) en stockant une liste de transactions pour chaque élément (représentation partiellement verticale), l'algorithme de division et de fusion n'utilise qu'une seule représentation de transaction, stockée sous forme de tableau. Ce tableau est traité avec un schéma de fractionnement et de fusion, qui calcule une base de données conditionnelle, traite

récursivement cette base de données conditionnelle et élimine enfin l'élément fractionné de la base de données d'origine (conditionnelle).

5.1.6. Algorithme Eclat

L'algorithme Eclat [17] consiste à effectuer un processus d'extraction des motifs fréquents dans la mémoire sans accéder au disque. L'algorithme procède en stockant une liste d'identifiants de transaction (TID) dans la mémoire de chaque item de la base de données. Pour déterminer le support d'un motif I, Eclat croise les TIDs de tous les items de I. Eclat effectue une recherche des itemsets fréquents en profondeur d'abord et se base sur le concept de classes d'équivalence. Par exemple, ABC et ABD appartiennent à la même classe d'équivalence. Deux k-itemsets appartiennent à une même classe d'équivalence s'ils ont en commun un préfixe de taille (k -1).

5.1.7. Algorithme FP-Growth

FP-Growth (Frequent-Pattern Growth) [18] a été considéré comme l'algorithme le plus performant par rapport aux autres algorithmes pour extraire des itemsets fréquents. L'algorithme consiste d'abord à compresser la base de données en une structure compacte appelée FP-tree (Frequent Pattern tree) et qui apporte une solution au problème de la fouille de motifs fréquents dans une grande base de données transactionnelle. L'algorithme FP-Growth ne repose sur aucune approche de génération de motifs candidats.
L'algorithme FP-Growth effectue deux passes (scans) à la base de transactions :

- **Passe 1** le premier passage de FP-Growth sur la base de données D est consacré à déterminer la valeur du support de chaque item dans D. L'algorithme ne retient que les éléments fréquents dans une liste F-List. Ensuite, FP-Growth trie F-List dans un ordre décroissant en fonction de la valeur de support et qui est comparé avec le seuil de support préfixé (MinSup).
- **Passe 2** un FP-Tree est construit par la création d'une racine vide et un second parcours de la base de données où chaque transaction est décrite dans l'ordre des items donné par la liste F-List. Chaque nœud de l'arbre FP-Tree représente un élément dans L et chaque nœud est associé à un compteur (c'est-à-dire, compte de support initialisé à 1). Si une transaction partage un préfixe commun avec une autre transaction, le compte de support de chaque nœud visité est incrémenté de 1. Pour faciliter la traversée de FP-Tree, une table d'entête est construite pour

que chaque élément pointe vers ses occurrences dans l'arbre via une chaîne de liens-nœuds. En dernier lieu, le FP-Tree est fouillé par la création des (sub-) fragment conditionnels de base. En fait, pour trouver ces fragments, on extrait pour chaque fragment de longueur 1 (suffix pattern) l'ensemble des préfixes existant dans le chemin du FP-Tree (conditionnel pattern base).
L'itemset fréquent est obtenu par la concaténation du suffixe avec les fragments fréquents extraits des FP-Tree conditionnels. En dernier lieu, le FP-Tree est fouillé par la création des (sub-) fragment conditionnels de base. En fait, pour trouver ces fragments, on extrait pour chaque fragment de longueur 1 (suffix pattern) l'ensemble des préfixes existant dans le chemin du FP-Tree (conditionnel pattern base).
L'itemset fréquent est obtenu par la concaténation du suffixe avec les fragments fréquents extraits des FP-Tree conditionnels.

5.1.8. Algorithme Pascal

L'algorithme Pascal [19] a pour rôle l'extraction efficace des motifs fréquents dans les grandes bases de données ainsi qu'une méthode de simplification des règles d'association. Basé sur une nouvelle optimisation, simple et efficace, de l'algorithme Apriori, il est donc facile à intégrer dans les implantations existantes. Cette optimisation, l'inférence de comptage, s'appuie sur les notions de classes d'équivalence des motifs et de motifs clés. Elle permet de calculer depuis la base de données les supports de certains motifs seulement – les motifs clés – contrairement aux algorithmes basés sur l'extraction par niveaux des motifs fréquents ou l'extraction des motifs fréquents maximaux.

5.2. Algorithmes d'extraction des itemsets fréquents fermés

Les premiers algorithmes d'extraction des itemsets fermés fréquents sont pour l'essentiel des algorithmes séquentiels. Un premier examen de ces algorithmes permet de les classer selon la technique adoptée pour l'exploration de l'espace de recherche, à savoir « tester et générer » et « diviser pour régner ».

- **La technique « tester et générer » :** les algorithmes parcourent l'espace de recherche par niveau. A chaque niveau k, un ensemble de candidats de taille k est généré. Cet

ensemble de candidats est, généralement, élagué par la conjonction d'une métrique statistique (e.g. le support) et des heuristiques basées essentiellement sur les propriétés structurelles des itemsets fermés.

- **La technique « diviser pour régner » :** les algorithmes essaient de diviser le contexte d'extraction en des sous-contextes et d'appliquer le processus de découverte des itemsets fermés récursivement sur ces sous-contextes. Ce processus de découverte repose sur un élagage du contexte basé essentiellement sur l'imposition d'une métrique statistique et d'heuristiques introduites.

Parmi ces algorithmes on trouve, ceux basés sur la technique « tester et générer » comme Close [23], A-Close [23], CHARM [24], TITANIC [25], et ceux basés sur la technique « diviser pour régner » comme Closet [26], Closet+ [27], FP-Close [28], LCM [29], PTClose [30].

5.2.1. Algorithme Close

L'algorithme CLOSE génère un ensemble de candidats en joignant les générateurs minimaux retenus durant l'itération précédente. CLOSE calcule alors leurs supports et leurs fermetures dans une même étape par le biais d'un accès au contexte d'extraction. La fermeture d'un candidat générateur minimal g est calculée en exécutant des intersections de l'ensemble des objets auxquelles appartient g. Afin de réduire l'espace de recherche, c'est-à-dire le nombre de candidats à tester, CLOSE utilise des stratégies d'élagage. Ces dernières sont basées sur une mesure statistique à savoir minsupp et sur l'idéal d'ordre des générateurs minimaux ainsi que sur le fait qu'un candidat générateur minimal g de taille k ne doit pas être couvert par la fermeture d'un de ses sous-ensembles de taille (k - 1).

5.2.2. Algorithme A-Close [23]

L'algorithme A-CLOSE est une variation de l'algorithme CLOSE. Il a pour objectif de réduire l'espace de recherche par les itemsets fermés au lieu de celui des itemsets, il effectue l'action de calcul des fermetures après la boucle principale, ce qui élimine le calcul local répétitif de la fermeture des itemsets fréquents dans chaque itération. Il inclue une nouvelle heuristique pour la phase d'élagage de l'ensemble de candidats qui consiste à éliminer un candidat à l'étape (k+1), s'il est aussi fréquent qu'un de ses k-sous-ensembles à l'étape (k). Ce critère réduit considérablement le nombre de candidats générés à chaque étape. L'algorithme A-CLOSE effectue le processus d'extraction en deux étapes qui sont implémentées par la fonction AC-GENERATEUR. Ainsi, l'algorithme A-CLOSE instancie les deux étapes comme

suit :

- **Étape d'élagage :** durant cette étape, la fonction AC-GENERATEUR est appliquée à chaque générateur de G_k, déterminant ainsi son support. Les k-générateurs infréquentés sont éliminés.
- **Étape de construction :** la fonction AC-GENERATEUR prend en entrée l'ensemble G_k des k-générateurs et calcule l'ensemble G_{k+1} contenant tous (k + 1) - générateurs, qui seront utilisés dans l'itération suivante. A ce niveau, comme l'algorithme A-CLOSE dispose seulement de l'ensemble des générateurs minimaux obtenus au niveau k, l'ensemble G_{k+1} est élagué comme suit. Si un élément de G_{k+1} est aussi fréquent qu'un de ses k-sous-ensembles de G_k, alors il est éliminé de G_{k+1}.

La boucle principale s'arrête quand il n'y a plus de générateurs à traiter.

5.2.3. Algorithme Closet [26]

L'algorithme CLOSET propose une approche originale et efficace, appelée « Diviser pour régner », pour la découverte des itemsets fermés fréquents. L'algorithme CLOSET proposait d'adopter une structure de données avancée, où la base de données peut être compressée pour arriver à achever le processus de fouille de données (e.g. la structure FP-tree). L'idée motivant cette structure de donnée compacte, basée sur la notion de trie, est que lorsque plusieurs transactions se partagent un item, alors elles peuvent être fusionnées en prenant soin d'enregistrer le nombre d'occurrences des items. L'algorithme CLOSET effectue le processus d'extraction des itemsets fermés en deux étapes successives :

a. Les items des transactions sont ordonnés par support décroissant. Ensuite, l'arbre FP-Tree est construit. Cette structure de donnée est construite comme suit. Premièrement, le nœud racine est créé et étiqueté par « root ». Pour chaque transaction du contexte, les items sont traités et une branche est créée pour chaque transaction. Dans chaque nœud du FP-tree, il y a un compteur qui garde trace du nombre d'apparitions de l'item dans la base de test. Spécifiquement dans le cas où une transaction présente un préfixe commun avec une branche du FP-tree, alors le compteur de chaque nœud appartenant à ce préfixe est incrémenté de 1 et une sous-branche va être créée contenant le reste des items de la transaction.

b. Au lieu d'une exploration en largeur des itemsets fermés candidats, il effectue une partition de l'espace de recherche pour effectuer ensuite une exploration en profondeur d'abord. Ainsi, il commence par considérer les 1-itemsets fréquents, et examine seulement leur base

conditionnelle. Une base conditionnelle ne contient que les items qui co-occurrent avec le 1-itemset en question. Le FP-Tree conditionnel associé est construit et le processus se poursuit récursivement. Il est essentiellement basé sur les propriétés suivantes :

a. L'itemset fermé, disons p, qu'on extrait d'une base conditionnelle est constitué par la concaténation des 1-itemsets qui sont aussi fréquents que p.

b. Il n'y a nul besoin de développer une base conditionnelle d'un itemset, disons p, qui est inclus dans un itemset fermé déjà découvert, disons c, tel que support(p)=support(c).

5.2.4. Algorithme FP-Close [28]

Dans cette section, nous présentons l'un des meilleurs algorithmes séquentiel FP-Close qui consiste à extraire les itemsets fermés fréquents par l'utilisation d'une nouvelle structure de représentation appelée « CFI-Tree » conçue pour compresser la base de données et pour faciliter la localisation des itemsets fermés fréquents, elle est une autre variation de la structure « FP-tree ». L'ordre des items dans la structure CFI-Tree est le même dans la table entête de la structure FP-tree et l'insertion d'un itemset fermé dans la structure FCI-tree est similaire à l'insertion d'une transaction dans FP-tree, sauf que le support d'un nœud n'est pas incrémenté, il est remplacé par le support maximal. Dans la structure CFI-tree, un nœud contient quatre champs : Nom de l'item, le niveau sur l'arbre, le support et un pointeur vers l'item suivant de même label.

L'algorithme FP-Close effectue le processus d'extraction des itemsets fermés en trois étapes successives :

a. Construction de FP-tree : cette étape nécessite deux scannes à la base de données, nécessaires pour le calcul de fréquence de chaque item et déterminer la liste des items fréquents F-Liste triés par ordre décroissant de leurs supports. En fonction de cette liste, l'arbre FP-tree est construit, où chaque item fréquent dans chaque transaction est inséré dans l'arbre selon l'ordre de F-Liste.

b. Construction de CFI-tree : dans cette étape il effectue une partition de l'espace de recherche en bases conditionnelles de chaque item de F-Liste, ensuite le FP-tree conditionnel associé est construit et le processus se poursuit récursivement. Les itemsets fermés fréquents trouvés de chaque base conditionnelle sont insérés dans l'arbre CFI-tree.

c. Génération des itemsets fermés fréquents : les itemsets fermés fréquents sont générés de CFI-tree afin d'éliminer toute itemset P inclus dans un autre itemset fermé fréquent

C tel que support (P) = support (C).

5.2.5. Algorithme Titanic

L'algorithme TITANIC est proposé par Stumme & al. [31]. TITANIC détermine dans chaque itération les générateurs minimaux fréquents associés, moyennant un accès au contexte d'extraction. Il utilise pour cela les mêmes stratégies d'élagage qu'A-CLOSE. Cependant, TITANIC évite le balayage coûteux effectué par A-CLOSE pour vérifier la dernière stratégie d'élagage. Pour cela, TITANIC utilise pour chaque candidat g de taille k une variable où il stocke son support estimé, c'est-à-dire le minimum du support de ses sous-ensembles de taille (k - 1), et qui doit être différent de son support réel, sinon g n'est pas minimal.

TITANIC évite aussi l'accès au contexte d'extraction pour calculer les fermetures des générateurs minimaux fréquents.

5.2.6. Algorithme ZART

ZART a été proposé dans [32]. Un algorithme d'extraction d'itemset multifonctionnel. En effet, ZART affiche un certain nombre de fonctionnalités supplémentaires et effectue les tâches suivantes, généralement indépendantes :

a. Mécanisme de comptage par inférence : cette partie de Zart est basée sur Pascal [19], qui utilise les propriétés du comptage inférence. À partir d'un certain niveau, tous les générateurs peuvent être trouvés, ainsi tous les itemsets fréquents restants et leurs supports peuvent être déduits sans autre passage de base de transactions.

b. Identification les itemsets fermés fréquents : cette phase consiste à identifier les itemsets fermés fréquents parmi les itemset fréquents. Par définition : Un motif(Itemset) fréquent est dit fermé s'il ne possède aucun sur motif qui a le même support.

c. Associer les générateurs à leurs fermetures : lorsqu'un itemset fermé fréquent est trouvé, tous ses sous-ensembles fréquents sont déjà connus. Cela signifie que ses générateurs sont déjà calculés, ils doivent seulement être identifiés.

5.2.7. Algorithme Prince

L'algorithme PRINCE [33] met alors en place un mécanisme de gestion des classes d'équivalence permettant de générer la liste intégrale des motifs fermés fréquents sans duplication et sans recours aux tests de couvertures. Il permet aussi de réduire d'une manière

notable le coût du calcul des fermetures. De plus et grâce à cette structure partiellement ordonnée, PRINCE permet d'extraire les bases génériques de règles sans avoir à l'associer avec un autre algorithme. La construction des liens de précédence est optimisée grâce à une gestion efficace des classes d'équivalence ainsi que la détection d'information pouvant rendre partielle cette construction.

PRINCE prend en entrée un contexte d'extraction K, le seuil minimum de support minsupp et le seuil minimum de confiance minconf. Il donne en sortie la liste des motifs fermés fréquents et leurs générateurs minimaux respectifs ainsi que les bases génériques de règles. PRINCE opère en trois étapes successives:

a. Détermination des générateurs minimaux;
b. Construction du treillis des générateurs minimaux;
c. Extraction des bases génériques de règles.

5.2.8. Algorithme LCM

LCM (connu sous la terminologie anglaise Linear time Closed item set Miner) a été proposé dans [27]. Cet algorithme est consacré pour l'extraction des itemsets fermés. LCM se distingue des autres algorithmes de type backtrack, c'est à dire, qu'il énumère linéairement l'ensemble des itemsets fréquents fermés par un parcoure en profondeur, sans explorer des motifs fréquents non nécessaires. Pour ce faire, un arbre sous forme de trajets transversaux contenant seulement tous les motifs fermés fréquents est créé. Deux techniques ont été utilisées pour accélérer les mises à jour sur les occurrences des motifs:

a. **Occurrence deliver** : Cette technique calcule simultanément les ensembles d'occurrences de tous les successeurs du motif courant durant une, et une seule, itération de balayage sur l'ensemble d'occurrences actuel.
b. **Diffsets** : Cette technique a été introduite pour réduire l'utilisation de la mémoire des calculs intermédiaires.

L'algorithme LCM repose sur un parcours optimisé de l'espace de recherche exploitant le concept de « core prefix ». Il faut aussi pouvoir définir un ordre sur les items (l'ordre alphabétique, par exemple). Intuitivement, le core prefix d'un itemset fermé *I* sert de « noyau » d'extension pour générer un autre itemset fermé *I*'. Le core prefix d'un itemset *I* est le plus petit préfixe (selon l'ordre sur les items) qui apparaît dans toutes les transactions où apparaît *I*.

5.2.9. Algorithme MAFIA

MAFIA [34] est un algorithme pour extraire des ensembles d'éléments fréquents maximaux à partir d'une base de données transactionnelle. L'algorithme est particulièrement efficace lorsque les itemsets de la base de données sont très longs. La stratégie de recherche de l'algorithme intègre un parcours en profondeur du treillis d'itemset avec des mécanismes d'élagage efficaces.

MAFIA stocke efficacement la base de données transactionnelle sous la forme d'une série de bitmaps verticaux, où chaque bitmap représente un ensemble d'éléments dans la base de données et un bit dans chaque bitmap représente si un client donné possède ou non l'ensemble d'éléments correspondant. Initialement, chaque bitmap correspond à un 1-itemset, ou un seul élément. Les jeux d'éléments dont la fréquence est vérifiée dans la base de données deviennent de plus en plus longs de manière récursive, et la représentation bitmap verticale fonctionne parfaitement en conjonction avec cette extension de jeu d'éléments.

5.2.10. Algorithme CHARM

Charm (Closed Association Rule Mining, avec un H sans correspondance) [24] est un algorithme de la famille des méthodes verticales initiées avec Eclat. Il utilise donc, d'une part, des paires motifs-listes de transactions associées et décompose l'espace de recherche, représenté par un arbre, en plusieurs classes d'équivalence qui sont traitées séparément. D'autre part, il exploite l'astuce des diffsets proposée dans dEclat pour les calculs des supports. Étant donné que la découverte de MFF requière des tests de fermeture, cet algorithme introduit plusieurs mécanismes d'élagage permettant de sauter des niveaux complets de l'espace du problème et ainsi de générer moins de candidats à vérifier

5.2.11. Algorithme Hmine

H-Mine [35] est un algorithme d'extraction d'itemsets fréquents dans les bases de données de transactions. Contrairement aux algorithmes précédents tels qu'Apriori, H-Mine utilise une approche de croissance de modèle pour découvrir des ensembles d'éléments fréquents. L'entrée de H-Mine est une base de données de transactions (c'est-à-dire un contexte binaire) et un seuil nommé minsup (une valeur comprise entre 0 et 100 %). H-Mine est un algorithme de découverte d'itemsets (groupe d'items) apparaissant fréquemment dans une base de données de transactions (itemsets fréquents). H-mine a de meilleures performances d'exécution sur les données

clairsemées et denses que FP-growth et Apriori. H-mine a une meilleure utilisation de l'espace sur les données clairsemées et denses que FP-Growth et Apriori. H-mine fonctionne bien avec de très grandes bases de données.

6. Conclusion

Dans ce chapitre, nous avons présenté l'ensemble des notions de bases relatives à l'extraction des motifs et des règles d'association que nous utiliserons au chapitre suivant.

Nous avons présenté une vue d'ensemble de l'état de l'art des algorithmes les plus importants basés sur la découverte des itemsets fréquents et des itemsets fréquents fermés. Nous avons également mené une étude critique sur ces méthodes, en indiquant leurs principaux avantages et inconvénients.

Dans le chapitre suivant, nous comparerons certains des algorithmes les plus performants sur une base de données de patients diabétiques (768 patients avec 9 variables), ce qui constitue notre première contribution.

Chapitre 2 - Comparaison des algorithmes d'extraction des motifs fréquents

1. Introduction

L'extraction des motifs fréquents est une tâche importante en fouille de données, qui est l'objectif de la première contribution presentée dans ce chapitre. A ce niveau, une étude comparative sur les meilleures algorithmes d'extraction est réalisée, à savoir : Apriori [9], FP-Growth [18], Charm [24] et Eclat [17]. Cette étude est appliquée sur une base de données des personnes diabétiques afin de générer les règles d'association et de donner les interprétations nécessaires, et selon plusieurs critères, en l'occurrence : le temps d'exécution, l'espace de mémoire utilisé et le nombre d'itemsets générés.

2. Présentation des données

Le dataset **Pima Indian Diabetes**, originaire de l'institut national du diabète et des maladies digestives et rénales, contient des informations sur 768 femmes d'une population vivant en Arizona, aux États-Unis. La population Pima est étudiée par l'insitut à des intervalles de 2 ans depuis 1965. Le dataset PIMA comprend des informations sur des attributs qui pourraient et devraient être lié à l'apparition du diabète et à ses complications futures.

Plusieurs contraintes ont été placées sur la sélection de ces instances:

- Tous les patients sont des femmes.
- Tous les patients ont au moins 21 ans.
- Tous les patients sont d'origine indienne Pima.

L'objectif était de prédire, sur la base de mesures diagnostiques, si un patient est atteint de diabète, en dépendant de dimensions analytiques incorporées dans la base de données. Certaines restrictions ont été ajoutées par le choix du phénomène à partir du plus grand ensemble de données. La base de données a : 9 attributs, 768 enregistrements concernant des patientes femmes avec 500 cas négatifs (65,1 %) et 268 cas positifs (34,9 %) avec aucune valeur manquante. Le tableau 2.1 et le tableau 2.2 illustrent respectivement la description des attributs de notre dataset et le dataset après transformation.

Tableau 2.1 : Description des colonnes

Attribut	Description	Valeur
Pregnancie	Nombre de grossesses	[0-17]
Glucose	Concentration plasmatique de glucose dans un test de tolérance au glucose	[0-199]
BloodPressure	Pression sanguine diastolique (mmHg)	[0-122]
SkinThickness	Epaisseur du pli cutané du triceps	[0-99]
Insulin	Insuline sérique 2h (mu U/ml)	[0-846]
BMI	Indice de masse corporelle (Poids en Kg/(taille en m^2)	[0-67]
DiabetesPedigreeFunction	Fonction généalogique du diabète 0-2,4	[0-2.45]
Age	Age d'individu	[21-81]
Outcome	Testé positif/négatif	(0,1)

Tableau 2.2 : Dataset avant transformation

Pregnancie	Glucose	BloodPress	SkinThickn	Insulin	BMI	DiabetesPe	Age	Outcome
6	148	72	35	0	33.6	0.627	50	1
1	85	66	29	0	26.6	0.351	31	0
8	183	64	0	0	23.3	0.672	32	1
1	89	66	23	94	28.1	0.167	21	0
0	137	40	35	168	43.1	2.288	33	1
5	116	74	0	0	25.6	0.201	30	0
3	78	50	32	88	31	0.248	26	1

3. Description statistique

Dans l'ensemble de données, le résultat (Outcome) est la variable dépendante et les 8 variables restantes sont des variables indépendantes. Dans cette partie, nous procéderons à un examen préliminaire des statistiques descriptives pour l'ensemble de données. Les statistiques générales des différentes variables sont présentées dans le Tableau 2.3. Ce tableau illustre le nombre d'observations, la valeur minimale, la valeur maximale, la moyenne et la médiane de

toutes les variables quantitatives. On note que le nombre d'observations est égal au nombre d'instances qui vaut 768. Comme les paramètres biologiques tels que le glucose, la pression artérielle, l'épaisseur de la peau, l'insuline et l'IMC ne peuvent pas avoir de valeurs nulles, en effet les valeurs nulles aient été codées comme des zéros.

Tableau 2.3 : Statistiques générales pour les variables quantitatives

Statistique	Pregnancies	Glucose	Blood Pressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age
Nb.d'observations	768	768	768	768	768	768	768	768
Minimum	0,000	0,000	0,000	0,000	0,000	0,000	0,078	21,000
Maximum	17,000	199,000	122,000	99,000	846,00	67,10	2,420	81,000
Médiane	3,000	117,000	72,000	23,000	30,500	32,00	0,373	29,000
Moyenne	3,845	120,895	69,105	20,536	79,799	31,99	0,472	33,241

La moyenne d'âge est de 33,241 avec la valeur minimale et maximale comprise entre 21 et 81 ans. Alors que la médiane est de 29 ce qui montre que la variable âge ne suit pas une distribution normale mais elle est asymétrique (Figure 2.1). La variable IBM semble symétrique car la moyenne et la médiane sont approximativement égales ($32 \approx 31.99$). On note également une forte différence entre la médiane et la moyenne de la variable Insulin qui se traduit par une dissymétrie de la distribution. L'insuline moyenne est de 79,79 et sa médiane est de 30,5.

La variable quantitative « Outcome » est la variable dépendante binaire, elle prend les valeurs booléennes 0 et 1. 0 indique que le test est négatif (non diabétique) et le 1 indique que le test est positif (diabétique) (Tableau 2.4). Le graphique sur la figure 2.1 montre que les données sont déséquilibrées. Le nombre de non diabétiques est de 268 parcontre le nombre de patients diabétiques est de 500 (Tableau 2.4). Il y a 65 % de 1 (diabétique) et 35 % de 0 (non diabétique) dans les données.

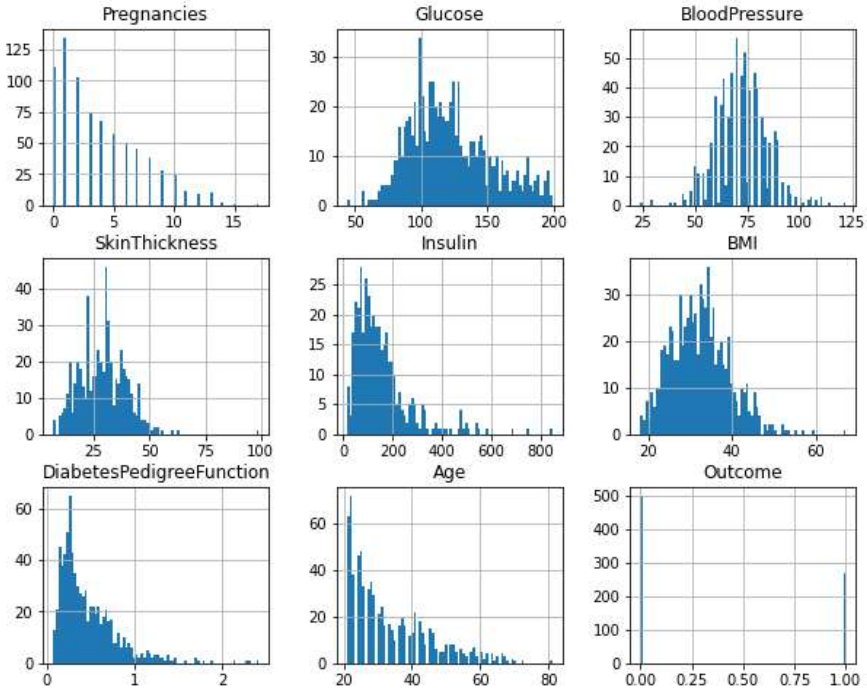

Figure 2.1. Histogrammes des variables

Tableau 2.4. Statistiques générales pour la variable qualitative (Outcome)

Variable\Statistique	Nb. d'observations	Nb. de modalités	Mode	Mode (effectif)	Modalités	Effectif par modalité	Fréquence par modalité (%)	Proportion par modalité
Outcome	768	2	0	500	0 (non diabétique)	500,000	65,104	0,651
					1 (Diabetic)	268,000	34,896	0,349

4. Corrélation

Un bon ensemble de données est un ensemble dans lequel les caractéristiques sont fortement corrélées à la classe cible et sont fortement non corrélées les unes aux autres. Pour trouver les attributs non corrélés, la sélection des caractéristiques se fait via une approche basée sur la corrélation utilisant un coefficient de corrélation.

→ Coefficient de corrélation : est un nombre qui indique la force de la relation entre deux variables. Il existe plusieurs types de coefficients de corrélation, mais le plus commun de tous est le coefficient de Pearson noté r, défini par : $r = Cov(X, Y) / \sigma_X \sigma_Y$

ou : Cov(X, Y) désigne la covariance des variables X et Y , σ_X et σ_Y désignent leurs écarts types . La valeur du coefficient de corrélation comprise entre -1 et +1.

→ 1 signifie qu'ils sont fortement corrélés (forte relation positive).

→ 0 signifie aucune corrélation.

→ -1 signifie qu'il existe une corrélation négative (forte relation négative). Le tableau 2.5 illustre la corrélation entre les différentes variables. Dans la carte thermique (figure 2.3), des couleurs plus vives indiquent une plus grande corrélation. Comme nous pouvons le voir dans le tableau 2.5 et la carte thermique, les niveaux de glucose, l'âge, l'IMC et le nombre de grossesses ont tous une corrélation significative avec la variable d'Outcome. Grâce au résultat d'analyse de corrélation les attributs Glucose, BMI et Age ont une relation de 0,47, 0,29, 0,24 avec Output. En analysant les valeurs de l'analyse de corrélation, Le glucose, l'IMC et l'âge ont la plus grande influence sur le diabète dans la corrélation entre les variables.

Figure.2.2. Distribution de la variable Outcome

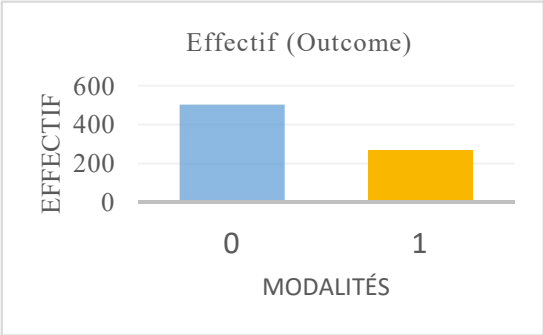

Tableau 2.5 : Corrélation de caractéristiques (et de Outcome)

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	PedigreeFunction	Age	Outcome
Pregnancies	1.000000	0.129459	0.141282	-0.081672	-0.073535	0.017683	-0.033523	0.544341	0.221898
Glucose	0.129459	1.000000	0.152590	0.057328	0.331357	0.221071	0.137337	0.263514	0.466581
BloodPressure	0.141282	0.152590	1.000000	0.207371	0.088933	0.281805	0.041265	0.239528	0.065068
SkinThickness	-0.081672	0.057328	0.207371	1.000000	0.436783	0.392573	0.183928	-0.113970	0.074752
Insulin	-0.073535	0.331357	0.088933	0.436783	1.000000	0.197859	0.183928	-0.042163	0.130548
BMI	0.017683	0.221071	0.281805	0.392573	0.197859	1.000000	0.140647	0.036242	0.292695
PedigreeFunction	-0.033523	0.137337	0.041265	0.183928	0.185071	0.140647	1.000000	0.033561	0.173844
Age	0.544341	0.263514	0.239528	-0.113970	-0.042163	0.036242	0.033561	1.000000	0.238356
Outcome	0.221898	0.466581	0.065068	0.074752	0.130548	0.292695	0.173844	0.238356	1.000000

En appliquant l'Analyse en Composante Principale (ACP) sur les données (Figure 2.4), les groupements de caractéristiques ont été obsevées dans le tracé ci-dessus, où les fléches qui sont proches les unes des autres representent des caractéristiques étroitement liées. On peut voir que les éléments suivant sont étroitement liés :

- Grossesse et âge
- Glycémie et tension arterielle
- IMC, DPF, niveau d'insuline et épaisseur de la peau

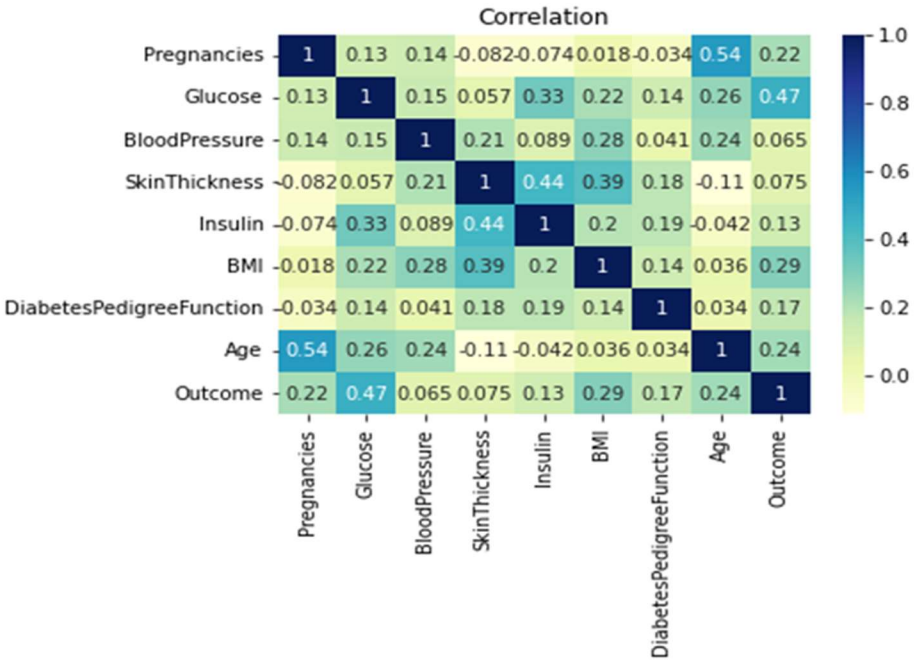

Figure 2.3. Carte thermique des corrélations de caractéristiques (et de Outcome)

Comme le montre le tableau 2.3 (la moyenne des caracteristiques), nous pouvons voir que les caractérstiques Glucose, IMC, âge, insuline sont très importants pour aider à classer les données. En revanche, le DPF, la tension artérielle et le nombre de grossesse sont très bas. Ces caractéristiques peuvent être utilisables par les algorithmes de classification de machine learning (ML).

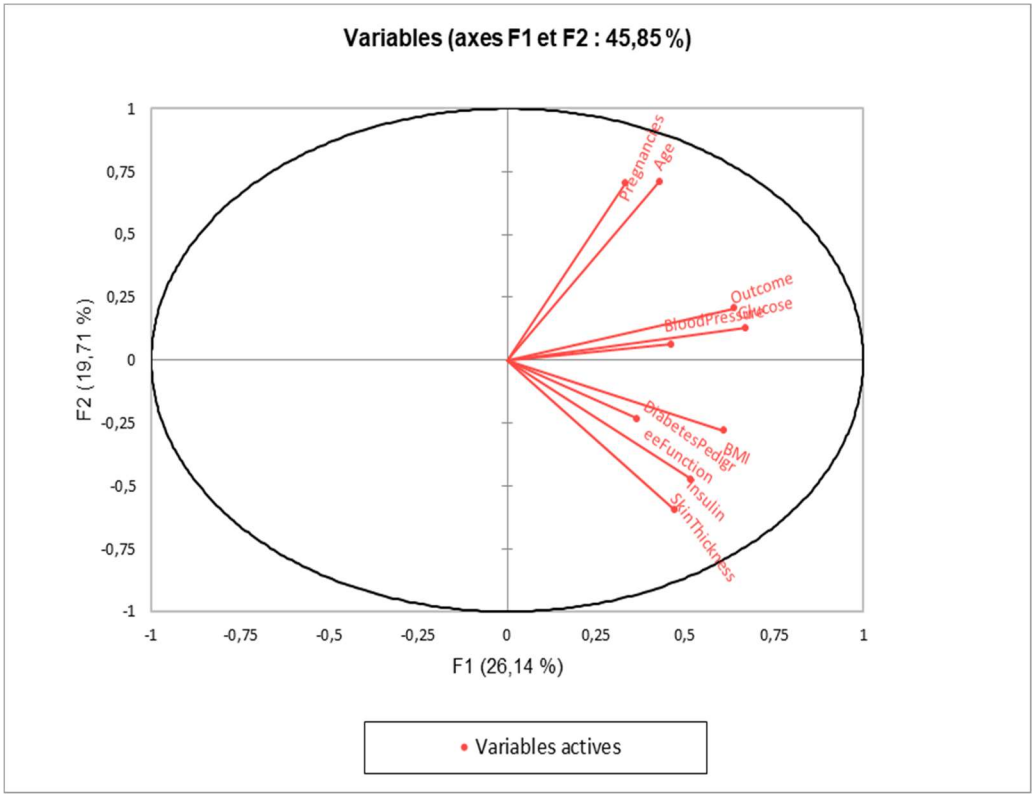

Figure 2.4 : Résultat de l'Analyse en Composante Principale

5. Transformation de données

Notre jeu de données contient uniquement les valeurs numériques, alors que la plupart des algorithmes d'extraction des motifs fréquents pour les règles d'association acceptent les jeux des données transactionnels. Dans la première étape du processus d'extraction des règles d'association, nous avons préparé les données par les transformations des données numériques à des données catégorielles en utilisant la technique d'intervalle de regroupement.

Le FP-Growth et ses variantes n'acceptent que les jeux de données transactionnels. L'ensemble de données sur le diabète (ensembles de données numériques) est transformé en un jeu de données transactionnel. Pour faire cette transformation chaque caractéristique est visualisée pour savoir comment elle évolue en fonction du nombre d'individus et pour diviser chaque caractéristique en domaines impliquant plusieurs individus.

Les résultats de la visualisation sont réalisés avec Matplotlib qui est l'un des packages Python les plus populaires utilisés pour la visualisation de données. Il permet de créer des tracés 2D à partir de données dans des tableaux. La visualisation de chaque attribut est présentée dans le graphique de la figure 2.5 pour l'attribut âge.

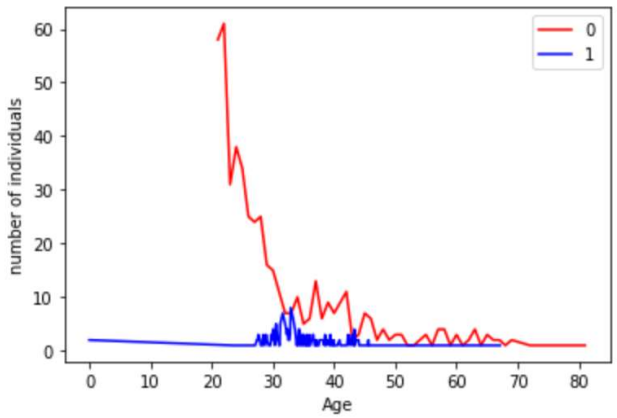

Figure 2.5: l'âge par rapport au nombre d'individus.

Comme nous pouvons le voir dans l'intervalle [20, 30], nous avons un nombre élevé de non-diabétiques par rapport au nombre d'individus diabétiques, et pour l'intervalle [30, 80], nous avons presque le même nombre d'individus pour les deux classes. 0 et 1 (0 : sans diabète, 1 : avec diabète), on peut donc simplement diviser la plage de la caractéristique en deux domaines : A1 : [0, 30] et A2 : [30, 80].

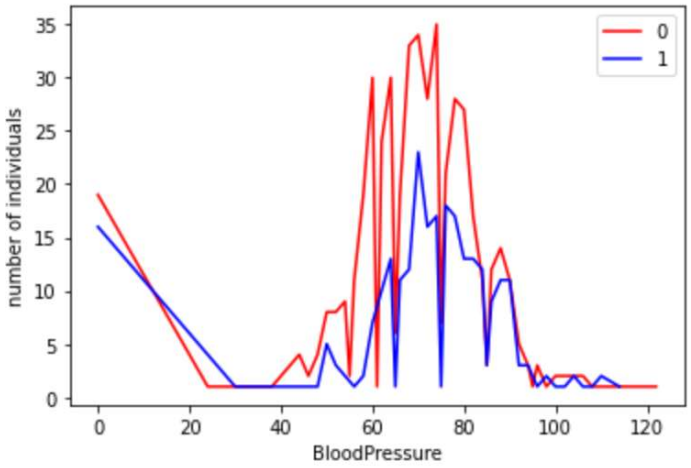

Figure 2.6: Tension artérielle par rapport au nombre d'individus.

La deuxième caractéristique est la pression artérielle, le résultat illustré à la Figure 2.6. Comme on peut le voir sur le graphique de la pression artérielle, cette caractéristique peut être divisée en trois domaines : B1 : [0, 40] ; B2 : [40, 90] ; B3 : [90, 120]. La troisième caractéristique est l'IMC, résultat de la visualisation illustrée dans la Figure 2.7.

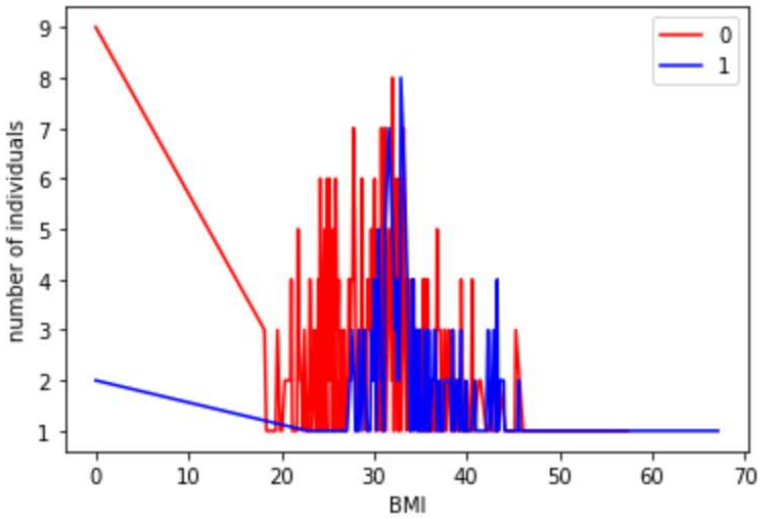

Figure 2.7: IMC par rapport au nombre d'individus.

Comme on le voit sur ce graphique, on peut diviser la plage de la fonction IMC en deux domaines, le premier IMC1 : [0, 30], où l'on a le nombre d'individus de la classe 0 supérieur à la classe 1, et le second est l'IMC2 : [30, 60] où les deux classes ont presque la même variation.

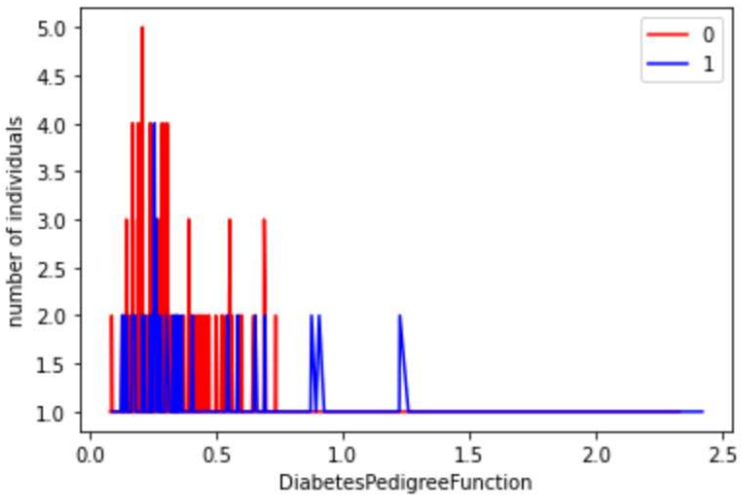

Figure 2.8 : Diabetes PredgreeFonction par rapport au nombre d'individus.

La fonction Diabetes Pedigree est divisée en deux domaines : D1 : [0, 0,8] et D2 : [0,8, 2,5]. (Figure 2.8).

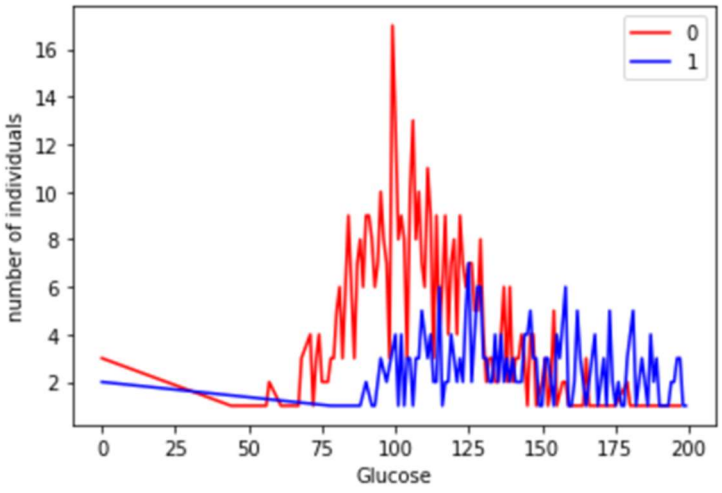

Figure 2.9 : Glucose par rapport au nombre d'individus.

On peut diviser la gamme caractéristique en deux domaines : G1 : [0, 125] et G2 : [125, 200] (Figure 2.9).

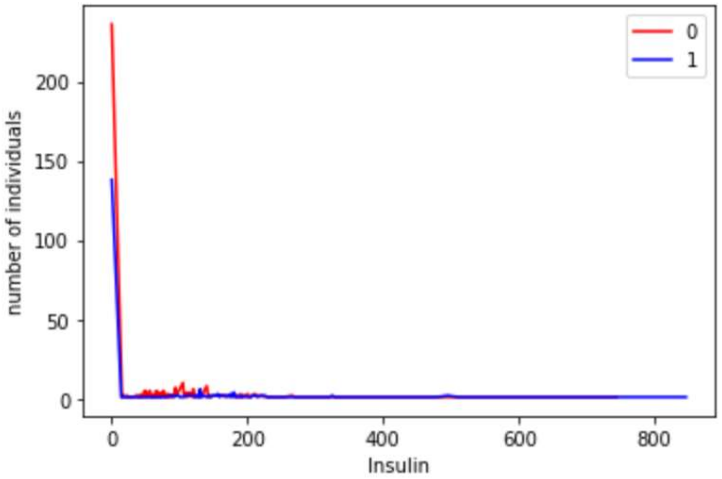

Figure 2.10 : Insuline par rapport au nombre d'individus.

Dans le graphique Insuline vs Insuline du nombre d'individus (Figure 2.10) dans l'intervalle [0, 30] nous avons presque la même variation pour les deux classes 0 et 1, et dans [30, 150] la classe 0 est supérieure à la classe 1, et dans l'intervalle [150, 800] on a aussi la même variation des classes 0 et 1, donc on divise cette caractéristique en trois domaines : I1 : [0, 30] ; I2 : [30 150] ; I3 : [150, 800]. La septième fonctionnalité est Grossesses, le résultat de la visualisation est illustré dans la figure 2.11.

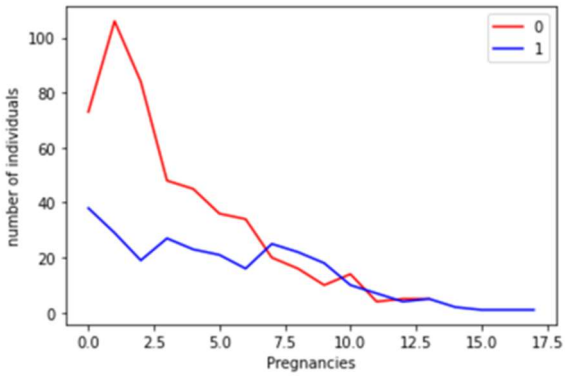

Figure 2.11 : Grossesses par rapport au nombre d'individus.

On peut diviser la plage de la fonction Grossesses en deux domaines, le premier est P1 : [0, 7], où l'on a le nombre d'individus de la classe 0 supérieur à la classe 1, et le second est P2 : [7, 17] où les deux classes ont presque la même variation.

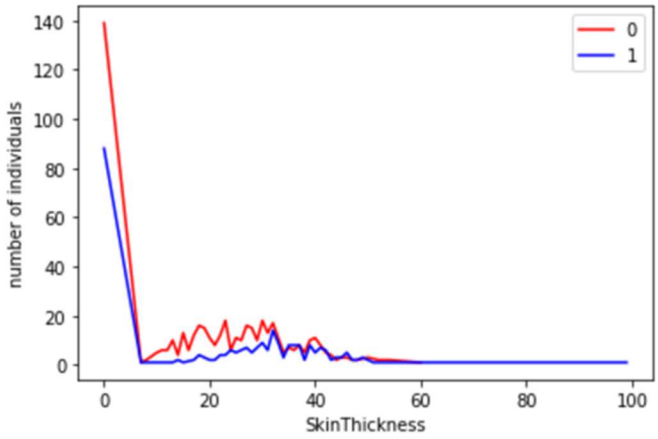

Figure 2.12 : Épaisseur de la peau par rapport au nombre d'individus.

Dans la figure 2.12, nous pouvons voir dans [0, 8], presque la même variation pour les deux classes, et dans [8, 45] la classe 0 a presque le plus grand nombre d'individus que la classe 1. De plus, pour la gamme [45, 60], on a aussi la même variation pour les deux classes, on peut donc scinder la caractéristique en trois domaines : S1 : [0, 8], S2 : [8, 45] et S3 : [45, 60].

6. Résultat de la transformation

Après toutes ces analyses, nous pouvons résumer brièvement toutes les informations dans le tableau 2.6. La nouvelle base de données est illustrée dans la figure 2.13. Certains algorithmes acceptant que des valeurs numériques comme le montre le tableau 2.7.

Tableau 2.6 : Résultat de la transformation

Variable	Intervalle
Age	A1 : [0, 30] ; A2 : [30, 80]
Grossesses	P1 : [0, 7] ; P2 : [7, 17]
Glucose	G1 : [0, 125] ; G2 : [125, 200]
Pression artérielle	B1 : [0, 40] ; B2 : [40, 90] ; B3 : [90, 120]
Épaisseur de la peau	S1 : [0, 8] ; S2 : [8, 45] ; S3 : [45, 60]
Insuline	I1 : [0, 30] ; I2 : [30, 150] ; I3[150, 800]
Fonction Pedigree du diabète	D1 : [0, 0,8] ; D2 : [0, 2,5]
IMC	IMC1 : [0, 30] ; IMC2 : [30, 60]

```
Out[5]: [['P1', 'G2', 'B2', 'S2', 'I1', 'BMI2', 'D1', 'A2', '1'],
         ['P1', 'G1', 'B2', 'S2', 'I1', 'BMI1', 'D1', 'A2', '0'],
         ['P2', 'G2', 'B2', 'S1', 'I1', 'BMI1', 'D1', 'A2', '1'],
         ['P1', 'G1', 'B2', 'S2', 'I2', 'BMI1', 'D1', 'A1', '0'],
         ['P1', 'G2', 'B1', 'S2', 'I3', 'BMI2', 'D2', 'A2', '1'],
         ['P1', 'G1', 'B2', 'S1', 'I1', 'BMI1', 'D1', 'A1', '0'],
         ['P1', 'G1', 'B2', 'S2', 'I2', 'BMI2', 'D1', 'A1', '1'],
         ['P2', 'G1', 'B1', 'S1', 'I1', 'BMI2', 'D1', 'A1', '0'],
         ['P1', 'G2', 'B2', 'S2', 'I3', 'BMI2', 'D1', 'A2', '1'],
         ['P2', 'G1', 'B3', 'S1', 'I1', 'BMI1', 'D1', 'A2', '1'],
         ['P1', 'G1', 'B3', 'S1', 'I1', 'BMI2', 'D1', 'A1', '0'],
         ['P2', 'G2', 'B2', 'S1', 'I1', 'BMI2', 'D1', 'A2', '1'],
         ['P2', 'G2', 'B2', 'S1', 'I1', 'BMI1', 'D2', 'A2', '0'],
         ['P1', 'G2', 'B2', 'S2', 'I3', 'BMI2', 'D1', 'A2', '1'],
         ['P1', 'G2', 'B2', 'S2', 'I3', 'BMI1', 'D1', 'A2', '1'],
         ['P1', 'G1', 'B1', 'S1', 'I1', 'BMI1', 'D1', 'A2', '1'],
         ['P1', 'G1', 'B2', 'S3', 'I3', 'BMI2', 'D1', 'A2', '1'],
         ['P1', 'G1', 'B2', 'S1', 'I1', 'BMI1', 'D1', 'A2', '1'],
         ['P1', 'G1', 'B1', 'S2', 'I2', 'BMI2', 'D1', 'A2', '0'],
```

Figure 2.13 : Nouvelle dataset

Tableau 2.7 : Transformation de la base de données Diabète

Variable	Intervalle
Age	11[0-30] et 12[30-80]
Pregnancies	21[0-7] et 22[7-17]
Glucose	31[0-125] et 32[125-200]
BloodPressure	41[0-40] et 42[40-90] et 43[90-120]
SkinThickness	51[0-8] et 52[8-45] et 53[45-60]
Insulin	61[0-30] et 62[30-150] et 63[150-800]
DiabetesPedigreeFunction	71[0-0.8] et 72[0.8-5]
MBI	81[0-30] et 82[30-60]

Le résultat de la transformation est illustré à la Figure 2.14

```
[['21', '32', '42', '52', '61', '82', '71', '12', '1'],
 ['21', '31', '42', '52', '61', '81', '71', '12', '0'],
 ['22', '32', '42', '51', '61', '81', '71', '12', '1'],
 ['21', '31', '42', '52', '62', '81', '71', '11', '0'],
 ['21', '32', '41', '52', '63', '82', '72', '12', '1'],
 ['21', '31', '42', '51', '61', '81', '71', '11', '0'],
 ['21', '31', '42', '52', '62', '82', '71', '11', '1'],
 ['22', '31', '41', '51', '61', '82', '71', '11', '0'],
 ['21', '32', '42', '52', '63', '82', '71', '12', '1'],
 ['22', '31', '43', '51', '61', '81', '71', '12', '1'],
 ['21', '31', '43', '51', '61', '82', '71', '11', '0'],
 ['22', '32', '42', '51', '61', '82', '71', '12', '1'],
 ['22', '32', '42', '51', '61', '81', '72', '12', '0'],
 ['21', '32', '42', '52', '63', '82', '71', '12', '1'],
 ['21', '32', '42', '52', '63', '81', '71', '12', '1'],
 ['21', '31', '41', '51', '61', '81', '71', '12', '1'],
 ['21', '31', '42', '53', '63', '82', '71', '12', '1'],
 ['21', '31', '42', '51', '61', '81', '71', '12', '1'],
 ['21', '31', '41', '52', '62', '82', '71', '12', '0'],
 ['21', '31', '42', '52', '62', '82', '71', '12', '1'],
 ['21', '32', '42', '52', '63', '82', '71', '11', '0'],
 ['22', '31', '42', '51', '61', '82', '71', '12', '0'],
 ['21', '32', '42', '51', '61', '82', '71', '12', '1'],
 ['22', '31', '42', '52', '61', '81', '71', '11', '1'],
 ['22', '32', '43', '52', '62', '82', '71', '12', '1'],
 ['22', '31', '42', '52', '62', '82', '71', '12', '1'],
 ['21', '32', '42', '51', '61', '82', '71', '12', '1'],
 ['21', '31', '42', '52', '62', '81', '71', '11', '0'],
```

Figure 2.14 : Nouvelle dataset après transformation

7. Performance du temps d'exécution

Le but de cette section est d'estimer la performance de vitesse en analysant le temps d'exécution des algorithmes. La figure 2.15 illustre le temps d'exécution des algorithmes en fonction du support minimal qui varie de 0.02 à 0.12. Le tableau 2.8 résume les valeurs obtenues.

Tableau 2.8 : Temps d'exécution (en s) en fonction de minsup

minsupp	Apriori	FP-Growth	CFP-Growth	CHARM	ECLAT
0,02	12,1239	0,435	9,347	40,34	0,102
0,05	2,99	0,289	6,791	28,2	0,109
0,07	1,87	0,318	7,93	18,84	0,074
0,09	1,29	0,321	5,239	13,29	0,094
0,12	0,699	0,21	4,599	9,83	0,141
0,15	0,553	0,173	4,291	7,39	0,142
0,17	0,402	0,2	4,385	6,08	0,138
0,19	0,328	0,13	1,75	4,18	0,09

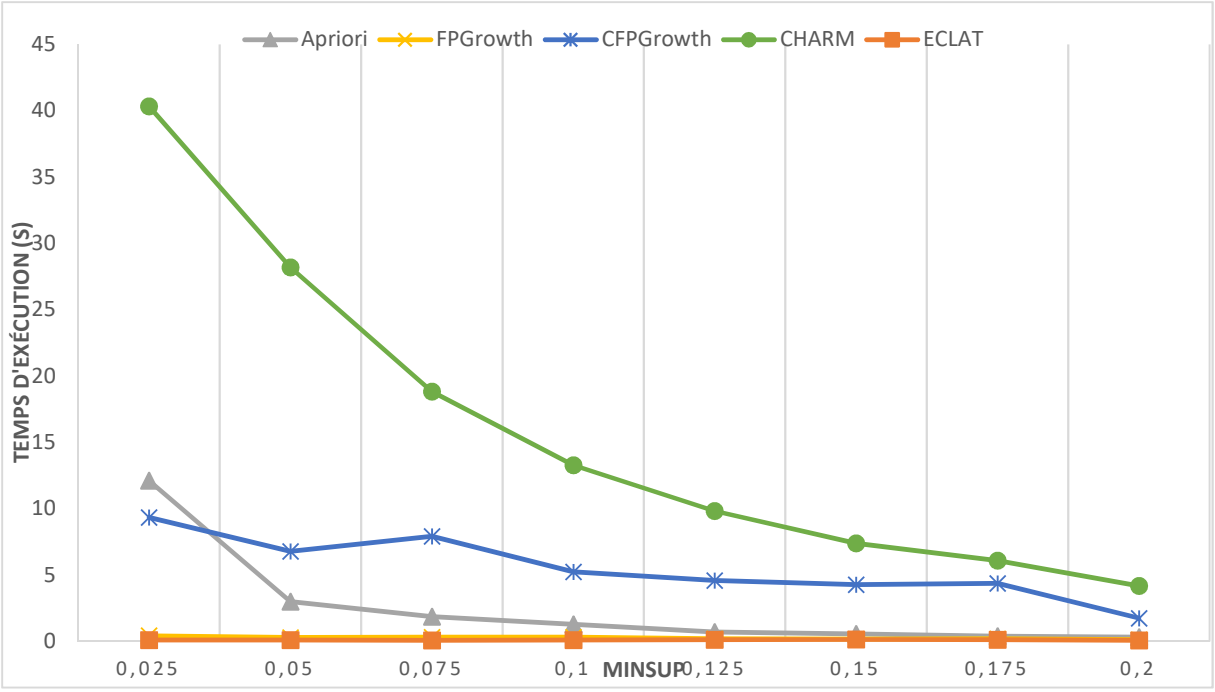

Figure 2.15. Temps d'exécution en fonction du minsup

En général, on remarque que le temps d'exécution de tous les algorithmes décroît en fonction du support minimum. Plus que le support minimal augmente, l'algorithme devient plus rapide. Les résultats montrent que les algorithmes Eclat et FP-Growth sont plus rapides par rapport à l'algorithme CHARM qui est clairement le plus lent. Apriori et CFP-Growth ont également donné des résultats plus au moins satisfaisants en terme de temps d'exécution.

8. Performance de l'espace mémoire

En analysant l'espace de mémoire des algorithmes, la figure 2.16 montre l'espace de mémoire spécifique en KB en fonction du support minimal presenté dans le tableau 2.9.

Tableau 2.9 : Espace mémoire (en KB) en fonction de minsup

minsupp	Apriori	FP-Growth	CFP-Growth	CHARM	ECLAT
0,02	3804,81	1634,58	26993,60	6978,63	1197,76
0,05	1945,14	1098,67	25870,01	4580,54	1263,20
0,07	1179,98	931,98	25718,95	4047,60	1127,18
0,09	783,26	795,42	22714,31	2963,79	1005,53
0,12	555,95	794,87	22675,80	2587,56	996,23

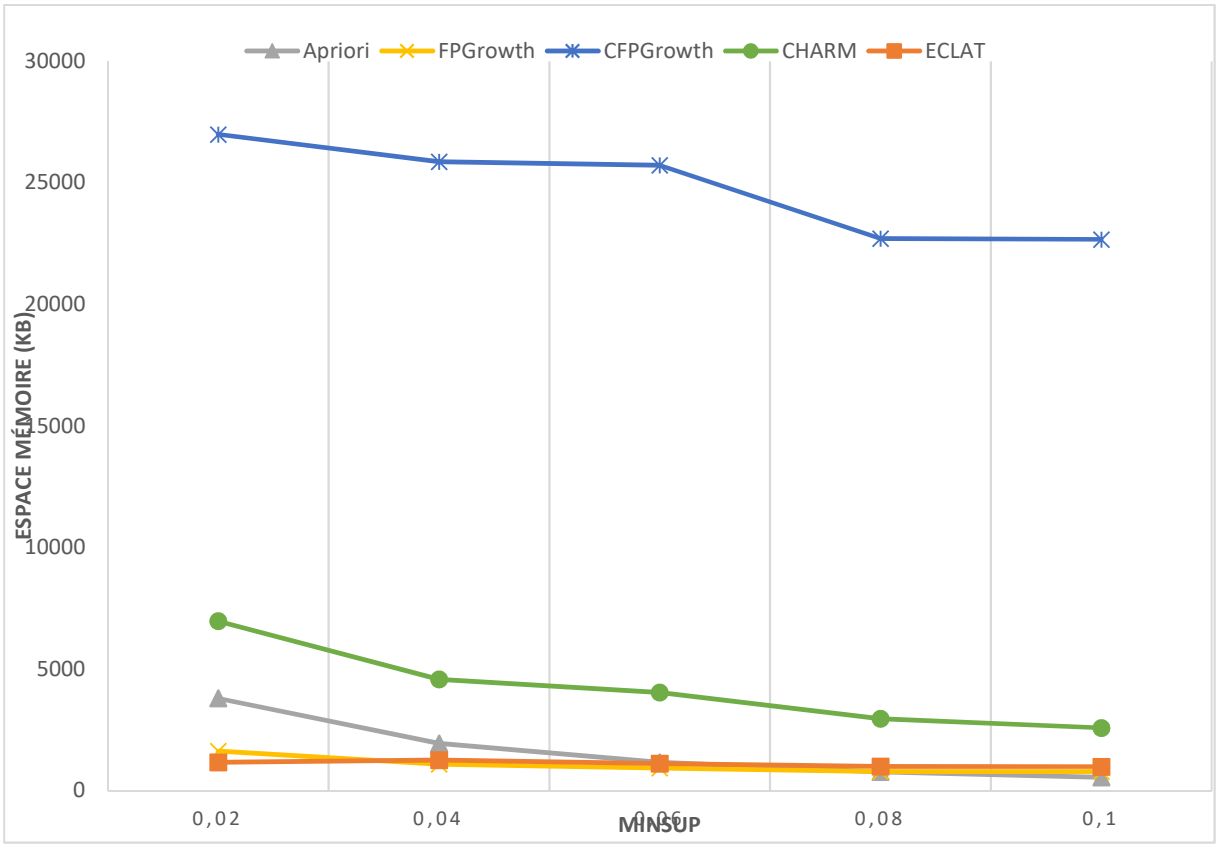

Figure 2.16 : Espace mémoire occupé en fonction du minsup

Nous pouvons facilement remarquer que l'algorithme qui consomme plus de mémoire est l'algorithme CFP-Growth, et que plus que le support minimal est grand plus la consommation de la mémoire se baisse.

9. Performance du nombre d'items générés

Une autre comparaison est faite sur le nombre d'itemsets générés par les algorithmes Apriori, FP-Growth, CFP-Growth, CHARM et ECLAT en fonction du support minimal comme il est décrit dans le tableau 2.11 et illustrés sur la figure 2.17. En effet, plus que le support minimum augmente, le nombre des itemsets diminue. Apriori et FP-Growth ont les mêmes les nombres des itemsets pour les différentes valeurs du support minimal. En effet les deux courbes représentent la même variation. Le tableau 2.10 montre le nombre des itemsets générées pour chaque valeur du support minimal allant de 0.02 à 0.12.

Tableau 2.10 : Nombre d'itemsets générés en fonction du minsup

minsupp	Apriori	FP-Growth	CFP-Growth	CHARM	ECLAT
0,02	5359	5359	3670	3742	14728
0,05	2915	2915	1846	2324	9624
0,07	1930	1930	1040	1605	7924
0,09	1326	1326	675	1151	6456
0,12	965	965	429	862	5665

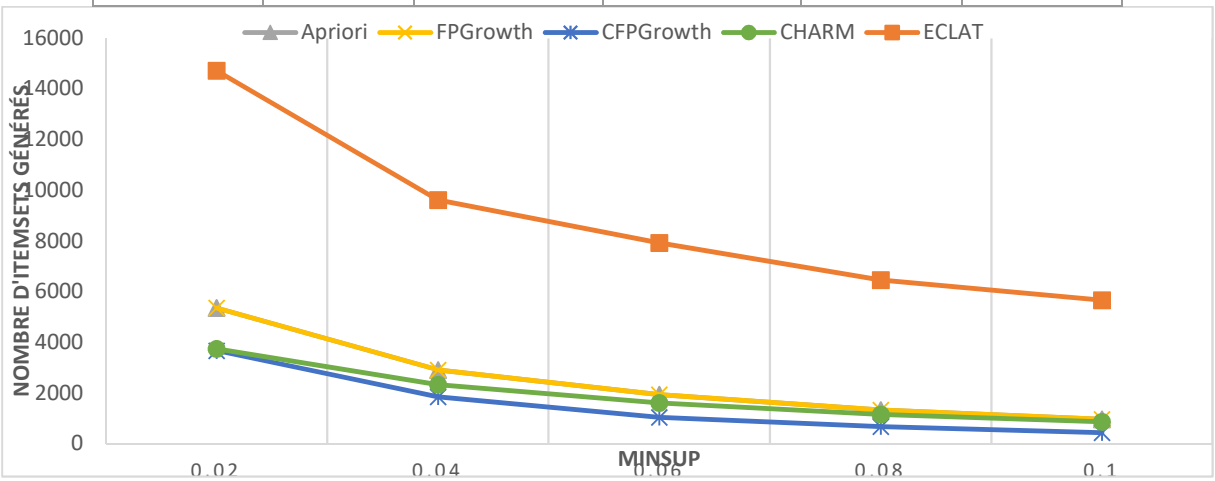

Figure 2.17 : Nombre d'itemsets générés en fonction du minsup

10. Règles d'association extraites

Nous utilisons le seuil de support minimum de 0.5 et nous fixons 60% comme seuil de confiance, nous obtenons les règles d'association présentées dans le tableau 2.11.

Tableau 2.11 : Règles d'association générées du dataset "Diabete"

	Règle	Conf	Supp	Lift	Conviction
1	['A1'] => ['P1']	99.4 %	54.07 %	1.84	76.16
2	['P1', 'S2'] => ['B2']	96.06 %	55.54 %	1.73	10.7
3	['D1', 'S2'] => ['B2']	95.73 %	54.72 %	1.75	10.02
4	['S2'] => ['B2']	95.61 %	63.84 %	1.5	7.57
5	['D1', '0'] => ['B2']	92.22 %	54.07 %	1.71	5.32
6	['0'] => ['B2']	91.77 %	59.93 %	1.53	4.22
7	['0', 'P1'] => ['B2']	91.39 %	53.58 %	1.71	4.8
8	['BMI2'] => ['B2']	91.08 %	54.89 %	1.66	4.46
9	['G1'] => ['B2']	90.77 %	56.03 %	1.62	4.14
10	['D1', 'P1'] => ['B2']	90.27 %	66.45 %	1.36	2.71

11	['B2', '0'] => ['D1']	90.22 %	54.07 %	1.67	4.1
12	['D1'] => ['B2']	90.15 %	77.52 %	1.16	1.42
13	['P1'] => ['B2']	89.81 %	76.06 %	1.18	1.5
14	['0'] => ['P1']	89.78 %	58.63 %	1.53	3.39
15	['0'] => ['D1']	89.78 %	58.63 %	1.53	3.39
16	['0', 'P1'] => ['D1']	89.72 %	52.61 %	1.71	4.03
17	['D1', '0'] => ['P1']	89.72 %	52.61 %	1.71	4.03
18	['B2', '0'] => ['P1']	89.4 %	53.58 %	1.67	3.78
19	['G1'] => ['D1']	87.6 %	54.07 %	1.62	3.09
20	['B2', 'P1'] => ['D1']	87.37 %	66.45 %	1.31	1.9
21	['G1'] => ['P1']	87.07 %	53.75 %	1.62	2.96
22	['B2', 'S2'] => ['P1']	86.99 %	55.54 %	1.57	2.78
23	['P1'] => ['D1']	86.92 %	73.62 %	1.18	1.17
24	['S2'] => ['P1']	86.59 %	57.82 %	1.5	2.48
25	['B2'] => ['D1']	86.23 %	77.52 %	1.11	0.73
26	['B2', 'S2'] => ['D1']	85.71 %	54.72 %	1.57	2.53
27	['D1', 'B2'] => ['P1']	85.71 %	66.45 %	1.29	1.57
28	['S2'] => ['D1']	85.61 %	57.17 %	1.5	2.31
29	['D1'] => ['P1']	85.61 %	73.62 %	1.16	0.97
30	['B2'] => ['P1']	84.60 %	76.06 %	1.11	0.66
31	['BMI2'] => ['D1']	83.51 %	50.33 %	1.66	2.41
32	['B2', 'P1'] => ['S2']	73.02 %	55.54 %	1.31	0.89
33	['D1', 'P1'] => ['0']	71.46 %	52.61 %	1.36	0.92
34	['B2'] => ['S2']	71.01 %	63.84 %	1.11	0.35
35	['D1', 'B2'] => ['S2']	70.59 %	54.72 %	1.29	0.76
36	['B2', 'P1'] => ['0']	70.45 %	53.58 %	1.31	0.81
37	['D1', 'B2'] => ['0']	69.75 %	54.07 %	1.29	0.74
38	['P1'] => ['0']	69.23 %	58.63 %	1.18	0.5
39	['P1'] => ['S2']	68.27 %	57.82 %	1.18	0.48
40	['D1'] => ['0']	68.18 %	58.63 %	1.16	0.44
41	['B2'] => ['0']	66.67 %	59.93 %	1.11	0.3
42	['D1'] => ['S2']	66.48 %	57.17 %	1.16	0.42
43	['P1'] => ['A1']	63.85 %	54.07 %	1.18	0.42
45	['P1'] => ['G1']	63.46 %	53.75 %	1.18	0.42
46	['D1'] => ['G1']	62.88 %	54.07 %	1.16	0.38
47	['B2'] => ['G1']	62.32 %	56.03 %	1.11	0.27
48	['B2'] => ['BMI2']	61.05 %	54.89 %	1.11	0.26

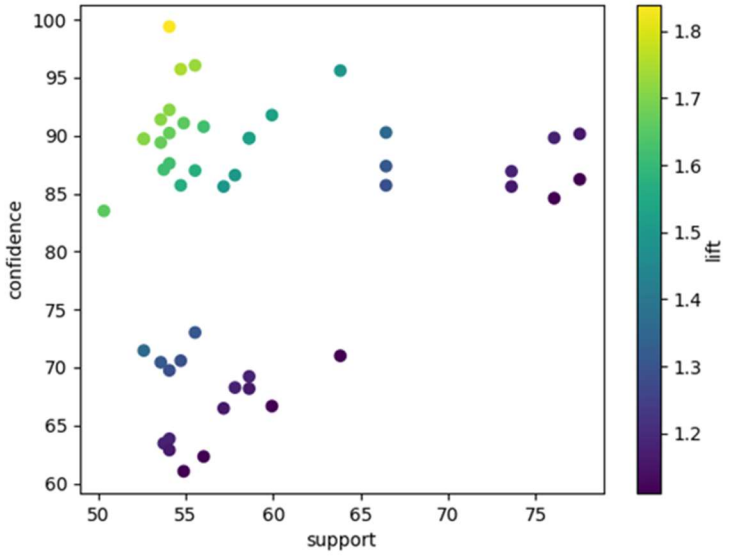

Figure 2.18 : Visualisation des règles

Plusieurs mesures de qualité des règles d'association ont été proposées dans la littérature. Nous avons choisi les plus connues d'entre elles : confiance, support, lit et la conviction. Un « lift » supérieur à 1 traduit une corrélation positive de X et Y, et donc le caractère significatif de l'association. La figure 2.18 illustre la visualisation des règles.

11. Interprétation des règles

Les règles générées ont majoritairement des supports assez proches compris entre 54.07% et 54.89%. Les valeurs de lift de ces règles sont toutes supérieures à 1 ce qui indique qu'il existe bien un lien entre les éléments de chaque règle. Les règles extraites peuvent être interprétées comme la probabilité conditionnelle d'incidence du diabète au sein d'une sous-population particulière. Par exemple, dans le Tableau 2.11 la règle 1 peut être interprétée comme suit : La probabilité de diabète chez les femmes âgées de A1= [0-30] et ont un nombre de grossesse appartient à l'intervalle P1= [0-7] correspond à 99.4 %. La valeur lift de cette règle peut être interprétée comme le rapport de cette probabilité (99.4 %) à la probabilité de diabète dans toute la population de la base de données et qui vaut 1.84. La conviction mesure aussi la dépendance mais pour les contre exemples, est plus grande pour cette règle ce qui montre que les deux attributs Age(A1) et Pregnancies(P1) sont positivement dépendants. La règle 8 fait ressortir le fait que l'obésité BMI2 peut affecter la pression artérielle B2, et cette règle est supportée par

97.08% des femmes. L'obésité chez les femmes est le facteur de risque le plus important, elle peut évoluer vers le diabète. La tension artérielle B2 et l'épaisseur de la peau S2 supérieur à 40 et 8 respectivement apparaissant dans deux règles 22 et 26. Au fur et à mesure que nous parcourons cette liste de règles, nous obtenons un degré de confiance plus faible. En ce qui concerne la conviction, la règle 1, 2 et 3 celles qui ont grande par rapport aux autres règles, cela indique que le nombre des contre exemples de la règle est inférieur à celui attendu dans hypothèse d'indépendance.

12. Conclusion

Dans ce chapitre, nous avons décrit la configuration expérimentale et avons ensuite présenté nos expérimentations. Pour chacun des algorithmes utilisés, nous avons décrit leurs techniques de fonctionnement ainsi que leurs propriétés. Nous avons présenté des graphiques de performance en termes de temps d'exécution, espace de mémoire utilisée et le nombre d'itemsets générés en fonction du support minimal. Finalement nous avons cité les règles d'association extraites et leurs interprétations. Les algorithmes décrits dans ce chapitre presentent un incovenient concernant le temps d'exécution lors de leur utilisation des données massives. Pour remedier à cet incovenient, nous avons fait appel à la Framework Spark.

Chapitre 3 - Extraction des Itemsets frequents avec Spark en Big data

1. Introduction.

L'object de ce chapitre est l'extraction des itemsets fréquents en utilisant Framework Spark de traitement des données en Big data par l'utilisation de trois algorithmes. Big Data se réfère souvent simplement à l'utilisation de l'analyse prédictive, de l'analyse du comportement des utilisateurs ou de toute autre méthode avancée d'analyse de données pour extraire de la valeur des données, et rarement à une taille particulière de l'ensemble de données. La précision des méga données peut conduire à une prise de décision plus confiante et de meilleures décisions peuvent se traduire par une plus grande efficacité opérationnelle et une réduction des coûts et des risques [37, 38,39].

2. Framework Spark.

Le modèle de programmation de Spark est basé sur une nouvelle abstraction de mémoire distribuée appelée Resilient Distributed Datasets (RDD), proposée pour les calculs en mémoire sur un grand cluster [40]. Un RDD est une collection immuable d'enregistrements de données. Il peut fournir une variété d'opérations intégrées pour transformer un RDD en un autre RDD. Spark mettra en cache le contenu des RDD en mémoire sur les nœuds de travail, ce qui accélérera considérablement la réutilisation des données. Les RDD peuvent atteindre une tolérance aux pannes basée sur les informations de lignage plutôt que sur la réplication. Spark suit suffisamment d'informations pour reconstruire les RDD en cas de défaillance d'un nœud. Le Framework Spark exécutera le code comme suit:

✓ L'exécution du programme commence au niveau du pilote (Driver), qui orchestre l'exécution réelle sur de nombreux serveurs de travail au sein du cluster ;

✓ Les données ne sont plus transférées au programme pilote et le programme pilote fonctionne avec des références de données plutôt qu'avec les données elles-mêmes. Ces références de données sont des identifiants pour localiser les données correspondantes résidant sur les serveurs de travail ;

✓ Le code est ensuite déplacé du programme pilote vers les travailleurs. Là, l'exécution se produit et les données sont modifiées en serveurs de travail chaque fois que possible sans quitter la machine ;

✓ Enfin, le programme pilote demande les données modifiées qui résident sur les nœuds de travail et seuls les résultats seront transférés au programme pilote.

Spark comprend deux parties importantes :

- Un modèle de programmation qui crée un graphe de dépendances ;
- Un système d'exécution qui utilise ce graphique pour planifier des unités de travail sur le cluster. Le système d'exécution distribue également le code aux nœuds de calcul et collecte les résultats finaux.

Au cœur du modèle de programmation Spark se trouvent les RDD [40], qui sont des abstractions d'un jeu de données distribué sur les nœuds de travail. Les opérations dans spark transformeront ces RDD comme s'il s'agissait de variables locales, tandis que les opérations réelles seront réparties sur les travailleurs (workers).

Spark peut optimiser le graphique de dépendance pour réduire la quantité de mouvement de données entre les nœuds. Ceci est essentiel car il existe deux types d'opérations. Celles qui ne nécessitent aucune communication entre les nœuds, comme :

- *map ()* qui transforme chaque élément du RDD en isolation ou ;
- *filter ()* qui fait une sélection des éléments en fonction sur un prédicat.

D'autres nécessitent des nœuds pour communiquer comme, par exemple *groupByKey ()* qui regroupera les éléments par une clé ou joindre deux RDD, qui nécessite qu'au moins un des RDD soit mis en cache ou persistant. Le choix de l'ordre et de la mise en œuvre effective de ces opérations conduit à bon escient à une réduction du coût de calcul.

3. Algorithmes traditionnels pour l'extraction des motifs fréquents.

3.1. Algorithme Apriori

L'algorithme Apriori proposé par Agrawal et ses co-auteurs en 1994 [19] est un algorithme de base qui permet d'extraire des motifs fréquents dans une base ayant plusieurs milliers d'attributs et plusieurs millions d'enregistrements. L'idée est d'effectuer une extraction par niveaux selon le principe suivant :

i. On commence par chercher les motifs fréquents de longueur 1 ;
ii. On combine ces motifs pour obtenir des motifs de longueur 2 et on ne garde que les fréquents parmi eux ;
iii. On combine ces motifs pour obtenir des motifs de longueur 3 et on ne garde que les fréquents parmi eux ;
iv. Continuer jusqu'à la longueur maximale.

Cette approche s'appuie sur les deux principes fondamentaux suivants (qui reposent sur la décroissance du support) :

- Tout sous-motif d'un motif fréquent est fréquent.
- Tout sur-motif d'un motif non fréquent est non fréquent.

Le pseudocode suivant décrit l'extraction de motifs fréquents selon ce principe:

Algorithme Apriori

Require: Base de données de transactions D, Seuil de support minimum σ
Ensure: Ensemble des items fréquents
$i \leftarrow 1$
$C1 \leftarrow$ ensemble des motifs de taille 1 (un seul item)
Tant que $Ci = \phi$ **do**
Calculer le Support de chaque motif $m \in Ci$ dans la base
$Fi \leftarrow \{m \in Ci | support(m) \geq \sigma\}$
$Ci+1 \leftarrow$ toutes les combinaisons possibles des motifs de Fi de taille $i + 1$
$i \leftarrow i + 1$
Fin tant que
Retourner $\cup(i \geq 1) Fi$

Afin d'illustrer les différentes étapes de l'algorithme Apriori on l'applique pour la base de données transactionnelle de l'exemple de la figure 3.1 pour extraire les itemsets fréquents en utilisant un support minimum S = 0.4. Ces processus sont répétés de manière itérative jusqu'à ce que les Itemset candidats ou les grands Itemset deviennent vides comme dans l'exemple illustré à la (Figure 3.1). Il y a deux limitations de cet algorithme : l'une est un processus complexe de génération d'éléments candidats qui consomme un grand espace mémoire et un temps d'exécution énorme et le deuxième problème est le nombre excessif d'analyses de base de données pour la génération de candidats. En général, dans le reste de ce travail on va découvrir des algorithmes parallèles qui sont basés sur l'algorithme Apriori et qui sont dédié aux Big data et qui vont être appliqué en Spark.

3.2. Algorithme FP-Growth

Dans le but de répondre au problème posé par l'algorithme de base Apriori, Han et al. [43] ont proposé l'algorithme FP-growth (Frequent Pattern growth) qui utilise une nouvelle technique pour générer les itemsets fréquents sans avoir extraire les itemsets candidats. Elle consiste d'abord à compresser la base de données en une structure compacte appelée FP-tree puis à diviser la base de données en sous-projections de la base de données appelées bases conditionnelles. Chacune de ces projections est associée à un item fréquent.

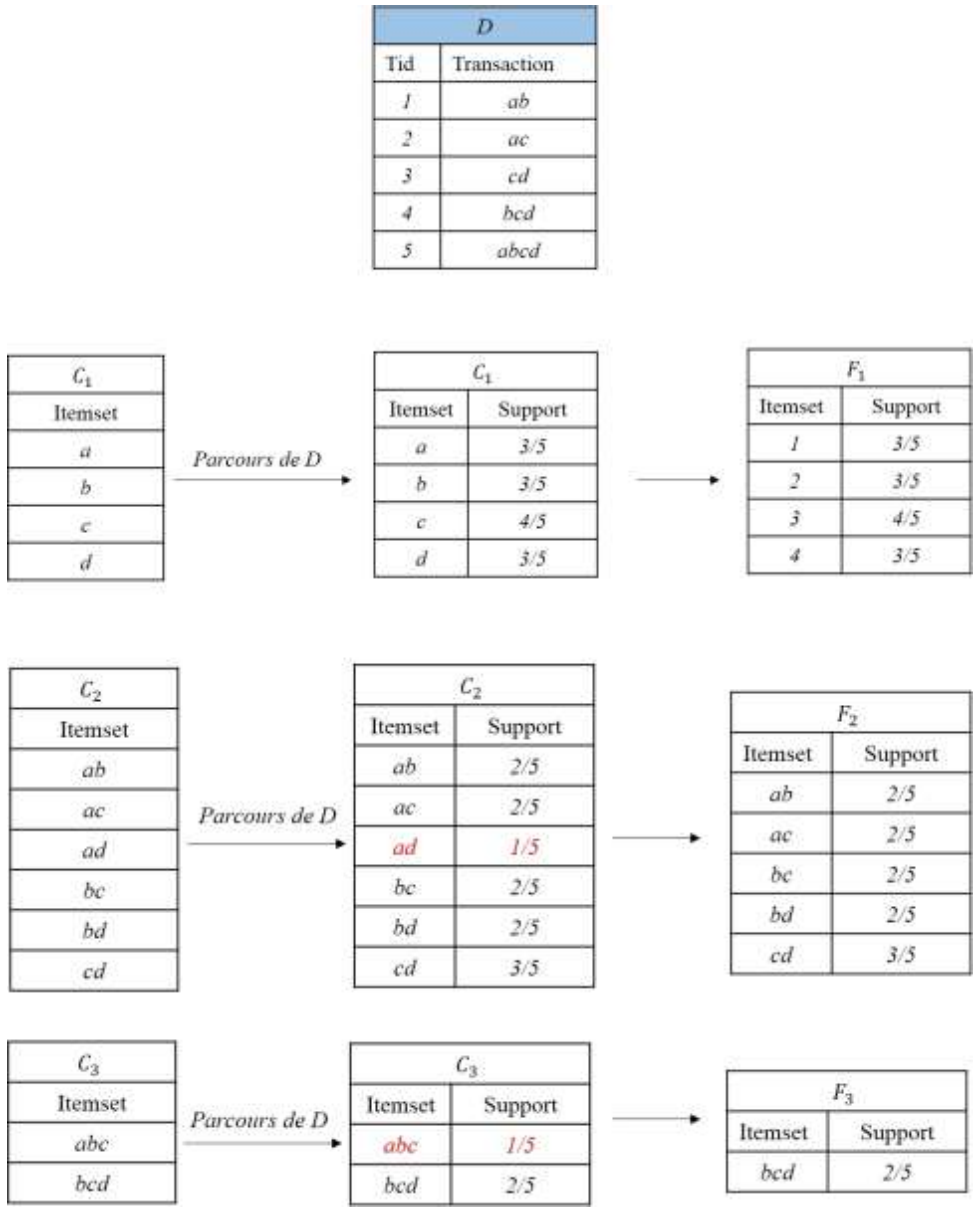

Figure 3.1. Processus apriori pour l'extraction des itemset fréquents

Structure d'un FP-tree :

Deux éléments essentiels constituent la structure d'un FP-tree qui sont:

a) Une structure sous forme d'un arbre avec une racine étiquetée 'null'.

b) Un index (une table des pointeurs des items fréquents).

L'arbre quant à lui est composé d'une racine 'null' et d'un ensemble de nœuds préfixé par l'élément représenté. Un nœud de l'arbre est composé par :

- Le nom de l'item (nom-item). Il s'agit de l'item que représente le nœud.

- Le nombre d'occurrence (count) de transaction où figure la portion de chemin jusqu'à ce nœud.
- Un lien vers le nœud suivant dans l'arbre (node-link). Il s'agit d'un lien inter-nœud vers les autres occurrences du même élément (ayant le même nom-item) figurant dans d'autres séquences de transactions. Cette valeur prend la valeur nulle s'il n'y a pas un tel nœud.
- L'index est une table d'en-tête qui contient la liste des items fréquents et qui pointe sur la première occurrence de chaque élément. Chaque entrée dans cette table contient :

a) Le nom de l'élément (nom-item).

b) Le pointeur tête de la séquence des nœuds ayant ce même nom-item.

L'avantage de cette représentation des données est qu'il suffit de suivre les liens inter-nœuds pour connaître toutes les associations fréquentes où figure l'élément fréquent. La construction du FP-tree passe par deux parcours de D et se fait de la manière suivante :

- Un premier parcours : On effectue un premier parcours pour déterminer les items fréquents pour un support minimum donné, ces itemsets fréquents seront triés par la suite par ordre décroissant de support dans une liste L.
- Un deuxième parcours : Dans le deuxième parcours de D chaque transaction est triée selon l'ordre des items dans L. Au début, le nœud racine (null) de l'arbre est crée, ainsi une branche sera créée pour chaque transaction, les transactions ayant un même préfixe partagent le même début d'une branche de l'arbre. Les items sont traités de plus fréquent au moins fréquent pour une bonne structure compacte à laquelle les items fréquents sont proches de la racine et sont mieux partagés par les transactions. Une fois l'arbre FP-tree est construit, on commence par des itemsets suffixes de taille 1, l'extraction se fait récursivement sur cette nouvelle sous-structure. Ces itemsets fréquents sont obtenus par concaténation du suffixe de la base conditionnelle avec les itemsets fréquents de sous arbre conditionnel.

Les différentes étapes de l'algorithme de FP-Growth est comme suit :

---

a. Balayer la base de transaction T une première fois
- Créer L, la liste des items frequents avec leur support
- Trier L en ordre décroissant du support

b. Créer l'arbre N contenant une racine "null"

c. Procédure FP-Growth (FP-Tree, null)

1. Si FP-Tree contient un seul chemin P alors
   - Pour chaque combinaison β de P faire
   - Générer l'itemset $\beta \cup \alpha$ de support=minimum des supports des nœuds de β

2. Sinon pour chaque a_i dans l'index de FP-Tree faire
   - Générer l'itemset $\beta = a_i \cup \alpha$ de support= a_i.support
   - Construire FP-Tree β

d. Si FP-Tree $\beta \neq \phi$ FP-Growth(FP-Tree β, β)

Les figures suivantes (Figure 3.2 et Figure 3.3) illustrent le passage de la construction de l'arbre FP-tree en prenant un exemple du tableau 3.1. Dans cette base nous avons cinq transactions et 33 motifs. Un motif est frequent si son support dépasse le seuil (minsupp). En fixant le minsupp à 50% les items frequents obtenus sont f, c, a, b, m et p (Tableau 3.2). Avec ces items on construit l'arbre FP en commençant par la première transaction (Figure 3.2).
L'arbre final est présenté à la Figure 3.3. Le tableau 3.4 représente le résultat par création de la base conditionnelle des items.

Tableau 3.1 : Base de données de transaction

TID	Items
1	f, a, c, d, g, i, m, p
2	a, b, c, f, l, m, o
3	b, f, h, j ,o
4	b, c, k, s, p
5	a, f, c, e, l, p, m, n

Tableau 3.2 : Items et support

item	Fréquence
f	4
a	3
c	4
b	3
m	3

Tableau 3.3 : Items ordonnés selon leur support

TID	Items	Items ordonnés selon leur fréquence
1	f, a, c, d, g, i, m, p	f, c, a, m, p
2	a, b, c, f, l, m, o	f, c, a, b, m
3	b, f, h, j, o	f, b
4	b, c, k, n, p	c, b, p
5	a, f, c, e, l, p, m, n	f, c, a, m, p

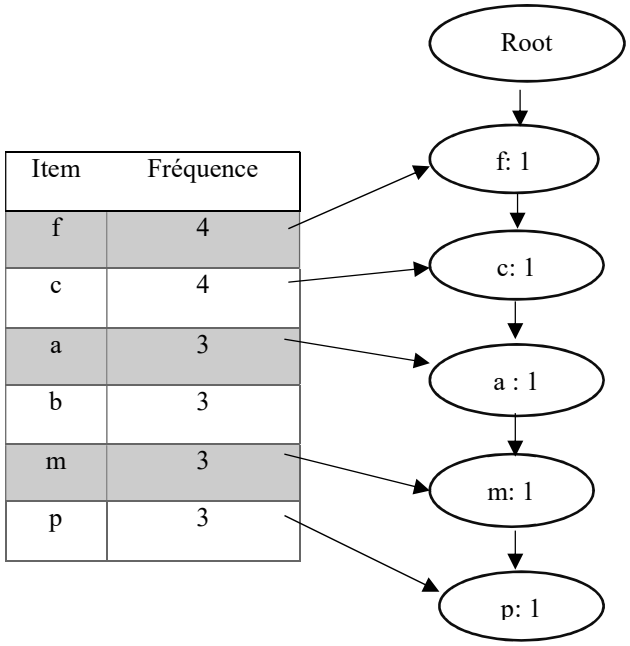

Figure 3.2 : Première étape de construction de l'arbre FP

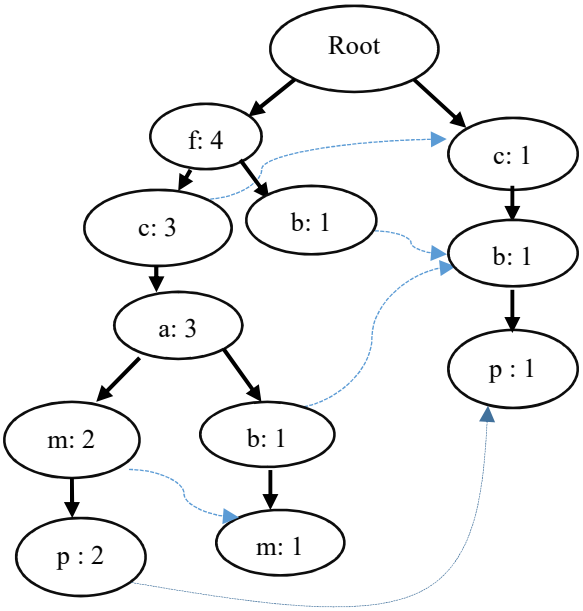

Figure 3.3 : l'arbre FP-Tree finale

Tableau 3.4 : Base conditionnelle des items

Item	Base conditionnelle	L'arbre FP conditionnel
p	{(fcam :2), (cb :1)}	{(c :3)}|p
m	{(fca :2), (fcab :1)}	{(f :3, c :3, a :3)}|m
b	{(fca :1), (f :1), (c :1)}	Vide
a	{(fc :3)}	{(f :3, c :3)}|a
c	{(f :3)}	{(f :3)}|c
f	vide	vide

4. Algorithmes en Spark

4.1. Algorithme YAFIM

YAFIM est un algorithme Apriori [41] parallèle basé sur le modèle Spark RDD. Il contient deux phases de traitement du flux de travail. L'algorithme 1 et l'algorithme 2 décrivent les codes des phases respectivement de l'algorithme YAFIM.

Algorithme 1 : Phase 1 **de** YAFIM

```
pour transaction t in Si
        flatMap (décalage de ligne, t)
                pour item I in t
                Out (I ,1)
                fin pour
         fin flatMap
fin pour
reduceByKey (I,count)
        som= 0
        tant que (Item I in partition)
                som+=count
        fin tant que
        si (som ≥ min sup)
                Out (I ,som)
        Fin si
Fin reduceByKey
```

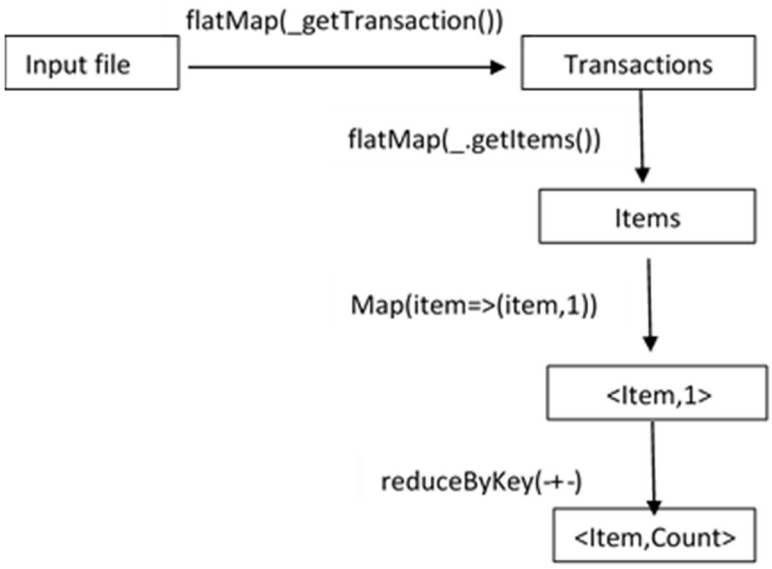

Figure 3.4. Lineage graph de YAFIM de la première étape

Tout d'abord, YAFIM est une implémentation Apriori sur Spark surpasse considérablement les implémentations d'Apriori. Initialement, les ensembles de données transactionnels de HDFS sont chargés dans Spark RDD (Resilient Distributed Datasets), qui

sont des objets de données basés sur la mémoire dans Spark, pour faire bon usage de la mémoire de cluster disponible dans 2 étapes. Dans la 1ère étape (Figure 3.4), il génère tous les éléments fréquents singleton et dans la 2ème étape (Figure 3.5), il utilise de manière itérative des ensembles d'éléments k-fréquents pour générer des ensembles d'éléments (k+1) -fréquents. Le processus est comme suit :

Algorithme 2 : Phase 2 de **YAFIM**

Read L_{k-1} from RDD
C_k=ap_gen(L_{K-1})
Pour transaction t en Si
    flatMap (décalage de ligne, t)
    C_t= sous-ensemble (C_K, t)
        Pour candidat c in C_t
            Out (c ,1)
        Fin pour
    fin flatMap
Fin pour
reduceByKey (c, count)
    som=0
    tant que (Item c in partition)
        som+=count
    fin tant que
    si (som ≥ min_sup)
        Out (c, som)
    Fin si
Fin reduceByKey

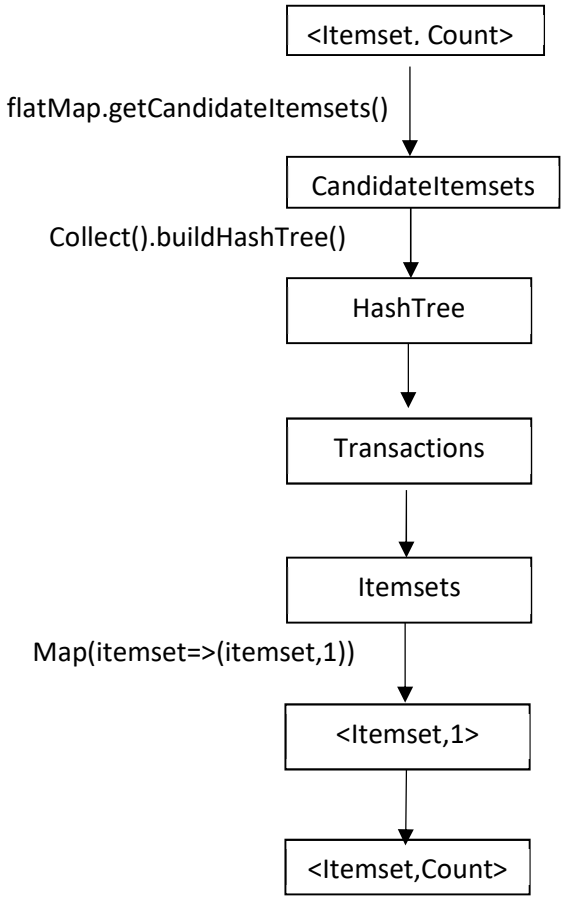

Figure 3.5. Lineage graph de YAFIM de la deuxième étape

Une fonction flatMap () (exécutée par les workers) sera appliquée au fichier d'entrée pour lire toutes les transactions et les mettre dans le RDD Transactions (Figure 3.4), (Algorithme 1). Ensuite, une fonction flatMap () sera appliquée sur chacune des transactions pour en extraire tous les éléments. Lorsque le processus flatMap est terminé, la fonction map() est appliquée pour transformer tous les éléments en paires clé / valeur <Item, 1> ;

- Enfin, la fonction reductionByKey () commence à compter la fréquence de chaque élément de l'ensemble de données transactionnel et élague également les éléments dont la fréquence est inférieure au support *minsup*. L'algorithme 1 décrit le pseudocode de la phase 1.
- Les ensembles d'éléments k-fréquents L_k sont lus et stockés en tant que RDD sous la forme de <itemset, count>. Ensuite, les ensembles d'éléments candidats C_{k+1} sont obtenus. Pour accélérer la vitesse de recherche des ensembles d'éléments (k + 1) fréquents à partir des ensembles d'éléments candidats, les C_{k+1} sont stockés dans un arbre de hachage**.**
- Analysons les transactions RDD existant pour lister toutes les occurrences de chacun des itemset candidats en appliquant une fonction flatMap (). De plus, une fonction map() sera appliqué pour émettre les paires clé / valeur <itemset, 1> pour chacun des itemset.

Enfin, la fonction reductionByKey () collecte tous les décomptes de support de chacun des ensembles d'éléments candidats et génère les paires <itemset, count> sous forme d'ensembles d'éléments (k + 1) -frequent lorsque le décompte n'est pas inférieur au nombre minimal de support minimal. L'algorithme 2 décrit le pseudocode de la phase 2.

4.2. Algorithme RApriori (An Efficient Apriori based Algorithm on Spark)

RApriori est un algorithme Apriori [42] parallèle basé sur le modèle Spark RDD. Il contient trois phases de traitement. R-APRIORI est une amélioration de YAFIM dans la troisième phase.

Phase 1 : Tout d'abord, les jeux de données transactionnels sous forme HDFS sont initialement chargés dans des RDD Spark. Le fichier d'entrée est diffusé sur chaque nœud de travail pour le rendre disponible localement à chaque Exécuteur.

Ensuite, une fonction flatMap () est appliquée au fichier d'entrée pour lire toutes les transactions et les mettre dans le RDD Transactions (Algorithme phase 1). Cette même function sera appelée à chaque transaction. Lorsque le processus flatMap est terminé, la fonction map () est appliquée pour transformer tous les éléments en paires clé / valeur <Item,1> . Enfin, la fonction reductionByKey () commence à compter la fréquence de chaque élément de l'ensemble de données transactionnel et élague également les éléments dont la fréquence est inférieure au support minsup. Les éléments restants sont stockés dans le jeu d'éléments frequents sous forme de singleton. Le graphique de lignage, montrant le flux de données et le contrôle à travers différentes étapes est

présenté à la (Figure 3.6).

Algorithme 3 Phase 1

```
Pour transaction t in Si
    flatMap (décalage de ligne, t)
        pour item I in t
            Out (I ,1)
        fin pour
    fin flatMap
fin pour
reduceByKey (I, count)
     som= 0
     tant que (Item I in partition)
       som+=compter
     fin tant que
     si (som ≥ min_sup)
        Out (I, som)
     Fin si
fin reduceByKey
```

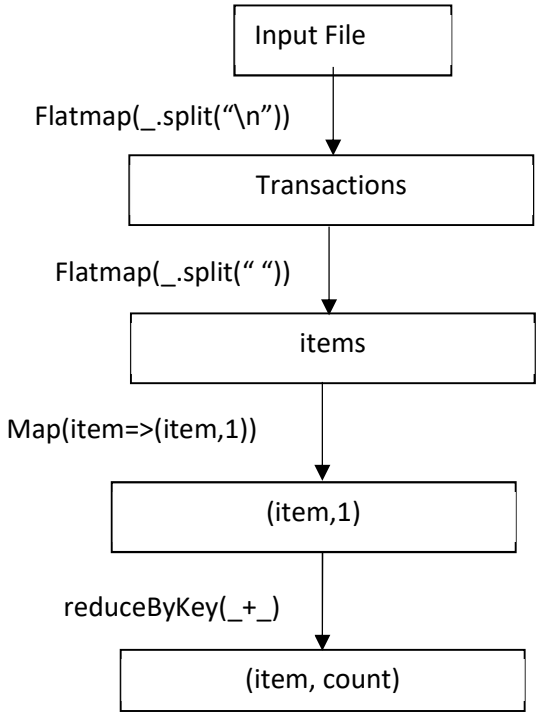

Figure 3.6. Lineage graph de RDDs pour Rapriori de la phase 1

Phase 2 :

Dans R-Apriori, les éléments fréquents singleton sont stockés dans un filtre bloom parce que les filtres de floraison sont plus rapides que les arbres de hachage et peuvent facilement stocker des items de longueur 1. La fonction **map()** élague chaque transaction de sorte qu'elle ne contienne que des tems qui existent dans le filter Cuckoo et génère toutes les paires possibles pour les éléments de la transaction élaguée. **Reducer** pend chaque pair comme clé et calcule le nombre total pour chaque pair et génère toutes les paires dont la fréquence est supérieure au support minimum. L'algorithme 4 représente le pseudocode de la phase 2. Le graphique de lignage est illustré à la Figure 3.7.

Algorithme 4 Phase 2 de RApriori

Si le nombre d'items fréquents en un seul singleton est important
  pour transaction t in Si
    F_T=intersect (T, L_{K-1})
    C_T=pair(F_T)
    Candidat c in C_t
    Out (c ,1)
  Fin pour
  reduceByKey (I, count)
    som=0
    t**ant que** (Item I in partition)
      som+=count
    fin tant que
      si (som ≥ min_sup)
        Out (c, som)
      Fin si
  Fin reduceByKey
fin si

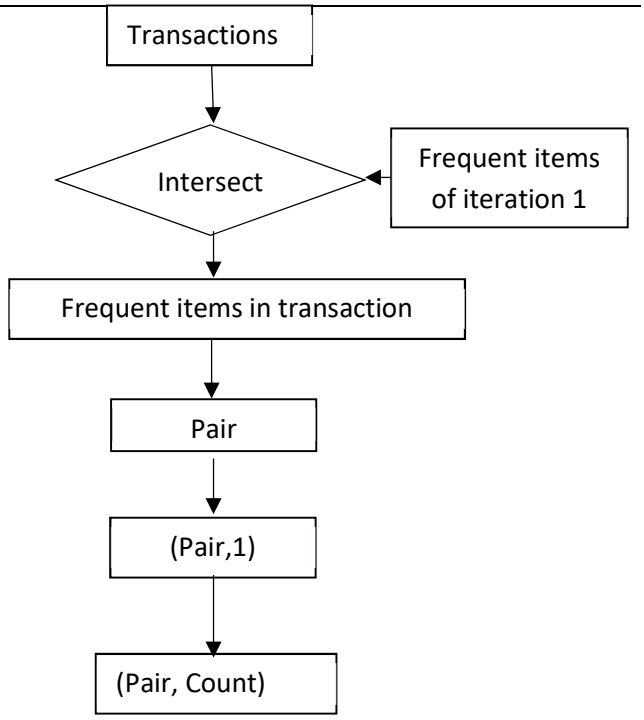

Figure 3.7 : Lineage graph de RDDs pour Rapriori dans la phase 2

Phase 3 :

Des itemset K-frequent sont utilisés pour générer des itemsets (K+1)-fréquents. Initialement, les ensembles d'éléments K-fréquent Lk1 sont insérés dans les RDD et l'ensemble candidat C_{k+1}, est généré et stocké dans un filtre pour augmenter la vitesse de recherche de l'ensemble d'éléments K+1.

- Ensuite une fonction flatMap est utilisée pour obtenir chaque transaction et vérifier les ensembles (k+1) - fréquents possibles.
- La fonction **map ()** quant à elle, permet de générer un (Item, I) pour chaque Item.

Finalement, la fonction **reduceByKey ()** calcule la fréquence de chaque élément et élague les éléments dont la fréquence est inférieure au support minimum. Les éléments restants sont stockés dans le jeu d'éléments K-fréquent. L'algorithme 5 représente le pseudocode de la phase 3. Le graphique de lignage (Figure 3.8) montre le flux de données et le contrôle à travers diffrentes étapes de la phase 3.

Algorithme 5 Phase 3 de RApriori

Read L_{k-1} from RDD
C_k=ap_gen(L_{K-1})
Pour transaction t en Si
  flatMap (décalage de ligne, t)
C_t= sous-ensemble (C_K, t)
Condidat c in C_t
Out (c ,1)
fin flatMap
Fin pour
reduceByKey (I, count)
Som=0
  tant que (Item I in partition)
    Som+=count
fin tant que
  si (som ≥ min_sup)
    Out (I, som)
  Fin si
Fin reduceByKey

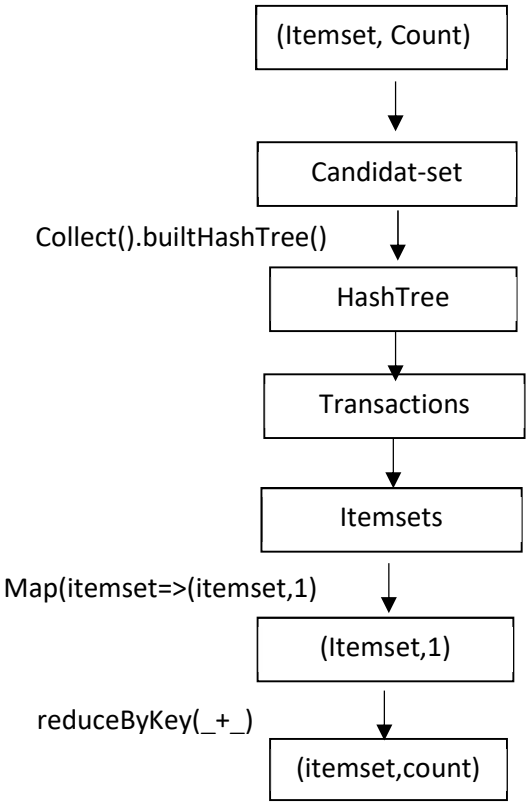

Figure 3.8. Lineage graph de RDDs pour Rapriori dans la troisième étape

4.3. Algorithme DFPS (Distributed FP-growth Algorithm Based on Spark)

Étant donné une base de données de transactions et le minimum seuil de support ξ, DFPS utilise trois étapes pour extraire les itemsets fréquents pour les données massives. DFPS est un algorithme FP-Growth [43, 44, 45] parallèle basé sur le modèle Spark RDD. DFPS charge l'ensemble de données de transaction de HDFS dans Spark en tant que RDD et compte le support de chaque élément.

Etape 1: Calcul de l'ensemble d'éléments fréquents

Dans cette étape, DFPS charge l'ensemble de données de transaction à partir de HDFS dans Spark en tant que RDD et compte le support de chaque élément. Ensuite, les éléments non fréquents sont supprimés et l'ensemble d'éléments fréquents singleton est obtenu. L'algorithme de l'étape 1 est présenté par l'algorithme 6.

Algorithme 6 : étape 1 de DFPS

```
Input : Transaction database DB , minimum support ξ.
Output : Frequent 1-itemset
Method : Call getSingleton (DB, ξ).
Procédure: getSingleton (DB, ξ)
{
        Foreach transaction T in DB do
                flatMap (line offset, T)
                        foreach item I in T do
                                Out(<I,1>) ;
        reduceByKey(I, count)
                Sum=0 ;
                while (item i in partition)
                        sum+=count ;
                Out (<I, sum>) ;
        Filter (I, count)
                If count>= ξ then
                        Out (<I, count>) ;
}
```

Étape 2 : Répartition de la Base conditionnelle

C'est l'étape cruciale de l'algorithme DFPS. Supposons qu'il y ait "l" ensemble d'éléments fréquents singleton à l'étape 1. L'algorithme DFPS mappe la base de modèle conditionnel (CPB) de l'ensemble fréquent d'éléments singleton en (I-1) partitions. Soit les l itemsets fréquents a_1, a_2, a_3, ..., a_l, triés par ordre décroissant. Il y aura donc (l-1) partitions $P(a_2)$, $P(a_3)$, …, $P(a_l)$, pour eux et l la base de modèle conditionnelle de a_k est mappée en $P(a_k)$. Il n'y a pas de partition $P(a_k)$ pour a_1. Parce que les modèles fréquents se terminent par a_1 sont a_1 lui-même. Ainsi, la base de modèle conditionnelle d'a_1 est nulle. Pour chaque transaction élaguée T= {a_1, a_2, a_3, …, a_n} dans DB après supprimer les éléments peu fréquents et trier T en décroissant, en commençant à partir de l'élément de queue a_n , map {a_1, a_2, a_3,…a_{n-1}}, la base conditionnel de motif de a_n , pour partitionner $P(a_n)$. Et supprimer a_n de la transaction T.

Ensuite, mappez la base de modèle conditionnel de a_{n-1} pour partitionner $P(a_{n-1})$ de la même manière que a_n. Après cette phase 2, tous les bases conditionnelle de a_2, a_3 , , ,...,a_l sont mappées en $P(a_2)$, $P(a_3)$, …,$P(a_l)$ respectivement. Le pseudo code de l'étape 2 de la répartition de la base conditionnelle est décrit ci-dessus (Algorithme 7). Le processus de l'étape 2 est illustré à la figure 3.9.

Algorithme 7

```
Input : Transaction RDD tRDD after step1, frequent 1-itemset F.
Output : Conditionnel pattern base of each item in F,
Method : Call getCPB( tRDD, F).
Procédure : getCPB( tRDD, F){
```

```
Foreach transaction T= {a1, a2, a3, …, an}
        In tRDD do
                flatMap(line offset, T)
                        foreach item ak in T except a1 do
                                Out{< ak, (a1, a2, a3, …ak-1)>} ;
        groupByKey(repartition()) ;
}
```

Etape 3 : Fouille des motifs Fréquents en parallèle

Les bases conditionnelles de a_1, a_2, a_3, ..., a_l sont listées en partitions $P(a_1)$, $P(a_2)$, $P(a_3)$, …, $P(a_l)$ respectivement après l'étape 2. DFPS construit l'arbre FP-Tree dans chaque partition et extrait les motifs fréquents dans chaque partition indépendamment. Le pseudo code est décrit ci-dessus (Algorithme 8).

5. Description de la base

La base de données est construite suivant une enquête qui recueille les réponses de plus de 400 000 Américains sur les comportements à risque liés à la santé, les conditions de santé chroniques et l'utilisation des services préventifs. Elle a été menée chaque année depuis 1984. Pour ce projet, un csv du jeu de données disponible sur Kaggle pour l'année 2015 a été utilisé.

Algorithme 8

Input: Pattern fragment α, the conditionnel pattern base *CPB* of α and a minimum support threshold ξ.
Output: Frequent itemset.
Method: Call DFPS (*CPB*,α,ξ).
Procédure: DFPS (*CPB*,α,ξ)

```
{
        build the conditional FP-tree Tree of α;
        if Tree contains a single path P
        then foreach combination (denoted as β) of the nodes in P do
                generate pattern φ = β  with γ.support =minimum support of nodes in β;
                if φ.support ≥ ξ then
                   Out ( <φ,φ.support > );
                   else foreach ai in the header of Tree do {
        generate pattern β =ai  with
                support = ai.support;
        construct β's conditional pattern base and then β's conditional FP-tree Treeβ ;
        if Treeβ ≠∅ then
                call DFPS(Treeβ , β ,ξ); }
        }
```

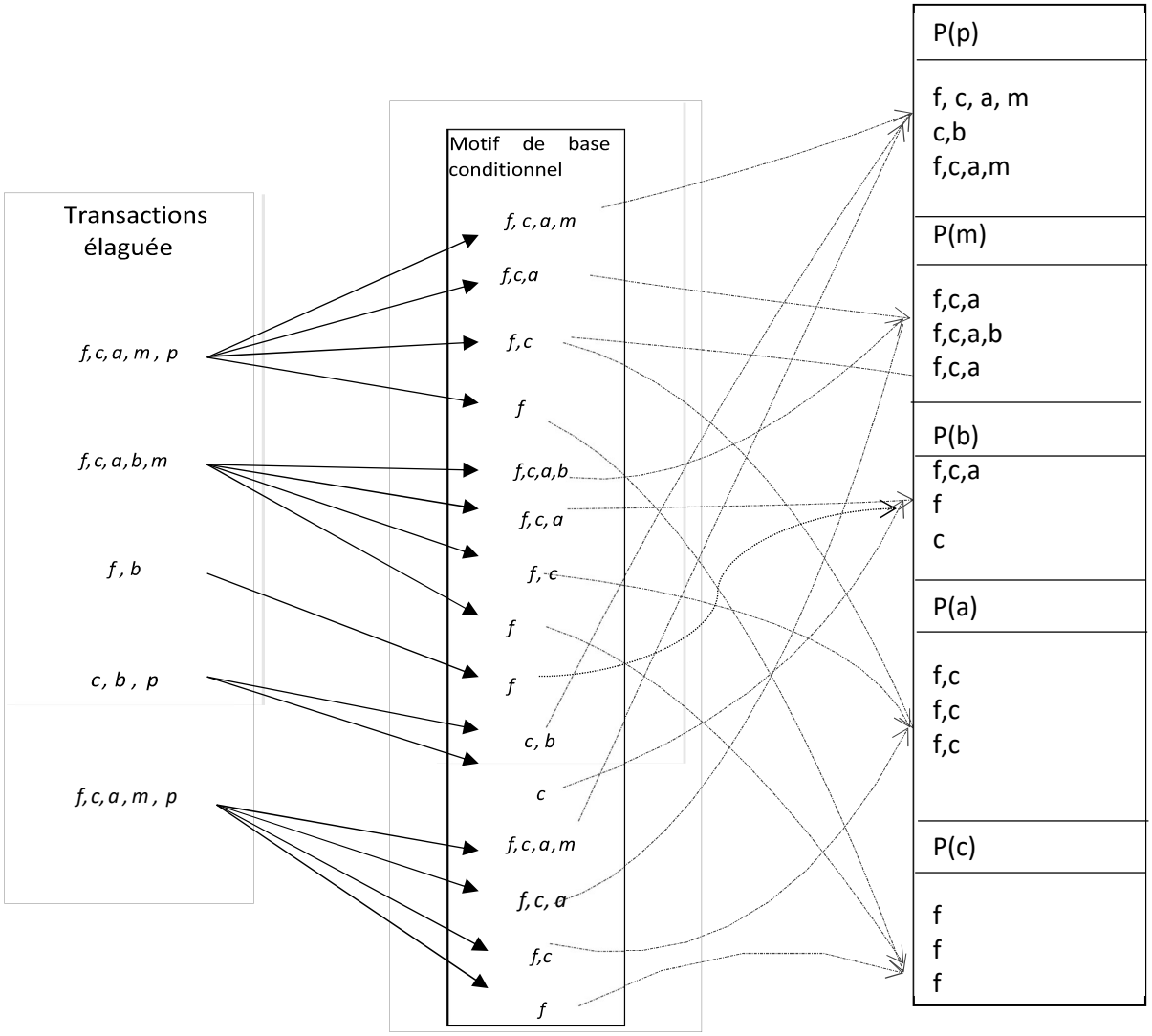

Figure 3.9 : Processus de l'étape 2 concernant la figure 3.3

Ce jeu de données original contient les réponses de 441455 personnes et comporte 330 caractéristiques. Ces caractéristiques sont soit des questions directement posées aux participants, soit des variables calculées à partir des réponses individuelles des participants. Cette base de données contient trois fichiers, celui utilisé dans ce projet de recherche contient 253680 réponses. La variable cible est celle de Diabètes012 qui représente trois classes "Diabètes", "Pré-diabètes" et "Non Diabètes". En gros, cette dataset contient 21 variables qui peuvent avoir une corrélation ou une causalité avec le diabète.

Les variables de cette base de données décrivent les facteurs de risques de l'existence de

diabète, ces derniers sont comme suit :

a. Diabètes012: cette variable décrit les classes existantes: ”Diabètes », « Pré-diabètes” et ”Non Diabètes”.
b. High Blood Pressure : décrit si la personne en question a été déjà informée qu’il a une Hypertension artérielle par un médecin.
c. High Cholesterol : décrit si la personne en question a été déjà informée qu’il a le Cholestérol élevé par un médecin.
d. BMI : Décrit l’indice de masse de chaque personne en question.
e. Smoking : Décrit le tabagisme plus spécifiquement si la personne en question à consommer plus de 100 cigarettes au long de sa vie.
f. Physical Activity : Si la personne en question est active physiquement dans les derniers 30 jours.
g. Alcohol Consumption : décrit si la personne en question à un excès de consommation d’alcool.
h. Health Care : Décrit si la personne en question à une couverture médicale (Assurances ou bien des organismes en charge) et s’il avait besoin d’un médecin et n’a pas pu le voir à cause de charge financière.
i. Education : Décrit le niveau d’étude.
j. Mental Health : Décrit si le patient en question à fait une visite à un psychiatre dans les 30 derniers jours.
k. Income : Décrit le revenu mensuel de la personne, cette variable est importante car il peut donner une approximation à la qualité de vie (Niveau de Stress) de la personne en question.
l. Healthcare disease or attak : Décrit si le patient souffre des malaises cardiaques (Crise, autre maladie cardiaque).

6. Prétraitement des données

6.1. Transformation des données

En premier lieu, les valeurs des données étaient des valeurs numériques qui substituent des valeurs catégoriques. La première transformation est de rendre tous les valeurs en texte pour les rendre comme des items d’une base transactionnelle. Pour le traitement de la dataset en tant qu’une base de données transactionnelle, cette étape est obligatoire afin d’avoir des éléments

uniques dans chacune des transactions. Pour mieux éclaircir, la figure 10.3 décrit les premières transactions avant et après la transformation :

- Cette transformation est obligatoire pour qu'on peut avoir des items uniques dans une
- Transaction (Une transaction étant une ligne des enregistrements).
- On ne peut pas connaître lequel des items existant ou non existant si on a laissé les valeurs en tant que 0 et 1, car on peut voir plusieurs fois des valeurs de 1, mais au niveau de l'interpréter à quoi ils reviennent, ça serait impossible.
- Cette transformation est une adaptation des données au fonctionnement des algorithmes d'extraction des règles d'associations.

La deuxième transformation est de découper les variables continues en intervalles dans le but de les rendre discrètes. A titre d'exemple dans la dataset, on avait les deux variables "**BMI**" et "**Age**" (Tableau 3.5 et Tableau 3.6).

Diabetes_012	HighBP	HighChol	CholCheck	BMI	Smoker	Stroke
0	1	1	1	40	1	0
0	0	0	0	25	1	0
0	1	1	1	28	0	0
0	1	0	1	27	0	0
0	1	1	1	24	0	0
0	1	1	1	25	1	0
0	1	0	1	30	1	0
0	1	1	1	25	1	0
2	1	1	1	30	1	0

Diabetes_012	HighBP	HighChol	CholCheck	BMI	Smoker	Stroke
No Diabetes	High BP	High Cholesterol	Cholesterol Check in 5 Years	Obeisity class III	Yes Smoking	No Stroke
No Diabetes	No High	No High Cholesterol	No Cholesterol Check in 5 Years	OverWeight	Yes Smoking	No Stroke
No Diabetes	High BP	High Cholesterol	Cholesterol Check in 5 Years	OverWeight	No Smoking	No Stroke
No Diabetes	High BP	No High Cholesterol	Cholesterol Check in 5 Years	OverWeight	No Smoking	No Stroke
No Diabetes	High BP	High Cholesterol	Cholesterol Check in 5 Years	Normal weight	No Smoking	No Stroke
No Diabetes	High BP	High Cholesterol	Cholesterol Check in 5 Years	OverWeight	Yes Smoking	No Stroke
No Diabetes	High BP	No High Cholesterol	Cholesterol Check in 5 Years	Obeisity class I	Yes Smoking	No Stroke
No Diabetes	High BP	High Cholesterol	Cholesterol Check in 5 Years	OverWeight	Yes Smoking	No Stroke
Diabetes	High BP	High Cholesterol	Cholesterol Check in 5 Years	Obeisity class I	Yes Smoking	No Stroke

(a) Items dans une transaction doivent être unique

HighChol	CholCheck
1	1
0	0
1	1
0	1
1	1
1	1
0	1
1	1
1	1

→

HighChol	CholCheck
High Cholesterol	Cholesterol Check in 5 Years
No High Cholesterol	No Cholesterol Check in 5 Years
High Cholesterol	Cholesterol Check in 5 Years
No High Cholesterol	Cholesterol Check in 5 Years
High Cholesterol	Cholesterol Check in 5 Years
High Cholesterol	Cholesterol Check in 5 Years
No High Cholesterol	Cholesterol Check in 5 Years
High Cholesterol	Cholesterol Check in 5 Years
High Cholesterol	Cholesterol Check in 5 Years

(b) Résultat de la transformation

Figure 3.10 : Première transformation de valeurs numérique au texte

— Le choix des intervalles pour la variable BMI qui réfère au indice de masse est basé sur

— L'interprétation des valeurs d'indices de masse dans le domaine de la santé.

Les valeurs de **BMI** avant la transformation sont des variables continues, le tableau ci-dessous décrit les différents intervalles existants après la transformation.

Le choix des intervalles pour la variable **Age** revient souvent à la collection de la base de données, lors des enregistrements. Ce genre de transformations a été appliqué sur la globalité du dataset, de sorte qu'on aura des données des variables catégoriques.

Tableau 3.5 : Valeurs de la variables BMI et répartition des intervalles

Intervalles des valeurs BMI	interprétation
[0,18.4[	Under Weigh
[18.5, 24.9]	Normal weight
[25, 29.9]	Over weight
[30, 34.9]	Obeisity classe I
[35, 39.9]	Obeisity classe II
40>	Obeisity classe III

Tableau 3.6 : Valeurs de la variable Age et répartition des intervalles

Code Age	Intervalle de refrences	Code Age	Intervalle de refrences
1	[18-24]	7	[50-54]
2	[25-29]	8	[55-59]
3	[30-34]	9	[60-64]
4	[35-39]	10	[65-69]
5	[40-44]	11	[70-74]
6	[45-49]	12	[75-79]

6.2. Visualisations des données

La visualisation des données est définie comme l'exploration visuelle et interactive de données. C'est un moyen rapide d'obtenir des informations à travers l'exploration visuelle, des rapports fiables et un partage d'informations aisées.

Dans cette partie on s'est basé sur des visualisations des données après leur transformation pour mieux découvrir la répartition des données selon les classes existantes, l'effectif de chaque caractéristique par rapport à un autre.

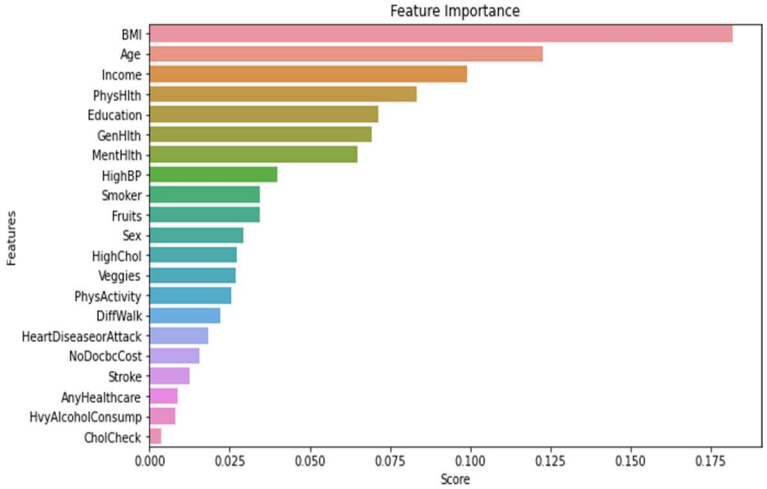

Figure 3.11. Mesures d'importance des variables

La figure 3.11 décrit l'importance des variables dans nos données, en les modélisant pour effectuer une prédiction, en effet l'importance mesurée est celle de l'effet de chaque variable pour faire la prédiction. En fait, on peut se baser sur les résultats obtenu pour bien déterminer les règles d'associations. La Figure 3.12 montre la répartition des variables pression artérielle et taux de cholestérol.

- Les personnes souffrant d'hypertension artérielle sont plus susceptibles de souffrir de diabète.
- Les personnes ayant un taux de cholestérol élevé sont plus susceptibles de souffrir de diabète.

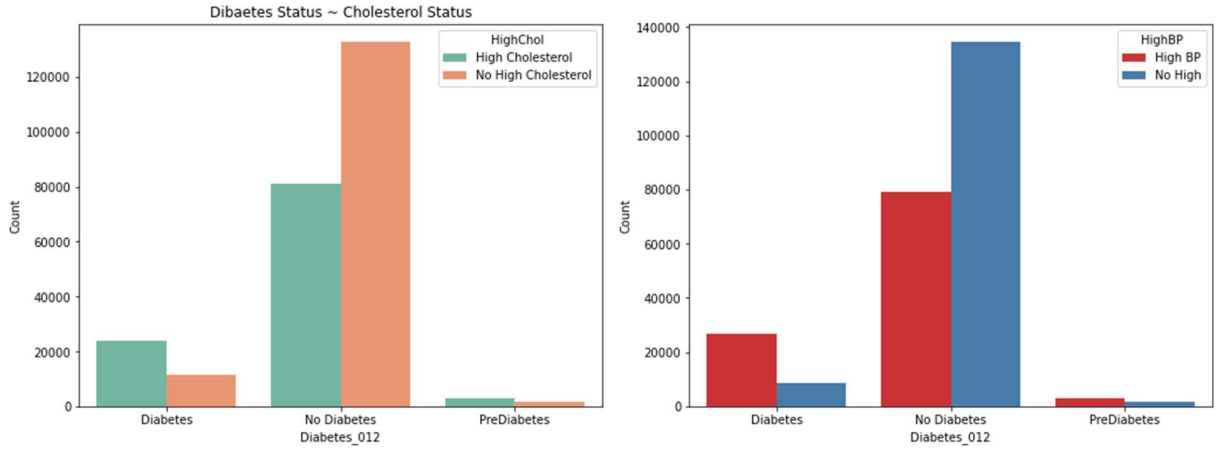

Figure 3.12 : Répartition des variables HighCol et HighBP

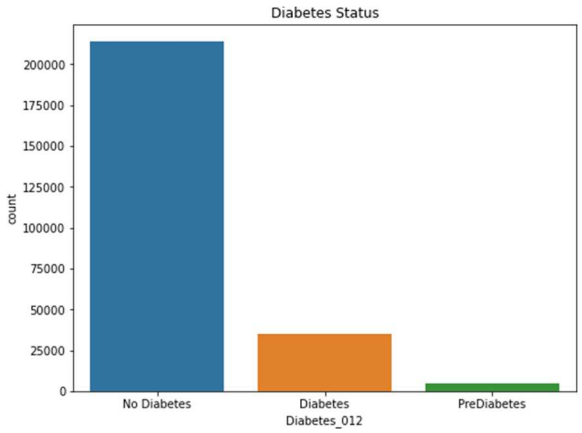

Figure 3.13 : Répartion des classes : diabètes, prédiabètes, non diabètes

L'attribut "Diabetes012" (Figure 3.13) montre qu'il y a 213703 personnes sur 253680 qui sont en bonne santé, 35346 sont diabétiques et le reste des 4631 sont en phase pré-diabétique. Cela signifie que 84,24 % des personnes sont en bonne santé. Le pourcentage de patients en phase pré-diabétique est 1,83 % et 13,93 % des personnes sont diabétiques.

7. Méthodologie d'extraction des règles d'association

Afin d'extraire les règles d'associations entre les patients diabétiques, on a procédé par trois méthodes (Figure 3.14), la première consiste à extraire les itemsets fréquents et les règles d'association en utilisant DFPS (Distributed frequent pattern Spark) sans regroupement de données, la deuxième consiste à extraire les itemsets fréquents en appliquant DFPS sur les groupes de données qui résulte de regroupement effectué par K-modes.

Ensuite, la troisième méthode s'agit d'extraire les règles d'associations en utilisant les deux implémentions distribuées d'Apriori dans spark, ces derniers sont YAFIM et RApriori.

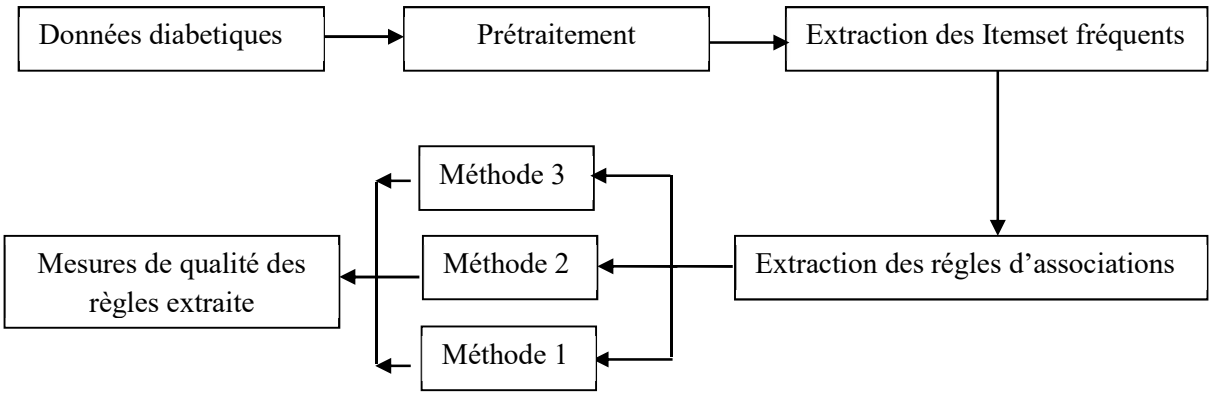

Figure 3.14 : Méthodologie d'extraction des règles d'associations

8. Résultats expérimentaux

8.1. Clustering avec K-modes

Afin d'effectuer un clustering, la première étape est de déterminer le nombre optimal de groupe homogène k. Pour se faire, on a utilisé la méthode de coude. La méthode de coude consiste à itérer les valeurs de k de 1 à 5 et calculer les valeurs des distorsions et l'inertie pour chaque valeur de k dans la plage donnée. Pour déterminer le nombre optimal de clusters, il faut sélectionner la valeur de k au « coude », c'est-à-dire le point après lequel l'inertie commence à diminuer de façon linéaire. Ainsi, pour les données, nous concluons que le nombre optimal de groupes pour les données est de 3 (Figure 3.15).

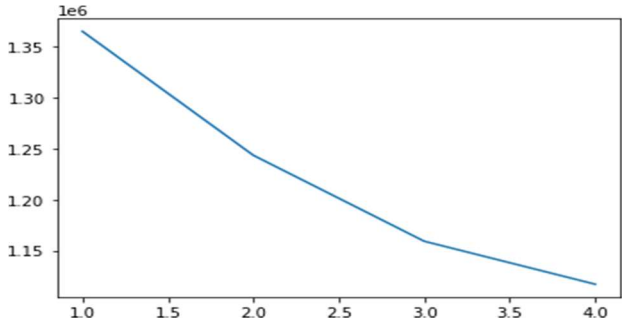

Figure 3.15 : Méthode de coude pour déterminer le nombre K

La figure 3.15 décrit le résultat de la méthode de coude qui démontre que le nombre optimal k est k = 3.

La méthode des k-modes est similaire à celle des k-moyennes mais adaptée pour les objets catégoriques, ce sont des objets qui ne contiennent pas de valeurs numériques mais des chaînes par exemple. Le principe de la méthode des k-modes se déroule ainsi :

- On initialise k modes (aléatoirement par exemple).
- On associe chaque objet de l'ensemble de données au mode le plus proche ou le plus similaire à lui.
- Puis on recalcule les modes de chaque ensemble, à partir de la fréquence des champs des objets.
- Et on recommence les étapes 2 et 3, jusqu'à ce que les modes ne changent plus.

La fonction de distance utilisée, retourne une valeur en fonction du nombre de champs identiques (La distance de Hamming). Pour trouver un mode, à partir d'un cluster, idéalement on essaie de construire l'objet qui minimise la somme des distances par rapport à tous les objets du cluster. Et de manière plus simple, on associe à chaque champ du mode la valeur du champ la plus fréquente parmi les éléments de son cluster.

8.2. Caractéristiques des groupes extraits

D'après la figure 3.16 on peut affirmer que :

- Le premier groupe englobe la majorité avec le sexe féminin et un grand âge et avec un indice de masse qui décrit leurs surpoids.
- Le deuxième groupe englobe la majorité avec le sexe masculin et la plupart de ses membres ont un indice de masse obésité de classe 2.

Le troisième groupe contient une majorité des femmes avec un indice de masse et âge répartis.

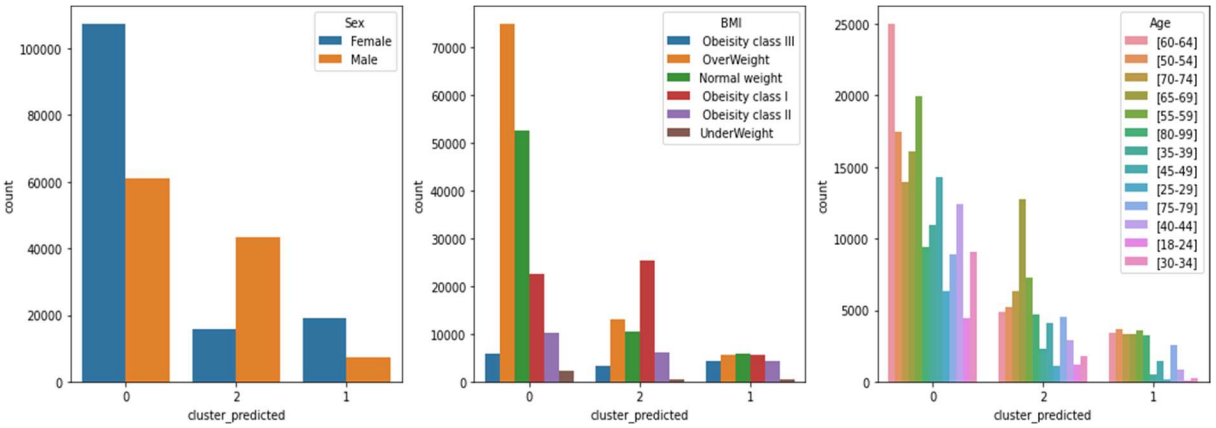

Figure 3.16 : Exploration des groupes séparés

8.3. Extraction des régles d'associations avec DFPS

Cette méthode consiste à extraire les itemsets fréquent et les règles d'associations sur les données après leur préparation sans avoir les classer ou bien cibler les données les plus descriptifs. Avant d'appliquer cette méthode, il est prévu qu'on aura un grand nombre d'items fréquents et de règles d'associations, ce qui est un inconvénient car on peut se retrouver avec plusieurs règles aberrantes.

L'extraction des règles d'associations sur la globalité de la dataset nécessite un temps d'exécution plus élevé surtout avec un support minimal inférieur (Figure 3.17). Ainsi que le nombre d'item sets fréquents est exponentielle en fonction des valeurs 0.02, 0.04, 0.08, 0.2, 0.5 et 0.7 de minsupport respectivement (Tableau 3.7), (Figure 3.18). En utilisant une valeur de support minimal assez petite, le nombre des règles devient exponentiel et difficile à interpréter dû à la redondance de ces derniers.

Ayant une grande base de données, la complexité de calcul reste encore grande en appliquant l'algorithme FP-Growth distribuées sur Spark. Pour cela, dans la section suivante on exploitera les résultats de clustering pour améliorer le temps de réponse et la complexité de calcul.

Tableau 3.7: Temps d'exécution et nombre d'itemsets fréquent

Minsupp	Nombre d'itemsets fréquents	Temps d'exécution
0.02	2045269	852.74
0.04	932022	407.38
0.08	357734	251.26
0.2	58217	162.51
0.5	2869	130.09
0.7	307	52.15

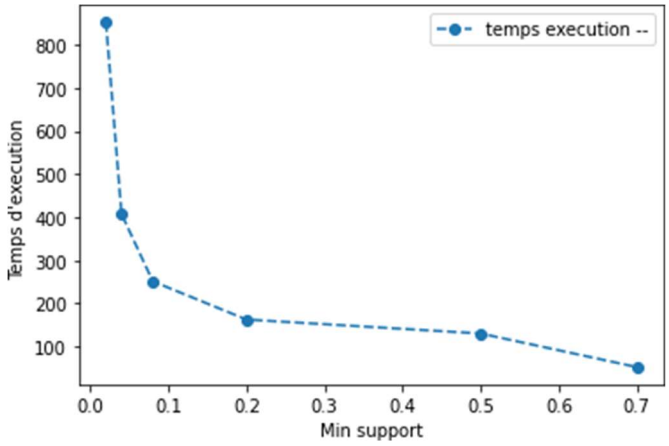

Figure 3.17 : Temps d'exécution vs Minsupp

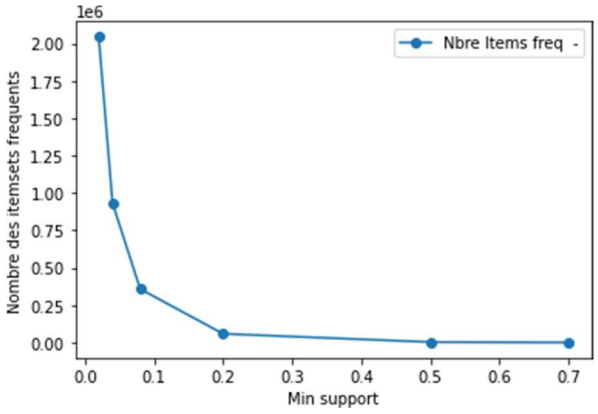

Figure 3.18 : Nombre des itemsets fréquents vs Minsupp

8.4. Extraction des règles d'associations en utilisant le clustering puis DFPS

Après avoir eu les résultats de clustering qui ont été présenté sur la partie précédente de ce chapitre des visualisations des groupes ont été envisagé afin de choisir le regroupement des patients diabétiques et ceux qui ont des facteurs de risques de diabètes.

En se basant sur les visualisations de chacun de ses groupes, le groupe 2 (cluster 2) été le plus adapté afin d'effectuer l'extraction des règles d'association entre les patients de diabètes et ceux qui ont des facteurs de risques. Le fichier de données de cette méthode contient un nombre d'enregistrements inférieur à la globalité de la dataset. En général, le nombre des itemsets fréquents et le nombre de règles extraites très similaires dans la plupart des cas. Les figures ci-dessus (Figure 3.19 et Figure 3.20) décrivent la variance de nombre des itemsets fréquents en fonction des valeurs de minsupport. Il est clair que le nombre des itemsets fréquents a démunie par rapport à la première méthode. Ceci rend les règles d'associations moins redondantes et interprétables.

Tableau 3.8 : Temps d'exécution vs nombre d'itemsets fréquents

Minsupp	Nombre d'itemsets fréquents	Temps d'exécution
0.02	337035	215.79
0.04	119093	78.99
0.08	38743	42.53
0.2	4964	31.51
0.5	369	18.88
0.7	53	16.49

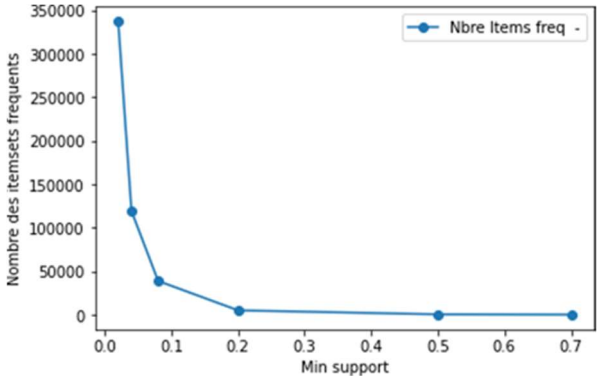

Figure 3.19 : Nombre des itemsets fréquent vs Minsupp

Pour le temps d'exécution, on le trouve décroissant lors de la croissance de minsupport, en général la complexité de calcul diminue par rapport à la première méthode.

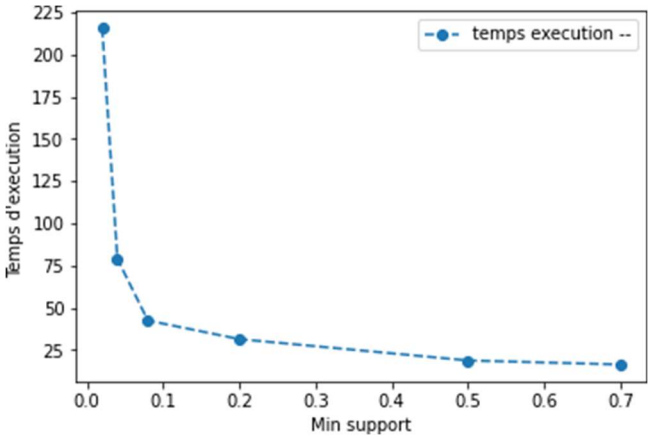

Figure 3.20 : Temps d'exécution vs Minsupp

8.5. Extraction des règles d'associations avec en utilisant le clustering puis YAFIM et R-Apriori

Pour mener la comparaison des deux algorithmes : Yafim et R-Apriori, on a appliqué ces algorithmes sur le fichier résultant du regroupement qui a été exploité dans la section précédente. La figure 3.21 présente le temps d'exécution de YAFIM et R-Apriori.

Par rapport à DFPS ces deux algorithmes présentent un temps d'exécution très élevé avec tout diminution de Min-support.

D'autre part R-apriori présente de meilleurs résultats par rapport à YAFIM à cause de filtre qu'il applique sur les itemsets fréquents avant la génération des règles d'associations.

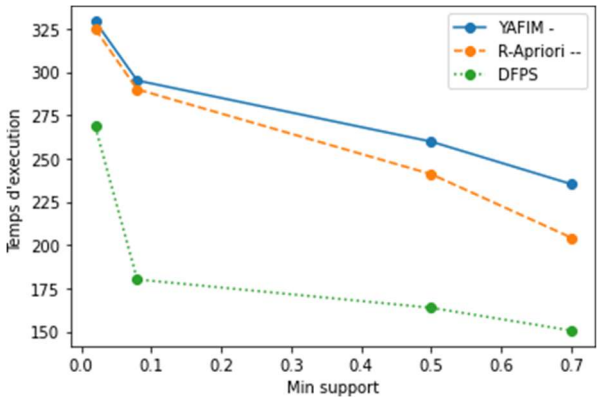

Figure 3.21 : Comparaison du temps d'exécution des trois algorithmes

9. Règles d'associations extraites entre les patients diabétiques

De nombreuses règles d'associations ont été extraites. La figure 3.22 affiche quelques-unes avec les mesures de confiance et le support. D'après la figure 3.22 on peut remarquer que

les antécédent qui contient un taux de cholestérol élevé avec la tension artérielle élevé et la présence d'un sur-poids ont comme conséquence la présence du diabètes, du coup on peut conclure que ces trois facteurs peuvent être des facteurs de risques du diabètes.

10. Visualisation des règles d'associations

La visualisation est un ensemble de méthodes de représentation graphique, en deux ou trois dimensions, elles permettent aux décideurs de voir les analyses présentées visuellement afin qu'ils puissent identifier de nouveaux modèles d'une manière plus simple. Pour visualiser les règles d'association extraites, nous avons utilisé arulesViz. Ce paquetage propose plusieurs techniques de visualisation connues et nouvelles telles que la visualisation matricielle, graphique, matrice groupée, etc. La figure 3.23 montre la dispersion des règles d'association en utilisant le support et la confiance sur les axes.

Antécedent	Conséquent	Confidence	Support
['High Cholesterol', 'Poor PhysHlth', 'High BP', 'Poor MentHlth']	['[Diabetes']	1	0.31695241
['No NoDocbcCost']	['[No Diabetes']	0.79011243	0.71916509
['No Fruits ', 'High Cholesterol', 'High BP', 'No HvyAlcoholConsump']	['[Diabetes']	1	0.22398574
[' OverWeight', 'High BP', 'Yes AnyHealthcare', 'No HvyAlcoholConsump', 'Cholesterol Check in 5 Years']	['[Diabetes']	1	0.20307814
['Yes Smoker']	['Poor PhysHlth']	0.8311951	0.61428352
['No Fruits ', 'High Cholesterol', 'High BP', 'No HvyAlcoholConsump']	['[Diabetes']	1	0.22398574
['High BP', 'No heart disease or attak', 'No NoDocbcCost', 'Cholesterol Check in 5 Years']	['[Diabetes']	1	0.50370622
['High BP', 'No heart disease or attak', 'No NoDocbcCost', 'Cholesterol Check in 5 Years']	['[Diabetes']	1	0.50370622
['Yes DiffWalk]', 'High Cholesterol']	['[Diabetes']	1	0.26727211
[' OverWeight', 'No HvyAlcoholConsump', 'Cholesterol Check in 5 Years']	['[Diabetes']	1	0.29273468

Figure 3.22 : Prise d'écran de quelques règles extraites

Ce type de visualisation décrit la dispersion des valeurs de confidence et de support de chacune des règles extraites ainsi qu'en utilisant l'ombrage de couleur on décrit la mesure de Lift, la clé de l'ombrage est fournie à droite du graphe.

La visualisation basée sur les graphes peut être utilisée pour obtenir une compréhension plus approfondie d'un ensemble plus restreint de règles et d'éléments.

Les techniques basées sur le graphique se concentrent sur la relation entre les éléments individuels de l'ensemble de règles. La figure 3.24 décrit les 100 meilleures règles

d'associations qui ont le diabète comme un item présent dans la partie de l'antécédent.

Ce type de visualisation est interactif c'est à dire on peut naviguer dans le graphe qui décrit les règles extraites en faisant notre analyse de facteurs de risques de la maladie de diabètes.

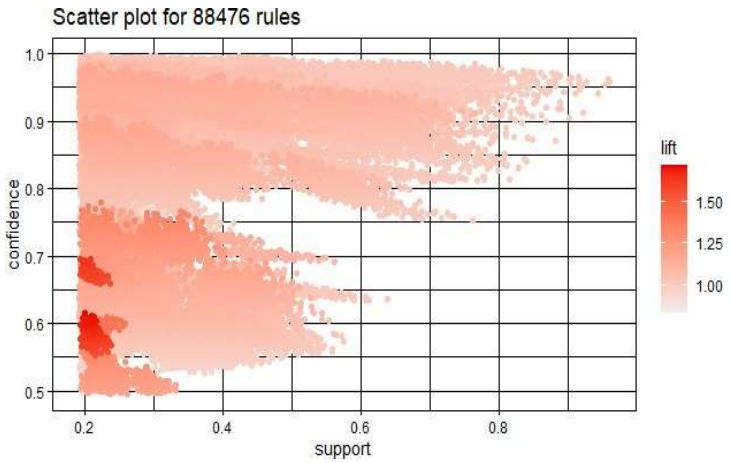

Figure 3.23 : Dispersion de quelques règles d'associations extraites

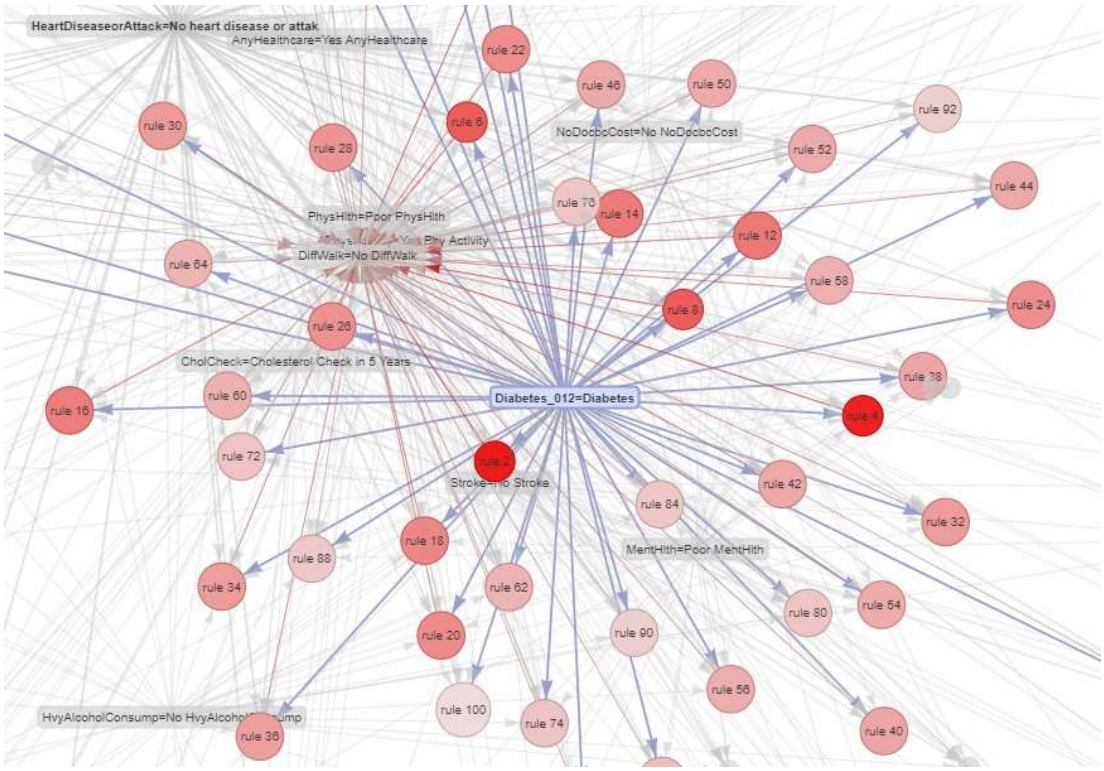

Figure 3.24 : Visualisation des règles d'associations en graphe

11. Conclusion

Après avoir élaboré cette comparaison en termes de temps d'exécution, l'implémentation distribuée de l'algorithme FP-growth en spark (DFPS) a donné de meilleur résultat par rapport aux autres. DFPS utilise trois étapes pour extraire les items fréquents en parallèle, la première s'agit de calculer l'ensemble d'items fréquents puis la répartition des motifs conditionnels ensuite l'étape cruciale d'extraire des itemsets fréquents en parallèle.

Chapitre 4 - Prédiction du diabéte par les algorithmes de machine learning

1. Introduction

Dans ce chapitre on s'intersse à l'utilisation des algorithmes d'apprentissage automatique pour la prédiction du diabète de type 2 qui est un dysfonctionnement du système de régulation de la glycémie, afin de réduire les risques de complication de cette maladie chronique sur la santé du patient.

Notre problématique nous permet de définir le diagnostic médical comme un processus de classification et l'utilisation de l'informatique devient de plus en plus fréquente pour mettre en œuvre cette classification bien que la décision de médecin soit le facteur le plus important dans le diagnostic. Les systèmes de classification sont d'une grande aide car ils réduisent les erreurs dues à la fatigue et au temps nécessaire au diagnostic. Avant d'entamer au problème de prédiction, il est souhaitable de donner un bref revu sur les différents types d'apprentissage.

12. Différents types d'apprentissage automatique

Les algorithmes peuvent être divisés en différentes catégories. On peut identifier les cinq bases différentes sur lesquelles l'apprentissage automatique peut être classé

a. Nature des données d'entrée
b. Nature du problème
c. Nature de l'algorithme
d. Nature de la solution
e. Nature des données de sortie

2.1. Classification basée sur la nature des intrants

En fonction du type de données d'entrée utilisées pour entraîner les algorithmes, les problèmes d'apprentissage automatique peuvent être classés en quatre catégories différentes.

2.1.1. Apprentissage supervisé

L'apprentissage supervisé est une technologie élémentaire mais stricte. Les opérateurs présentent à l'ordinateur des exemples d'entrées et les sorties souhaitées, et l'ordinateur recherche des solutions pour obtenir ces sorties en fonction de ces entrées. Le but est que l'ordinateur apprenne la règle générale qui s'occupe des entrées et des sorties.

L'apprentissage supervisé peut être utilisé pour faire des prédictions sur des données indisponibles ou futures (on parle alors de "modélisation prédicative"). L'algorithme essaie de développer une fonction qui prédit avec précision la sortie à partir des variables d'entrée
L'apprentissage supervisé peut se subdiviser en deux types :

- Classification : la variable de sortie est une catégorie.
- Régression : la variable de sortie est une valeur spécifique.

Les principaux algorithmes d'apprentissage supervisé sont les suivants : forêts aléatoires, arbres de décision, algorithme K-NN (k-Nearest Neighbors) [48], régression linéaire, algorithme de Naive Bayes [49,50], machine à vecteurs de support (SVM) [50,51], régression logistique et boosting de gradient [52].

2.1.2. Apprentissage non supervisé

Au contraire à l'apprentissage supervisé, l'apprentissage non supervisé ne comporte pas d'étiquettes. Un algorithme non supervisé reçoit un grand nombre de données et dispose des outils nécessaires pour comprendre les propriétés de ces données. A partir de ces données, il peut apprendre à grouper, regrouper et organiser les données de manière à ce qu'un humain (ou un autre algorithme intelligent) puisse intervenir et donner un sens aux données nouvellement organisées.

2.1.3. Apprentissage par renforcement

L'apprentissage par renforcement est assez différent de l'apprentissage supervisé et non supervisé. Alors que nous pouvons facilement voir la relation entre l'apprentissage supervisé et non supervisé (la présence ou l'absence d'étiquettes), la relation avec l'apprentissage par renforcement est un peu plus obscure. Certaines personnes essaient de rapprocher l'apprentissage par renforcement des deux types d'apprentissage en le décrivant comme un type d'apprentissage qui repose sur une séquence d'étiquettes dépendant du temps.

2.2. Classification basée sur la nature du problème

En fonction du type de problème, on peut classer le problème d'apprentissage automatique en trois catégories différentes.

Problème de classification: - la classification est un problème qui nécessite des algorithmes d'apprentissage automatique qui apprennent à attribuer une étiquette de classe aux exemples du domaine du problème.

Problème de régression: - la régression est un problème qui nécessite des algorithmes d'apprentissage automatique qui apprennent à prédire des variables continues.

Problème de clustering: - le clustering est un type de problème qui nécessite l'utilisation d'algorithmes d'apprentissage automatique pour regrouper les enregistrements de données donnés en un nombre spécifié d'unités cohésives.

2.3. Classification basée sur la nature de l'algorithme

En fonction de la nature de l'algorithme utilisé dans le processus d'apprentissage automatique, l'apprentissage automatique peut être classé en trois catégories.

Apprentissage automatique classique: - les algorithmes qui utilisent des équations statistiques et mathématiques pour dériver les relations dans les données de formation relèvent de cette catégorie. Ces algorithmes sont également appelés algorithmes d'apprentissage automatique statistique. Il a l'avantage de la capacité d'explication (la capacité d'expliquer la raison de certaines prédictions pour l'entrée donnée).

Réseaux de neurones: - Algorithmes inspirés du cerveau humain. Dans le processus de ces algorithmes, un modèle mathématique complexe avec un grand nombre de paramètres pouvant être entraînés est construit. Ces paramètres sont entraînés à l'aide de données d'entraînement. Les réseaux de neurones semblent assez prometteurs mais font face à de nombreuses limitations lorsque la complexité du modèle augmente.

Deep-Learning: le principe de base de l'apprentissage en profondeur est le même que celui des réseaux de neurones, mais certains progrès en termes d'architecture des couches sont introduits pour prendre les limites des réseaux de neurones. Les algorithmes d'apprentissage en profondeur sont capables d'apprendre les relations spatiales / temporelles dans les données d'apprentissage.

2.4. Classification basée sur la nature de la sortie

En fonction de la nature de la solution, on peut classer l'apprentissage automatique en deux catégories différentes. Naturellement, les algorithmes ML sont conçus pour apprendre les

données d'entrée historiques, faire des inférences à partir de ces données historiques et prédire la sortie des entrées futures. Pour prédire la sortie, le modèle peut adopter deux approches:

Modèles paramétriques: Modèles qui ne prennent en compte que les entrées futures pour prédire la sortie. Ces modèles prennent l'analogie des données d'entraînement et s'attendent à ce que la même analogie soit suivie dans les données de test invisibles. La régression linéaire et les réseaux de neurones sont des exemples de modèles paramétriques.

Modèles non paramétriques: la prédiction de la sortie dépend des caractéristiques d'entrée et des sorties précédentes que le modèle a prédites précédemment. Dans cette approche, la valeur de sortie prédite est dérivée des valeurs de sortie dans des scénarios similaires identifiés à partir des données d'apprentissage. KNN et les arbres de décision sont des exemples de modèles non paramétriques.

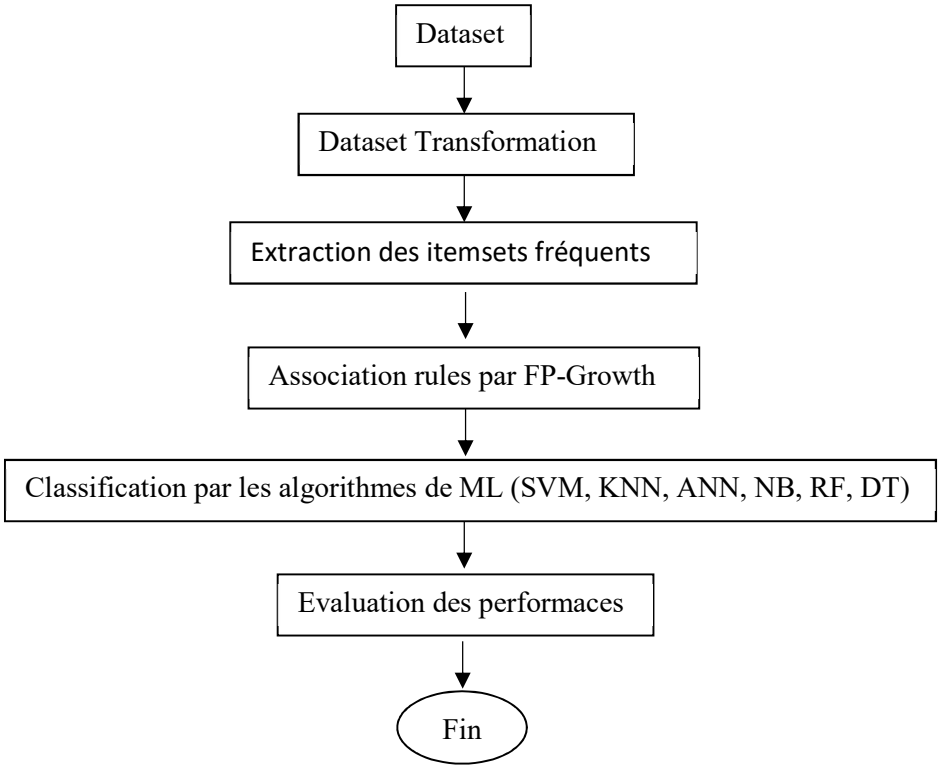

Figure 4.1 : Processus de prédiction

3. Processus de prédiction adopté

Ce travail est caractérisé par l'application des différents algorithmes de classification d'apprentissage supervisé (K nearest neighbors [49], Decision Trees, Random Forest [50],

Support Vector Machine [51], Naiıves Bayes [52] aux données diabétiques (PIMA) afin de déduire le meilleur algorithme qui donnera comme résultat une classification performante des patients en termes de taux de la précision et de la sensibilité du modèle. Figure 4.1 illustre le processus de la prédiction adoptée.

4. Algorithmes de l'apprentissage automatique utilisés

4.1. K nearest neighbors (KNN)

K nearest neighbors (KNN) ou K plus proche voisins en français est l'un des méthodes d'apprentissage supervisé le plus simple, utilisé pour résoudre des problèmes de classification et de la régression. Son fonctionnement est de classer les nouveaux points de données en fonction de la similarité aux points de données voisins.

- KNN est un algorithme qui ne fait aucunes hypothèses sur la structure des données et de la distribution, ce qui signifie qu'il s'agit d'un algorithme non paramétrique.
- Il est également appelé algorithme de l'apprenant paresseux, car il n'apprend pas immédiatement de l'ensemble d'apprentissage, mais stocke l'ensemble de données et, au moment de la classification, il exécute une action sur l'ensemble de données.
- KNN fonctionne par classification ou prédiction sur la base d'un nombre fixe (K) de points de données les plus proches de point d'entrée. Cela signifie que pour une valeur choisie de K, un point d'entrée serait classé ou devrait appartenir à la même classe que la classe la plus proche des nombres des points K voisins. [50,51]

Algorithme de construction de KNN

- Sélectionnez le nombre K des voisins.
- Pour chaque exemple de l'ensemble de données :
  - Calculez la distance entre l'exemple de requête et l'exemple actuel à partir des données.
  - Ajouter la distance et l'index de l'exemple à une collection ordonnée.
- Trier cette collection de distances et d'indices du plus petit au plus grand (par ordre croissant) ordonnée par les distances.
- Choisi les k premiers entrée de collections.
- Attribuer l'exemple de requête à la classe où maximal (classe le plus fréquent).

4.2. Decision Trees (Arbre de décision)

L'arbre de décision est un algorithme parmi les algorithmes d'apprentissage supervisé le plus utilisé et le plus pratique, qui est adapté pour résoudre tout type de problèmes (classifications ou régressions) tels-que :

- Un arbre de décision est une structure arborescente semblable à un organigramme où un nœud interne représente une caractéristique (ou un attribut), la branche représente une règle de décision et chaque nœud feuille représente le résultat, cette structure aide pour prendre la décision.
- C'est un algorithme non-paramétrique signifie qu'il n'y a pas d'hypothése sous-jacente sur la distribution des données. [48]

Mesure de sélection d'attribut

Le principal problème qui se pose lorsque la construction d'un arbre de décision si comment choisi ou sélectionné le meilleur attribut pour le nœud racine et qui sépare mieux l'ensemble de données. Pour résoudre ce problème il existe un technique qui appelé Mesure de sélection d'attribut qui contient deux mesures principales et populaires sont [52]:

- Indice de Gini
- Gain d'information

Algorithme de construction d'un arbre de décision

a. Sélectionne le meilleur attribut (nœud racine) : pour chaque attribut le gain d'information est calculé, et celui qui il est maximal est sélectionner et des branches sont créer pour chaque valeur de cette attribut.

b. Continuez la division : pour chaque branche s'étendant à partir de nœud, en répétant récursivement le processus.

c. Arrête la division si :
- Nous obtenons un nœud pur, c'est-à-dire un nœud qui ne contient que des points de données positifs ou négatifs.
- Nous obtenons très peu de points dans un nœud.
- On atteint une certaine profondeur de l'arbre.

4.3. Random Forest (forêts aléatoires)

Random Forest ou forêts aléatoires [54,56] est un algorithme d'apprentissage supervisé très populaire Il est également utilisé pour les problèmes de régression ou de classification basé sur un ensemble des algorithmes d'apprentissage, qui est un processus de combinaison de plusieurs algorithmes pour résoudre un problème complexe et améliorer les performances du modéle. C'est un algorithme qui créer de nombreux arbres de décision (c'est la raison pour laquelle il est appelé une forêt) sur divers sous-ensembles de l'ensemble de données. Elle prend la prédiction de chaque arbre et sur la base des votes majoritaires des prédictions, et elle prédit le résultat final. [28]

Algorithme de construction de Random forest

- S´electionnez des échantillons aléatoires à partir d'un ensemble de données d'entrainement.
- Créer des arbres de décision pour chaque échantillon (sous-ensembles). Ensuite on obtient le résultat de prédiction de chaque arbre de décision
- Pour les nouveaux points le vote sera effectué pour chaque résultat prédit.
- Sélectionnez le résultat de prédiction le plus voté comme résultat de prédiction final. [30]

4.4. Support Vector Machine (SVM)

Une machine à vecteurs de support [55], est un algorithme d'apprentissage automatique supervisé qui permet de résoudre des problèmes tant de classification que de régression ou de détection d'anomalie. Ils sont connus pour leurs solides garanties théoriques, leur grande flexibilité ainsi que leur simplicité d'utilisation même sans grande connaissance de data mining.

Les SVMs reposent sur l'idée de trouver un hyperplan soit une frontière dans le but de séparer les données en classes, de telle façon que la distance entre les différents groupes de données et la frontière qui les sépare soit maximale. Cette distance est aussi appelée « marge » et les SVMs sont ainsi qualifiés de « séparateurs à vaste marge », les « vecteurs de support » étant les données les plus proches de la frontière. Cette notion de frontière suppose que les données soient linéairement séparables (Figure 4.2), ce qui est rarement le cas. Pour y pallier, les SVMs reposent souvent sur l'utilisation de « noyaux ». Ces fonctions

mathématiques permettent de séparer les données en les projetant dans un espace vectoriel (Figure 4.3). La technique de maximisation de marge permet, quant à elle, de garantir une meilleure robustesse face au bruit et donc un modèle plus généralisable.

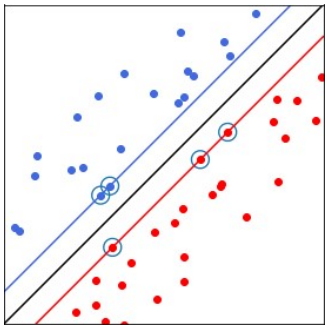

Figure 4.2 : Exemple de classe linéairement séparable.

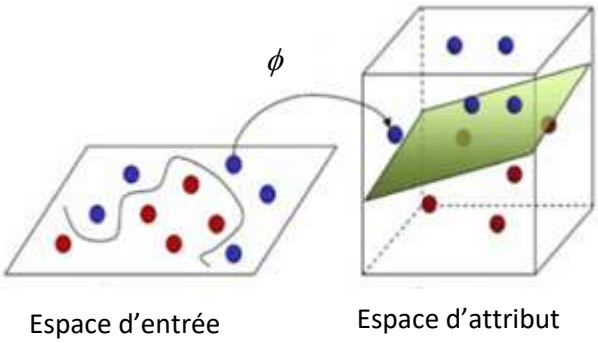

Figure 4.3 : Exemple d'un problème non linéairement séparable.

4.5. Naives Bayes

Naive bayésienne fait partie des algorithmes d'apprentissage automatique supervisé qui sont principalement utilisés pour la classification [58]. C'est un classificateur probabiliste simple basé sur l'application de théorème de Bayes et qui aide à construire des modèles d'apprentissage automatique rapides qui peuvent faire des prédictions rapides.

La classification naïve bayésienne repose sur l'hypothèse que toutes les caractéristiques sont conditionnellement indépendantes les unes des autres. Le théorème est donné par la formule suivante :

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{4.1}$$

- A et B sont des événements.
- (A) est la probabilité d'observer l'événement A
- P (B) est la probabilité d'observer l'événement B.
- P (A | B) est la probabilité conditionnelle d'observer A, sachant qu'un autre événement B de probabilité non nulle s'est réalisé.

Dans un problème de classification, notre tâche est de trouver l'étiquette la plus probable A, étant donné les caractéristiques B, le théoreme de Bayes devient :

$$P(y|x_1,...,x_n) = \frac{P(x_1,..,x_n|y)P(y)}{P(x_1,...,x_n)} \tag{4.2}$$

Où n représente le nombre de caractéristiques, y est l'évènement qu'on cherche à classer.

Par conséquent, en tenant compte de l'hypothèse d'indépendance, Bayes prédit la classe qui constitue la probabilité la plus élevée:

$$\bar{y} = \arg\max_y P(y)\prod_{i=1}^{n} P(x_i|y) \tag{4.3}$$

5.4. Réseaux de neurones artificiels :

Le terme réseau de neurones est une référence à la neurobiologie. Originalement, ce concept est inspiré du fonctionnement des neurones du cerveau humain, apprend essentiellement de l'expérience.

Le fonctionnement exact du cerveau humain est encore un mystère, mais certains aspects sont connus. En particulier l'élément fondamental du cerveau est un type spécifique de cellule incapable de se régénérer. Ces cellules nous fournissent la capacité de nous rappeler, de penser et de réagir en basant sur des expériences antérieures. Les réseaux de neurones artificiels tentent de reproduire que les éléments les plus fondamentaux de cet organisme complexe et puissant.

Un réseau de neurones est une organisation hiérarchique de neurones connectés entre eux. Ces derniers transmettent un message ou un signal à d'autres neurones en fonction des paramètres d'entrés reçus et forment un réseau complexe. La figure 4.4 illustre une représentation simpliste d'un réseau de neurones de base. Un réseau de neurones [59,60] est en général composé d'une

succession de couches. Ce modèle comporte trois ensembles de règles : multiplication, sommation et activation. Les données d'entrées sont consommées par les neurones de la première couche cachée. Chaque couche peut avoir un ou plusieurs neurones. La connexion entre deux neurones de couches successives aurait un poids associé qui définit l'influence de l'entrée sur la sortie du neurone suivant et éventuellement sur la sortie finale globale. Sur chaque neurone artificiel, chaque valeur d'entrée est multipliée par le poids correspondant [61].

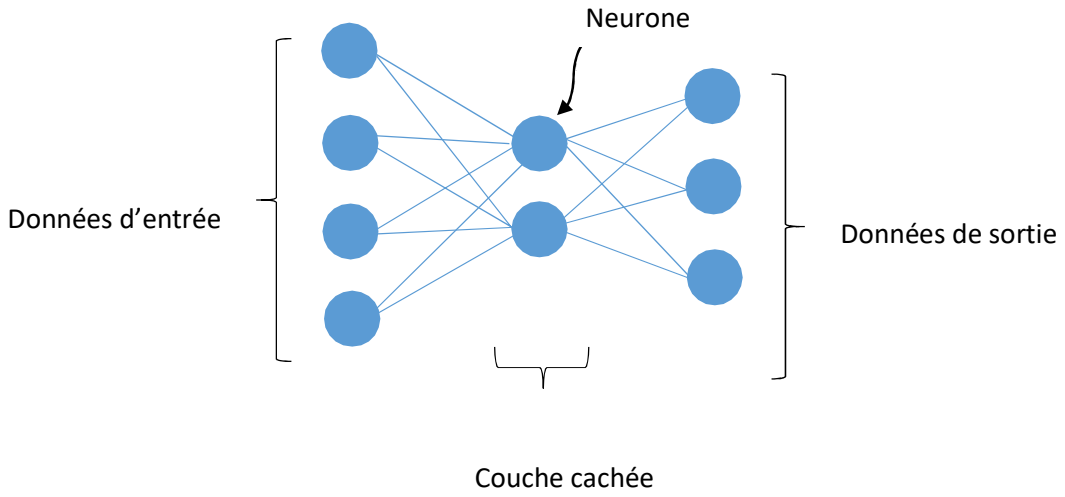

Figure 4.4 : Exemple simple d'un réseau de neurones artificiel.

Ensuite, le résultat obtenu sera additionné à ce qu'on appelle le biais pour lui appliquer à la fin une fonction d'activation et la valeur de sortie sera donc de la forme :

$$y = F(\sum_{i=0}^{m} \omega_i . x_i + b) \tag{4.4}$$

avec

- ω_i : représentent les poids.
- x_i : représentent les données d'entrées.
- b : représente le biais.
- F : représente la fonction d'activation.
- y : représente la valeur de sortie.

Dans un réseau de neurones, les poids initiaux seraient tous aléatoires lors de la formation du modèle, l'inconnu principal est sa fonction de transfert, elle peut être toute fonction mathématique. Le choix de cette fonction dépend du problème posé. Dans la majorité des cas, elle est choisie non linéaire. Cela permet aux réseaux de neurones d'apprendre des transformations non linéaires et complexes des données. Les fonctions d'activation les plus utilisées sont : la fonction Heaviside, la fonction sigmoide, la fonction de la tangente hyperbolique et la fonction rampe.

5. Critères d'évaluation et Mésure de performance

Le critère d'évaluation est un facteur clé à la fois dans l'évaluation de la performance de classification et guidance de la modélisation de classificateur. Pour comparer de façon synthétique les performances des différentes méthodes et de différents outils retenus pour notre étude nous utilisons la matrice de confusion donnée ci-dessous avec :

- Vrai positive (VP) : les cas positives classé positives
- Vrai négative (VN) : les cas négatifs classé négative
- Faux positive (FP) : les cas positive classé négative
- Faux négative (FN) : les négatives classées positive

	Diabétique (1)	Non-Diabétique (0)
Diabétique (1)	VP	FP
Non-Diabétique (0)	FN	VN

Les mesures de performance utilisées sont : Accuracy (L'exactitude), Precision (Précision), Recall (Sensibilité), Spécificité, et F-measure.

- **Accuracy (L'exactitude):** Pour prédire les performances d'un classeur sur les données non observées, on utilise la méthode d'évaluation nommée Accuracy en anglais, elle est définie comme le ratio entre les exemples bien classés et le nombre total d'exemples :

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \qquad (4.5)$$

- **Precision :** La précision est la proportion d'élèments bien classés pour une classe donnée.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{4.6}$$

- **Recall (Sensibilité) :** La sensibilité permet de représenter la fréquence des réponses positives au test. En d'autres termes, c'est la probabilité conditionnelle d'avoir un résultat du test positif si M est présente.

$$\text{Sensiblité} = \frac{TP}{TP + FN} \tag{4.7}$$

- **Spécificité :** La spécificité représente la fréquence des réponses négatives au test parmi les patients qui ne sont pas atteints de la maladie M. En outre, c'est la probabilité conditionnelle d'avoir un résultat du test négatif avec l'absence de M. Elle correspond à 1 s'il n'existe pas de faux positif.

$$\text{Specificité} = \frac{TN}{TN + FP} \tag{4.8}$$

- **F-mesure :** Représente la moyenne pondérée de Precision et Recall. Ces deux dernières apportent une contribution relative équitable au F-measure.

$$\text{F-measure} = \frac{2 * \text{Re}\,call * \text{Pr}\,ecision}{\text{Re}\,call + \text{Pr}\,ecision} \tag{4.9}$$

6. Résultats expérimtaux

Cette partie relate les résultats de l'étude comparative entre les outils de classification des données en utilisant les six algorithmes (RF, DT, ANN, SVM, KNN, NB) appliqués à la base de données (PIMA).

L'objectif de chaque algorithme est de parvenir à mieux classer les futures observations en minimisant l'erreur de classification. Les algorithmes ont été testé en utilisant trois scénarios : trois facteurs (glucose, IMC et age), cinq facteurs (glucose, IMC, age, insulin et skin) et pour le dernier processus on a utilisé tous les motifs. Les tableaux ci-dessous montrent les matrices de confusion pour chaque algorithme utilisé.

6.1.Prediction en utilisant tous les motifs de la base

Les tableaux ci-dessous prensentent respectivement les matrices de confusion obtenues par utilisation des six techniques de classification (Réseau de neurones, Machine à Vecteurs Support, arbre de décision, foret aleatoire, Naive Bayes et k-plus proche voisin). Ces algorihmes ont été appliqués à l'ensemble d'apprentissage pour prédire les résultats de l'ensemble de test qui ont abouti au résultat presenté au Tableau 4.8. Une observation du tableau est que la précision de toutes les méthodes est plus sur notre ensemble de données puisque la première a plus de champs pertinents pour évaluer le risque de diabète. Le modèle Random Forest dispose du meilleur score en termes d'Accuracy et precision.

Tableau 4.1 : Matrice de confusion pour KNN

	Diabetic	No_diabetic
Diabetic	VP = 134	FN = 16
No_diabetic	FP = 28	VN = 53

Tableau 4.2 : Matrice de confusion pour Forêt aleatoire.

	Diabetic	No_diabetic
Diabetic	VP = 139	FN = 11
No_diabetic	FP = 16	VN = 65

Tableau 4.3 : Matrice de confusion pour SVM.

	Diabetic	No_diabetic
Diabetic	VP = 125	FN = 25
No_diabetic	FP = 29	VN = 52

Tableau 4.4 : Matrice de confusion pour ANN.

	Diabetic	No_diabetic
Diabetic	VP = 134	FN = 16
No_diabetic	FP = 16	VN = 65

Tableau 4.5 : Matrice de confusion pour DT.

	Diabetic	No_diabetic
Diabetic	VP = 133	FN = 17
No_diabetic	FP = 21	VN = 60

Tableau 4.6 : Matrice de confusion pour NB.

	Diabetic	No_diabetic
Diabetic	VP = 130	FN = 31
No_diabetic	FP = 21	VN = 49

Tableau 4.7 : Metriques pour différents algorithmes.

	Accuracy %	Precision %	Sensibilité %	Specificité %	F-mesure %
KNN	81	80	77	65,43	78
RF	**88,31**	**88**	**86**	**80,24**	**87**
SVM	77	74	74	64,19	74
ANN	**86**	**85**	**85**	**80,24**	**85**
NB	**77,48**	**80,74**	**86,09**	**74**	**87,86**
DT	84	82	81	70	82

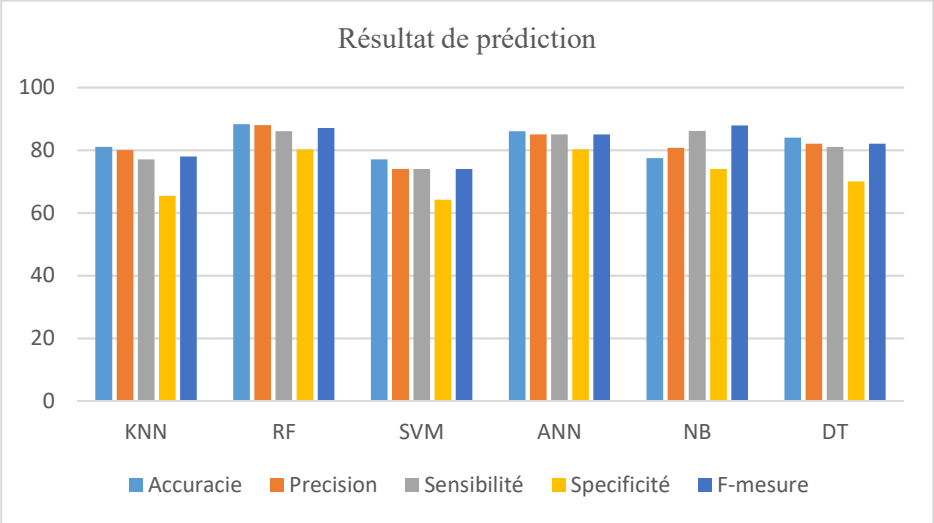

Figure 4.5 : Comparaison des résultats avec les différentes métriques (avec tous les motifs)

On peut dire que le modèle Random Forest est celui qui dispose des meilleures performances en générale en tenant en compte le fait que ses scores pour les quatres métriques sont plus ou moins proches les uns des autres avec le meilleur recall possible, c'est-à-dire une meilleure capacité à retrouver toutes les instances positives (de classe malade qui est la classe d'intérêt).

À partir du Tableau 4.8 et la figure 4.5 qui montre les performances graphiques de tous les algorithmes de classification sur la base d'instances classées. Nous pouvons conclure que l'algorithme de classification Forêt aléatoire surpasse comparativement les autres algorithmes. L'algorithme Forêt aléatoire est considéré comme la meilleure méthode d'apprentissage automatique supervisé de cette expérience car il donne une plus grande accurancie par rapport aux autres algorithmes de classification avec une précision de 88,31 %.

6.2.Prediction par trois motifs

Les attributs (items) sont glucose, IMC, et l'age. Les matrix de confusion par utilisation des algorithmes Random forest et Naive Bayes sont illustés respectivement aux Tableau 4.9 et Tableau 4.10. Les valeurs de précision, de l'Accuracy, de la spécificité, du recall et de F-mesure sont données au Tableau 4.11 et illustés dans la Figure 4.6. L'algorithme Forêt aléatoire est considéré comme la meilleure méthode d'apprentissage automatique supervisé de cette expérience.

Tableau 4.8 : Matrice de confusion pour Forêt aleatoire (trois motifs).

	Diabetic	No_diabetic
Diabetic	VP = 123	FN = 28
No_diabetic	FP = 31	VN = 48

Tableau 4.9 : Matrice de confusion pour Naive Bayes.

	Diabetic	No_diabetic
Diabetic	VP = 133	FN = 30
No_diabetic	FP = 18	VN = 49

Tableau 4.10 : Résultat pour 3 attributs.

	Accuracy (%)	Precision (%)	Sensibilité (%)	Specificité (%)	F-mesure (%)
RF	**75,22**	**82,12**	**80,52**	**64,47**	**81,31**
NB	**79,13**	**81,60**	**88,08**	**62,03**	**84,71**

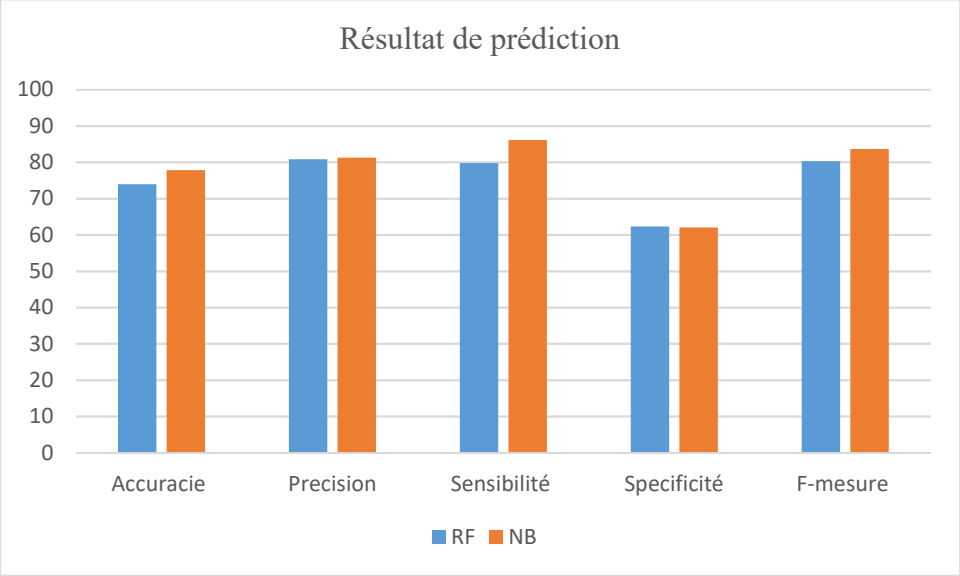

Figure 4.6 : Comparaison des résultats avec les différentes métriques (trois motifs)

6.3. Prédiction par cinq attributs

Les attributs utilisés sont glucose, IMC, age, insulin et skin. Les matrix de confusion par utilisation des algorithmes Random forest et Naive Bayes sont illustés respectivement aux Tableau 4.12 et Tableau 4.13. Les valeurs de précision, de l'Accuracy , de la spécificité, du recall et de F-mesure sont données au Tableau 4.14 et illustés dans la Figure 4.7. L'algorithme Forêt aléatoire est considéré comme la meilleure méthode d'apprentissage automatique supervisé de cette expérience.

Tableau 4.11 : Matrice de confusion pour Forêt aleatoire.

	Diabetic	No_diabetic
Diabetic	VP = 121	FN = 30
No_diabetic	FP = 30	VN = 49

Tableau 4.12 : Matrice de confusion pour Naive Bayes.

	Diabetic	No_diabetic
Diabetic	VP = 130	FN = 30
No_diabetic	FP = 21	VN = 49

Tableau 4.13 : Résultat pour cinq motifs.

	Accuracy (%)	Precision (%)	Recall (%)	Specifité (%)	F-mesure (%)
RF	**73,91**	**80,79**	**79,74**	**62,34**	**80,26**
NB	**77,83**	**81,25**	**86,09**	**62,03**	**83,60**

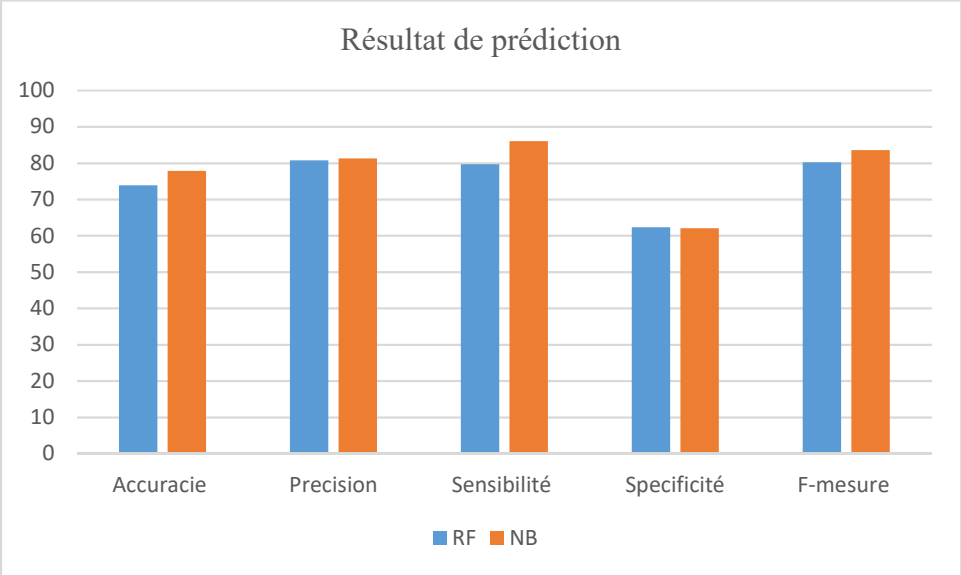

Figure 4.7 : Comparaison des résultats avec les différentes métriques (cinq motifs).

7. Conclusion

Ce travail nous a amené au développement d'une étude comparative entre six algorithmes ANN, DT, RF, SVM et KNN appliqués à une base de données sélectionnée du domaine médical. L'objectif est de donner la possibilité aux professionnels d'étudier et d'appliquer les différentes techniques et outils de classifications. Donc nous avons tout d'abord étudié les approches et les notions fondamentales de la classification des données.

Par application des algorithmes mentionnés précedement, les résultats ont montré que l'algorithme de forêt aléatoire est considéré comme le meilleure algorithme d'apprentissage automatique pour le cas de tous les motifs mais pour les cas (3 motifs) ou bien (5 motifs) Naive Bayes est meilleur par rapport à l'algorithme de Forêts aléatoires.

CONCLUSION GÉNÉRALE

La fouille de données est une nouvelle discipline qui consiste à extraire des connaissances cachées dans les données en utilisant différentes techniques empruntées aux statistiques, à l'informatique et aux mathématiques. Les méthodes de fouille de données ont été utilisées dans de nombreux domaines de la recherche clinique tels que le clustering, la classification et l'extraction des motifs fréquents et les règles d'association afin de découvrir des informations significatives à partir des données massives.

Des approches de data mining sont présentées dans ce mémoire pour extraire de connaissances sur le diabète. Le premier objectif de ce travail consiste à identifier les variables ayant le plus d'impact sur les patients diabétiques à partir des règles d'association extraites, après l'extraction des motifs fréquents par un ensemble d'algorithmes, dont nous faisons une étude comparative entre eux. Le deuxième objectif est d'utiliser la frame work Spark afin de réduire le temps d'écution dans le cas des bases de données massives lors de l'extraction des règles d'associations. Le troisième objectif est l'utilisation des algorithmes de machine learning pour prédire le diabéte.

Dans la première contribution, nous avons comparé cinq algorithmes les plus cités, à savoir Apriori, FP-Growth, CFP-Growth, ECLAT et CHARM, ces algorithmes ont été choisis pour faire l'extraction des motifs fréquents et être évaluer selon les meilleures performances en termes de temps d'exécution, de la taille de mémoire consommée et du nombre d'itemsets générés. A chaque fois qu'on varie le support minimal, on visualise le résultat. Ces résultats expriment que si le degré de support minimal augmente, le temps d'exécution et le nombre d'itemsets générés diminuent pour tous les algorithmes. On remarque que l'algorithme CHARM prend les plus grandes valeurs de temps d'exécution, par contre les algorithmes FP-Growth et CFP-Growth sont bien performants par rapport à CHARM. La consommation de la mémoire pour l'algorithme CFP-Growth confirme sa performance dans la première comparaison.

La deuxième contribution consiste à utiliser une base de données massives, dans ce cas il est necessaire de faire appel aux parallelismes des algorithmes. Les algorithmes DFPS, YAFIM, et RAPRIORI ont été choisis et évalués selon les meilleures performances en termes de temps d'exécution et de la taille de mémoire consommée par chaque algorithme. Il a été observé que l'algorithme DFPS a de meilleures performances globales que YAFIM et à RApriori.

Dans la troisième contribution, nous avons utlisé les algorithmes de machine learning afin de prédire le diabète. Six algoritmes ont été proposés qui sont KNN, ANN, DT, RF, NB et SVM. La prédiction a été faite en utilisant trois façons différentes : trois attributs, cinq attributs et huit

attributs. Les résultats expérimentaux ont montré que l'algorithme de RF est plus performant lors de l'utilisation de huit atributs. Nous résumons les résultats dans les points suivants :

- Un taux de glucose élevé indique un diabète.
- Les personnes ayant un nombre élevé de grossesses sont les plus susceptibles d'être atteintes de diabète.
- Les personnes souffrant d'insuffisance pondérale (IMC très bas) risquent également de contracter le diabète si l'épaisseur de leur peau est relativement élevée.
- Les personnes en surpoids (IMC élevé) courent un risque plus élevé de contracter le diabète.
- Le glucose est le facteur le plus important dans la détermination de l'apparition du diabète suivi de l'IMC et de l'âge.
- D'autres facteurs tels que la fonction de pedigree du diabète, les grossesses, la tension artérielle, l'épaisseur de la peau et l'insuline contribuent également à la prédiction.
- Le niveau de glucose est l'une des premières choses qui est réellement surveillée chez les patients à haut risque.
- Une augmentation de l'IMC pourrait également indiquer un risque de développer un diabète de type II.

PERSPECTIVES

Bien que les objectifs visés ont été atteints, mais il reste toujours des perspectives et des améliorations qui peuvent être réalisées, nous nous proposons :

- Inclure les différents types de diabètes pour ensuite faire la distinction.
- Envisager une base de données plus consistante et plus adéquate et intégrer d'autres méthodes de prédiction.
- Améliorer les algorithmes d'extraction des motifs fréquents pour être optimisés pour le Big data.
- Utuliser une base de données Marocainne.

REFERENCES

[1] L. Szathmary, S. Maumus, P. Petronin, Y. Toussaint, and A. Napoli. « Vers l'extraction de motifs rares », In Acte de la conférence Extraction et gestion des connaissances (EGC'06), France, volume 2, pages 499–510, Janvier 2006.

[2] Hacène Cherf, « Etude et réalisation d'un système d'extraction de connaissances à partir de textes », thèse de doctorat, université Henri Poincaré, Nancy 1, 15, november 2004

[3] Usama Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth, « The KDD Process for Extracting Useful Knowledge from Volumes of Data », Communications of the ACM November 1996/Vol. 39, No. 11

[4] B.Kalpana, R.Nadarajan, and J.Senthil Babu, « A Galois Lattice Framework to Handle Updates in the Mining of Closed Itemsets in Dynamic Databases », COMPUTE '08: Proceedings of the 1st Bangalore Annual Compute Conference , January 2008, https://doi.org/10.1145/1341771.1341788

[5] Border G. Liu, J. Li, and L. Wong, « A new concise representation of frequent itemsets using generators and a positive border », Knowledge and Information Systems, 17(1) :35–56, 2008.

[6] L. Szathmary, A. Napoli, and P. Valtchev, « Towards rare itemset mining », In Proceedings of the 19th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'07), October, 2007, Greece, volume 1, pages 305–312, 2007

[7] P. Lenca, P. Meyer, B. Vaillant, and S. Lallich, « On selecting interestingness measures for association rules : User oriented description and multiple criteria decision aid », European Journal of Operational Research, vol. 184, no. 2, pp. 610 – 626, 2008.

[8] Addi Ait-Mlouk, « Fouille de données et analyse de qualité des régles d'association dans les bases de données massives: application dans le domaine de la sécurité routière », thèse de doctorat, université cadi Ayyad, Maroc, 2018.

[9] T. I. R. Agrawal and A. Swami, « Mining association rules between sets of items in large databases », in International Conference on Management of Data, (Washington, U.S.A.), pp. 207–216, ACM, 2013.

[10] Gosta Grahne, Jianfei Zhu, « Fast Algorithms for Frequent Itemset Mining Using FP-Trees », IEEE Transactions on knowledge and data engineering vol.17, no 10, October 2005

[11] C. Luchesse, s. Orlando, and R. Perogo, « DCI-Closed: Afast and memory efficient algorithm to mine frequent closed itemsets », In B. Goethals, M. J. Zaki, and R. Bayardo, editors, Proceedings of the IEEE ICDM Workshop on Frequent Itemset Mining

Implementations (FIMI 2004), volume 126 of CEUR Workshop Proceedings, Brighton, UK, 1 November 2004.

[12] Zhi-Chao Li, Pi-Lian He, Ming Lei, « A High Efficient Aprioritid Algorithm For Mining association Rule », Proceedings of the Fourth International Conference on Machine Learning and Cybernetics, Guangzhou, 18-21 August 2005.

[13] Anup Bhat B et al, « A Dynamic Itemset Counting Based Two-Phase Algorithm for Mining High Utility Itemsets », 15th IEEE India Council International Conference (INDICON) 978-1-5386-8235-7/18/$31.00 ©2018 IEEE.

[14] S. Sharmila and S. Vijayarani, « Improved RElim and FIN Algorithm for Frequent Items Generation », Proceedings of the International Conference on Inventive Computing and Informatics (ICICI 2017) IEEE Xplore Compliant - Part Number: CFP17L34-ART, ISBN: 978-1-5386-4031-9, 2017.

[15] C. Borgelt, and Xiaomeng Wang, « SaM: A Split and Merge Algorithm for Fuzzy Frequent Item Set Mining », IFSA/EUSFLAT Conf. 2009.

[16] Christian Borgelt , « Simple Algorithms for Frequent Item Set Mining », Advances in Machine Learning III, SCI 263, pp. 351–369. springerlink.com c Springer-Verlag Berlin Heidelberg 2010.

[17] Manjit kaur, Urvashi Grag, « ECLAT Algorithm for Frequent Itemsets Generation », International Journal of Computer Systems, ISSN-(2394-1065), Vol. 01, Issue,03, December, 2014.

[18] Chin-Hoong Chee et al., « Algorithms for frequent itemset mining: a literature review », Artificial Intelligence Review volume 52, pages 2603–2621, 2019.

[19] Yves Bastide, « Pascal : un algorithme d'extraction des motifs fréquents », Technique et science informatiques. Volume 21 - n◦ 1, 2002.

[20] Farzad Nadi and Atefeh Foroo zandeh, « New Method for Mining Maximal Frequent Itemsets based on Graph Theory », 4th international conference on computer and knowledge engineering, 2014.

[21] Roberto J. Bayardo, « Efficiently Mining Long Patterns from Databases », SIGMOD '98: Proceedings of the 1998 ACM SIGMOD international conference on Management of dataJune 1998 Pages 85–93https://doi.org/10.1145/276304.276313.

[22] Mohammed J. Zaki, « Scalable Algorithms for Association Mining », IEEE transactions on knowledge and data engineering, vol. 12, no. 3, may/june 2000.

[23] Maryam Shekofteh, Amir Masoud Rahmani, and Mashaallah Abbasi Dezfuli, « A-close+: An Algorithm for Mining Frequent Closed Itemsets », 2008 International Conference on Advanced Computer Theory and Engineering. 978-0-7695-3489-3/08 $25.00 © 2008 IEEE DOI 10.1109/ICACTE.2008.135.

[24] Mohammed J Zaki and Ching-Jui Hsiao, « Charm : An efficient algorithm for closed itemset mining », In Proceedings of the 2002 SIAM international conference on data mining, pages 457–473. SIAM, 2002.

[25] Y. Fakir and R. Elayachi, « Mining Frequent Pattern by Titanic and FP-Tree algorithms », International Journal of Scientific Research in Computer Science Engineering and Information Technology, vol.5, issue 2, 2020

[26] J Pei, J Han, R Mao, « CLOSET: An efficient algorithm for mining frequent closed itemsets », workshop on research issues in data mining, 2020.

[27] J. Wang, J. Han, and J. Pei, « CLOSET+: Searching for the best strategies for mining frequent closed itemsets », proc. Int'l Conf. Knowledge Discovery and Data Mining, PP. 236-245, 2003

[28] G¨osta Grahne and Jianfei Zhu, « Efficiently Using Prefix-trees in Mining Frequent Itemsets », Proceedings of FIMI'03 Workshop on Frequent Itemset Mining Implementations, Melbourne 2003.

[29] Takeaki Uno, Tatsuya Asai, Yuzo Uchida, and Hiroki Arimura, « LCM: An efficient algorithm for enumerating frequent closed item sets », Proceedings of FIMI'03 Workshop on Frequent Itemset Mining Implementations, Melbourne 2003.

[30] J. Tahmores Nezhad, M.H. Sadreddini, « PTclose: A novel algorithm for generation of closed frequent itemsets from dense and sparse datasets », Proceedings of the World Congress on Engineering 2007 Vol I WCE 2007, July 2 - 4, 2007, London, U.K.

[31] Gerd Stumme et al, « Computing iceberg conceptlattices with TITANIC », Knowledge Engineering42 (2002) 189–222

[32] Laszlo Szathmary, Amedeo Napoli, Sergei O. Kuznetsov, « ZART: A Multifunctional Itemset Mining Algorithm », 5th International Conference on Concept Lattices and Their Applications (CLA ’07), Oct 2007, Montpellier, France. pp.26–37.

[33] Tarek Hamrouni, Sadok Ben Yahia, and Engelbert Mephu Nguifo, « Efficient construction of the lattice of frequent closed patterns and simultaneous extraction of generic bases of rules », Mathematics and Social Sciences, N°. 195, 2011(3), p. 5–54.

[34] Doug Burdick, Manuel Calimlim, Johannes Gehrke, « MAFIA: A Maximal Frequent Itemset Algorithm for Transactional Databases », IEEE Xplore, Proceedings 17th International Conference on Data Engineering,

[35] Jian Pei, Jiawei Han, Hongjun Lu, Shojiro Nishio, Shiwei Tang and Dongqing Yang, « H-Mine: Fast and space-preserving frequent pattern mining in large databases », IIE Transactions (2007) 39, 593–605.

[36] Bhawna Nigam, A. Nigam and P. Dalal, « Comparative Study of Top 10 Algorithms for Association Rule Mining », International Journal Of Computer Sciences And Engineering 5(8):190-195, 2017.

[37] X. Jin, B. W. Wah, X. Cheng, Y. Wang, « Significance and challenges of big data research », Big Data Research 2 (2), visions on Big Data, 2015.

[38] M. Solaimani, M. Iftekhar, L. Khan, B. Thuraisingham, « Statistical technique for online anomaly detection using Spark over heterogeneous data from multi-source VMware performance data », IEEE International Conference on Big Data, 2014.

[39] Norulhidayah Isa, Siti Khadijah Neddy, Norizan Mohamed, « Association Rule Mining using FP-Growth Algorithm to Prevent Maverick Buying », IEEE 11th IEEE Symposium on Computer Applications & Industrial Electronics (ISCAIE), 2021.

[40] O. Y. Al-Jarrah, P. D. Yoo, S. Muhaidat, G. K. Karagiannidis, K. Taha, « Efficient machine learning for big data: A review », Big Data Research 2 (3), 2015.

[41] Qiu, H., Gu, R., Yuan, C., & Huang, Y., « YAFIM: A Parallel Frequent Itemset Mining Algorithm with Spark, « IEEE International Parallel & Distributed Processing Symposium Workshops, 2014.

[42] Rathee S, Kaul M, Kashyap A., « R-Apriori: an efficient apriori based algorithm on Spark », PIKM'15 October 19, 2015.

[43] Rice Novita, Mustakim and Febi Nur Salisah, « Determination of the relationship pattern of association topic on Al-Qur'an using FP-Growth Algorithms », Annual Conference on Computer Science and Engineering Technology (AC2SET) 2020.

[44] Shi, X., Chen, S., & Yang, H., « DFPS: Distributed FP-growth algorithm based on Spark », In 2017 IEEE 2nd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC) (pp. 1725-1731). 2017.

[45] Yaling Xun, Jifu Zhang, Haiflug Yang and Xia Qin, « HDPFP-DC: a parallel frequent itemset using Spark », Parallel computing 101, 2021.

[46] B. Premamayudu et al, « Diabetes Prediction Using Machine Learning KNN -Algorithm Technique », International Journal of Innovative Science and Research Technology, Volume 7, Issue 5, 2022.

[47] Muhammad Exell Febriana et al, « Diabetes prediction using supervised machine learning », 7th International Conference on Computer Science and Computational Intelligence 2022.

[48] S. D. Jadhav and H. P. Channe, « Comparative Study of K-NN, Naive Bayes and Decision Tree Classification Techniques »," Int. J. Sci. Res., vol. 5, no. 1, pp. 1842–1845, 2016.

[49] Heri Kuswanto*, Rizky Mubarok, « Classification of Cancer Drug Compounds for Radiation Protection Optimization Using CART », The Fifth Information Systems International Conference 2019

[50] X. Wu, S. Wang, and Y. Zhang, « Review of K nearest neighbor algorithm theory and application », Computer Engineering and Application, vol. 53, no. 21, pp. 1–7, 2017.

[51] S Appavu alias Balamurugan; U. V Swathi, « An amalgam KNN to predict diabetes mellitus », 2013 IEEE International Conference ON Emerging Trends in Computing, Communication and Nanotechnology (ICECCN)

[52] Ulka Shirole Manjusha Joshi Pritish Bagul, Cardiac, « diabetic and normal subjects classification using decision tree and result confirmation through orthostatic stress index », Informatics in Medicine Unlocked Volume 17, 2019, 100252

[53] W. Xu, L. Jiang, « An attribute value frequency-based instance weighting filter for naive Bayes », Journal of Experimental & Theoretical Artificial Intelligence 31(4), 225–236 (2019).

[54] V. Svetnik, A. Liaw, C. Tong, J.C. Culberson, R.F. Sheridan, B.P. Feuston, « Random Forest: A Classification and Regression Tool for Compound Classification and QSAR Modeling », J. Chem. Inf. Comput. Sci., 43 (6) (2003), pp. 1947-1958

[55] A. Matsumoto, S. Aoki, H. Ohwada, « Comparison of Random Forest and SVM for Raw Data in Drug Discovery: Prediction of Radiation Protection and Toxicity Case Study », International Journal of Machine Learning and Computing, 6 (2) (2016), pp. 145-148

[56] Marijana Zekić-Sušaca AdelaHasa Marinel aKnežev, « Predicting energy cost of public buildings by artificial neural networks, CART, and random forest Forest », Neurocomputing, Volume 439, 7 June 2021, Pages 223-233

[57] Mani Butwall and Shraddha Kumar, « A Data Mining Approach for the Diagnosis of Diabetes Mellitus using Random Forest Classifier », International Journal of Computer Applications, Volume 120 - Number 8,2015.

[58] H. Kuswanto, R. Mubarok, H. Ohwada, « Classification Using Naive Bayes to Predict Radiation Protection in Cancer Drug Discovery: a Case of Mixture Based Grouped Data », International Journal of Artificial Intelligence, 17 (1) (2019), pp. 186-203

[59] R. S. Wahono, N. S. Herman, and S. Ahmad, « Neural Network Parameter Optimization Based on Genetic Algorithm for Software Defect Prediction », vol. 20, no. 10, pp. 1951–1955, 2014.

[60] Vaishal iIngle Prashant Ambad, « Diabetic retinopathy classifier with convolution neural network », Materials Today: Proceedings, Volume 47, Part 1, 2021, Pages 185-1