# Introduction de l'élagage pour l'extraction de règles d'association de classe sans génération de candidats

Emna Bahri*, Stéphane Lallich*

*Laboratoire ERIC, Université de Lyon, 5, avenue Pierre Mendès-France, 69500, Bron
emna.bahri | stephane.lallich@univ-lyon2.fr, http://eric.univ-lyon2.fr

**Résumé.** Ce travail porte sur l'extraction des règles d'association prédictive en apprentissage supervisé. On sait que les algorithmes d'extraction de règles d'association développés en apprentissage non-supervisé extraient toutes les règles dont le support et la confiance dépassent des seuils préfixés. L'extraction des règles d'association prédictive à l'aide de ces algorithmes, avec un filtrage a posteriori des règles, pose plusieurs problèmes. Dans la première étape, on extrait des itemsets fréquents inutiles, ceux qui ne contiennent pas la classe, alors que la seconde étape peut être simplifiée, puisqu'un itemset contenant la classe ne donne lieu qu'à une seule règle de classe. Pour travailler avec les seuils de support les plus bas possibles, nous proposons FCP-Growth-P une adaptation de FP-Growth qui a différents avantages. Tout d'abord, FCP-Growth-P élimine les itemsets fréquents qui ne contiennent pas d'item de classe. D'autre part, pour ne pas désavantager la classe la moins nombreuse, nous adaptons le seuil de support afin de travailler avec le même seuil de support à l'intérieur de chaque classe. Enfin, pour un meilleur élagage, FCP-Growth-P intègre la condition d'élagage de Li sur la spécialisation des règles.

## 1 Introduction

Cet article porte sur l'apprentissage supervisé par règles à partir des tableaux attributs-valeurs issus des grandes bases de données. Ce type d'apprentissage est particulièrement intéressant car il fournit des connaissances directement interprétables, ce qui est très utile dans certains domaines, par exemple la médecine ou le scoring bancaire. Parmi les méthodes d'apprentissage supervisé à partir de règles, on citera notamment les arbres de décision, pour une présentation desquels nous renvoyons à Rakotomalala (2005), et la classification associative initiée par Liu et al. (1998), qui organise la prédiction de la classe à partir des règles d'association dont le conséquent est l'une des modalités de la variable de classe, appelées règles d'association de classe. C'est à cette dernière méthode qu'est consacré notre travail.

Comme nous le verrons en section 2, pour construire les règles de classe, de nombreux algorithmes de classification associative utilisent les algorithmes d'extraction de règles d'association conçus dans le cadre de l'apprentissage non-supervisé. Ces algorithmes d'extraction de règles d'association, par exemple Apriori (Agrawal et Srikant (1994)) ou FP-Growth (Han et al. (2000)), dont le principe sera rappelé en section 2, extraient toutes les règles dont le support et la confiance dépassent des seuils préfixés. Dans une première étape, ces algorithmes extraient tous les itemsets fréquents, ceux dont le support dépasse le seuil de support, noté $minsupp$. C'est l'étape la plus coûteuse en termes de temps d'exécution, car le nombre de ces items fréquents dépend exponentiellement du nombre d'items manipulés (pour $p$ items, on a $2^p$ itemsets possibles). Pour que cette étape soit exécutable, on doit parfois augmenter le seuil de support, ce qui donne un rôle central à la condition de support et rend plus difficile l'extraction de pépites, à savoir les règles ayant une très forte confiance et un très faible support. Lors de la seconde étape, ces algorithmes subdivisent les itemsets fréquents en deux parties disjointes, pour construire toutes les règles possibles, en ne gardant que les règles dont la confiance dépasse le seuil de confiance choisi, noté $minconf$.

Cette utilisation des algorithmes non-supervisés pour déterminer les règles de classe, assortie d'un filtrage ultérieur des règles, présente différents inconvénients. D'une part, dans la première étape, on extrait des itemsets fréquents inutiles, à savoir ceux qui ne contiennent pas une modalité de classe, ce qui empêche de baisser le seuil de support autant qu'on le voudrait. D'autre part, la seconde étape peut être simplifiée, puisqu'un itemset contenant l'une des modalités de la classe (ou itemset de classe) ne peut donner lieu qu'à une seule règle de classe. Par ailleurs, l'utilisation d'un seuil de support s'appliquant sur la totalité de la base désavantage la classe minoritaire qui est généralement la plus intéressante et la plus difficile à prédire correctement. Enfin, ces différents algorithmes se fondent seulement sur le support et la confiance pour extraire les règles intéressantes, alors que différents travaux (Guillet et Hamilton (2007)) ont montré qu'on peut améliorer la qualité des règles extraites en utilisant d'autres mesures d'intérêt, plus pertinentes que la confiance et le support.

Face à ces problèmes, nous présentons FCP-Growth une adaptation de FP-Growth qui dans l'étape de construction des itemsets fréquents permet de ne générer que les itemsets fréquents de classe. En outre, pour tenir compte d'un éventuel

déséquilibre des classes, le même seuil de support est utilisé à l'intérieur de chaque classe. Enfin, nous introduisons une condition d'élagage dans FCP-Growth pour éviter d'extraire les spécialisations de règles qui n'ont pas d'intérêt, définissant ainsi l'algorithme FCP-Growth-P. Cet article présente d'abord un état de l'art des méthodes de construction de règles d'association prédictive, en s'intéressant plus particulièrement aux différents algorithmes de recherche d'itemsets fréquents et aux différentes méthodes utilisées pour l'élagage (Section 2). Puis, nous expliquons les améliorations apportées à FP-Growth (Section 3) et nous comparons les résultats de FCP-Growth-P et de FCP-Growth à ceux de FP-Growth, la méthode standard, après les avoir appliqués à sept bases de données différentes (Section 4). Nous terminons en dressant les perspectives qu'ouvre ce travail (Section 5).

# 2 Etat de l'art

Après avoir présenté les principaux algorithmes de classification associative, nous rappelons les différentes méthodes d'extraction de règles d'association en non-supervisé et les stratégies d'élagage.

## 2.1 Principaux algorithmes de classification associative

Depuis l'article fondateur de Liu et al. (1998), différents travaux ont mis en évidence les bons résultats de la classification associative en termes de diminution du taux d'erreur. La classification associative est fondée sur la prédiction de la classe à partir de règles d'association particulières, dites règles d'association de classe (*class association rules*) ou règles d'association prédictives. Une règle d'association de classe est une règle dont le conséquent doit être la variable indicatrice de l'une des modalités de la classe. Une telle règle s'écrit donc $A \rightarrow c_i$, où $A$ est une conjonction de descripteurs booléens et $c_i$ est la variable indicatrice de la $i_e$ modalité de classe. L'intérêt des règles de classe est de permettre la focalisation sur des groupes d'individus, éventuellement très petits, homogènes du point de vue des descripteurs et présentant la même classe. Par rapport aux arbres de décision, la classification associative a l'intérêt de ne pas pratiquer de stratégie gloutonne, ce qui permet d'explorer l'ensemble des règles construites.

Les méthodes de classification associative procèdent en deux phases, une phase de construction des règles d'association de classe suivie d'une phase de prédiction à partir des règles de classe obtenues. La première phase se décompose elle-même en deux étapes, l'une consacrée à l'extraction des itemsets fréquents, compte tenu du seuil de support choisi, l'autre portant sur la construction des règles d'association de classe satisfaisant le seuil de confiance préfixé.

Parmi ces méthodes, on citera les algorithmes CBA (*Classification Based on Association*, Liu et al. (1998)), qui utilise Apriori pour générer toutes les règles d'association satisfaisant aux seuils de support et de confiance choisis au départ, CPAR (*Classification based on Predictive Association Rules*, Yin et Han), qui s'inspire du principe de la méthode FOIL (*First Order Inductive Learner*, Quinlan et Cameron-Jones (1995)) pour générer les règles, évitant ainsi la répétition des calculs pour les règles redondantes et CMAR (*Classification based on Multiple Association Rules*, Li et al. (2001a)) qui utilise FP-Growth pour extraire les règles. Ces méthodes de classification associative ont donné de bons résultats. Cependant, elles se fondent sur une première phase d'extraction des règles d'association qui n'est pas en relation avec la seconde phase de classification.

## 2.2 Extraction des règles d'association

Durant la première phase de leur mise en oeuvre, celle d'extraction des règles, les algorithmes usuels de classification associative ont recours aux algorithmes d'extraction de règles d'association utilisés en apprentissage non-supervisé. Parmi ceux-ci, les plus utilisés sont Apriori et FP-Growth.

La stratégie utilisée par l'algorithme Apriori (Agrawal et Srikant (1994)) pour générer les itemsets fréquents est simple. Il détermine ceux-ci dans un ordre croissant de longueur, en se fondant sur la propriété d'antimonotonie de la condition de support. Selon cette propriété, tous les sur-itemsets d'un itemset non fréquent sont non fréquents et tous les sous-itemsets d'un itemset fréquent sont fréquents. Au niveau $k$ du treillis des itemsets, en fusionnant deux par deux tous les itemsets de longueur $k-1$, Apriori génère tous les itemsets candidats de longueur $k$ et ne retient parmi ceux-ci que ceux dont la fréquence est supérieure ou égale au seuil de support $minsupp$. Ensuite, à partir de chaque itemset fréquent $X$, Apriori examine toutes les règles du type $X - Y \rightarrow Y$, où $Y \subset X$, $Y \neq \oslash$ et ne retient que celles dont la confiance est au moins égale à $minconf$, le seuil de confiance choisi. Ces règles sont alors ajoutées à ER, l'ensemble des règles d'association qui satisfont aux seuils de support et de confiance préfixés.

A l'opposé d'Apriori qui génère des itemsets candidats et qui les teste pour ne conserver que les itemsets fréquents, FP-Growth (Han et al. (2000)) construit les itemsets fréquents sans génération de candidats. En fait, il utilise la stratégie "diviser et dominer" (*divide-and-conquer*). Tout d'abord, il compresse les itemsets fréquents représentés dans la base de données à l'aide d'un FP-Tree (*frequent-pattern tree*) dont les branches contiennent les associations possibles des items.

Chaque association peut être divisée en fragments (*pattern fragment*) qui constituent les itemsets fréquents. La méthode FP-Growth transforme le problème de la recherche de l'itemset fréquent le plus long par la recherche du plus petit et sa concaténation avec le suffixe correspondant (le dernier itemset fréquent de la branche aboutissant à l'item considéré). Ceci permet de réduire le coût de la recherche.

**Exemple explicatif de FP-Growth** : Pour expliquer le principe de FP-Growth, on choisit la base de données simplifiée décrite dans le tableau 1 qui contient des items descriptifs ($I$) et des items de classe ($C$).

| Id Transaction | Liste d'item-TID |
|---|---|
| T1 | $I1, I2, I5, C1$ |
| T2 | $I2, I4, C2$ |
| T3 | $I2, I1, I3, C1$ |
| T4 | $I1, I2, I4, C2$ |

TAB. 1: Exemple

| ITEMS | Support |
|---|---|
| $I2$ | 4 |
| $I1$ | 3 |
| $I4$ | 2 |
| $C1$ | 2 |
| $C2$ | 2 |
| $I3$ | 1 |
| $I5$ | 1 |

TAB. 2: Items associés à leur support

– Premièrement, FP-Growth parcourt la base de donnée et lit tout les items existant en calculant leur support et les compare avec le seuil de support préfixé. Ainsi, il n'accepte que les items ayant un support supérieur ou égal au seuil appelés items fréquents. On choisit ici un seuil de support absolu égal à 1. FP-Growth enregistre les items fréquents dans une liste $L$ où ils figurent par ordre décroissant de support. Dans notre exemple, la liste correspondante est : $L = I2 : 4, I1 : 3, I4 : 2, C1 : 2, C2 : 2, I3 : 1, I5 : 1$

– Deuxièmement, 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 L. Par exemple, la première transaction $T1 : I1, I2, I5, C1$ sera sauvegardée en $T1 : I2, I1, C1, I5$. Cette première transaction, une fois ordonnée, va construire la première branche du FP-Tree avec 4 noeuds $(I2 : 1), (I1 : 1), (C1 : 1)$ et $(I5 : 1)$. La seconde transaction T2, contenant T2 dans l'ordre de $L : I2, I4, C2$, construit une branche où le noeud I2 est lié à la racine, le noeud I4 lié à I2 et le noeud C2 lié à I4. Cette branche partage un préfixe commun I2, avec T1. Ainsi, lors de construction du noeud I4 :1, doit-on incrémenter de 1 le compteur du noeud I2 (I2 :2).

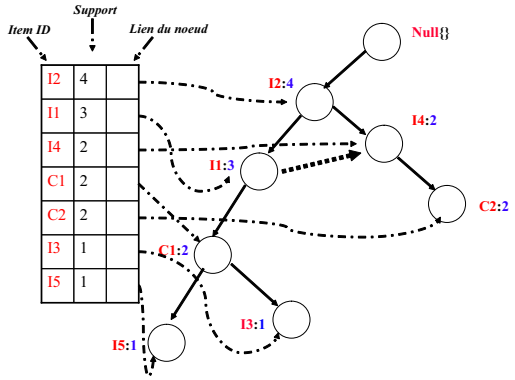


FIG. 1: Construction du FP-Tree

– 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 (*conditional pattern base*). L'itemset fréquent est obtenu par la concaténation du suffixe avec les fragments fréquents extraits des FP-Tree conditionnels (voir tableau 3)

| Item | Motifs conditionnels | FP-Tree conditionnels | Motifs fréquents |
|---|---|---|---|
| I5 | $I1, I2, C1, I2, I1, I3$ | $(I2 : 2, I1 : 1)$ | $I2, I5 : 2, I1, I5 : 2, I2, I1, I5 : 2$ |
| I4 | $I2 : 1, I2, I1 : 1$ | $(I2 : 2)$ | $I2, I4 : 2$ |

TAB. 3: Fouille du FP-Tree

Par ailleurs, il faut noter que les règles d'association de classe obtenues par filtrage des règles extraites à partir des algorithmes non supervisés se trouvent confrontées au problèmes des classes déséquilibrées. En effet, dans l'étape de génération des itemsets fréquents, ces algorithmes sélectionnent les itemsets avec un seuil de support qui ne prend pas en compte le poids des différentes modalités de la classe à prédire, ce qui désavantage mécaniquement les règles de classe associées aux modalités les moins fréquentes.

Dans notre étude, nous avons choisi de retenir FP-Growth, en raison de sa structuration (FP-Tree), qui est unanimement considéré comme plus efficace qu'Apriori. Nous proposons ainsi FCP-Growth, une variante de FP-Growth qui ne construit que les itemsets de classe, dont le principe est exposé dans (Bahri and Lallich (2009)).

### 2.3 Elagage

Lors de l'extraction des itemsets fréquents, les différents algorithmes cités se basent sur un seul principe d'élagage, celui de la condition de support. Dans la littérature, il existe plusieurs façon d'élaguer qui donnent des bons résultats. On trouve notamment :

1. La règle la plus intéressante (principe du CSA) : étant données deux règles R1 et R2, R1 est plus intéressente que R2 si $conf(R1) > conf(R2)$, ou $conf(R1) = conf(R2)$ et $sup(R1) > sup(R2)$, ou enfin $conf(R1) = conf(R2)$, $sup(R1) = sup(R2)$ et R1 possède moins d'items. Le principe d'élagage correspondant consiste à supprimer la règle la moins générale. Ce principe est notamment utilisé par CMAR.
2. La règle la plus performante : On peut citer le critère FOIL (Quinlan et Cameron-Jones (1995)), utilisé par CPAR (Yin et Han), qui se base sur la mesure "$FOIL - Prune(R) = (pos - neg) \div (pos + neg)$" où $pos$ et $neg$ sont les nombres de t-uples positifs et négatifs couverts par la règle R. Un autre critère possible est la corrélation de l'antécédent d'une règle avec la classe mesurée par le $\chi^2$. Ce critère est utilisé par CMAR (Li et al. (2001a)).
3. Comportement de la règle vis-à-vis des contre-exemples : les travaux de Li (Li (2006), Li et Zhang (2003) et Li et al. (2001b)) montrent que si la spécialisation d'une règle ne diminue pas le nombre de contre-exemples, alors ni cette spécialisation, ni aucune sur-spécialisation n'est plus intéressante au sens de la confiance ou de différentes mesures d'intérêt. C'est ce dernier critère d'élagage, utilisé par Li avec Apriori, que nous avons choisi d'introduire dans l'algorithme FCP-Growth pour obtenir FCP-Growth-P.

## 3 La méthode proposée : FCP-Growth

Afin de diminuer le coût des règles d'association prédictives en termes de temps d'exécution et de stockage, on doit se limiter à la génération des itemsets fréquents de classe, c'est-à-dire les itemsets fréquents qui contiennent l'une des modalités de classe. A cette fin, nous avons proposé FCP-Growth (*Frequent Class Pattern*) une variante de FP-Growth qui ne stocke un itemset fréquent dans le FCP-Tree (*Frequent Class pattern Tree*) que si celui-ci est un itemset de classe, du type $c_i A$. Un tel itemset ne génère qu'une seule règle de classe, la règle $A \rightarrow c_i$. Ceci nous permet un double gain, en temps d'exécution et en espace de stockage.

Simultanément, pour résoudre le problème des classes déséquilibrées, nous proposons de fixer un seuil de support ajustable, qui tient compte de l'effectif de chaque modalité de la classe à prédire. Si $n_i$ désigne l'effectif de $c_i$, la $i^e$ modalité de classe, une fois choisi le seuil $\sigma$ sur la base entière, le seuil de support adapté à la classe $c_i$ s'écrit $\sigma_i$, où $n \times \sigma_i = n_i \times \sigma$. Le seuil de support adapté est donc proportionnel à l'effectif de classe, de telle sorte que si l'on exprime ce seuil en proportion de l'effectif de classe et non plus de l'effectif de la base, celui-ci soit le même pour chaque classe. De la sorte, on évite que les classes les moins nombreuses soient systématiquement désavantagées lors de la construction des itemsets de classe et des règles d'association prédictives associées.

**Algorithme**. Notre adaptation a pour principe d'ajouter des modules à la version de base FP-Gowth.

1. Prétraitement de la base de données : Une fois la classe à prédire choisie par l'utilisateur parmi les attributs, les données sont réorganisées sous forme d'un tableau dont la première colonne contient les différentes modalités assignées à la classe. Cette méthode nous assure de construire le FCP-Tree des seuls itemsets de classe fréquents. Pour expliquer notre méthode et la comparer avec la version d'origine FP-Growth, nous gardons le même exemple. Au terme de la phase de prétraitement, la base de données sera organisée comme indiqué dans le tableau 4.

| TID | List of item-TID |
|---|---|
| T1 | $C1, I1, I2, I5$ |
| T2 | $C2, I2, I4$ |
| T3 | $C1, I2, I1, I3$ |
| T4 | $C2, I1, I2, I4$ |

TAB. 4: Base de données après le prétraitement

| ITEMS | Supp pour C1 | Supp pour C2 |
|---|---|---|
| $I2$ | 4 | 4 |
| $I1$ | 3 | 3 |
| $I4$ | 2 | 2 |
| $I3$ | - | 1 |
| $I5$ | - | 1 |

TAB. 5: Items fréquents accordés au support de classe

2. Ajustement du seuil de support : Cette étape nous permet de définir, non pas un seuil de support fixe comme dans la version de base de FP-Growth, mais un seuil de support adapté à chaque modalité $c_i$, qui dépend de $n_i$, le nombre de transactions validant la modalité. Par exemple, pour la modalité $c_i$, on aura $\sigma_i = \sigma \times n_i/n$. Durant cette phase,

au lieu d'une seule liste L qui contient les items fréquents par ordre décroissant de support comme FP-Growth, FCP-Growth construit autant de listes $L$ que de modalités de classe et attribue aux items de classe un poids $P$ plus grand que le support maximal des autres items, considéré comme support. Ainsi, chaque liste commencera par l'item de classe correspondant à son support mininum de classe. Pour notre exemple, on supposera que le support de $C1$ est 200 et C2 de 100. Si on définit $\sigma$ à $1\%$ alors on aura un support minimun de 2 pour $C1$ et de 1 pour $C2$.

Pour l'exemple, on aura deux listes, $L1$ pour la classe $C1$ et $L2$ pour la classe C2 :
– $L1 = (C1 : P1), (I2 : 4), (I3 : 3), (I4 : 2)$
– $L2 = (C2 : P2), (I2 : 4), (I3 : 3), (I4 : 2), (I3 : 1), (I5 : 1)$
Donc d'après $L1$, la première transaction sera enregistrée comme suit : $C1, I2, I1$

3. Construction du FCP-Tree et des motifs fréquents : Cette étape se déroule de la même façon que pour FP-Growth. Cependant, seules les transactions qui commencent par une modalité de la classe à prédire seront traitées. La phase de prétraitement nous a facilité cette tâche. Ainsi aura-t-on autant de fils de la racine du FCP-Tree que de modalités de la classe. On commence alors par la construction de la racine vide, puis on parcourt la base de données et à chaque fois qu'il y a une modalité de classe différente, on construit un nouveau noeud lié à la racine du FCP-Tree. Pour notre exemple, on construit deux noeud fils, $C1$ et $C2$. La première branche du FCP-Tree est construite par la première transaction $T1$. Par exemple, la première branche correspondant à $T1$ sera lié au noeud $C1$ par deux noeuds $I2 : 1$ et $I1 : 1$. Ainsi, on est sûr que pour chaque item fréquent, on a un motif conditionnel qui contient un item de classe.

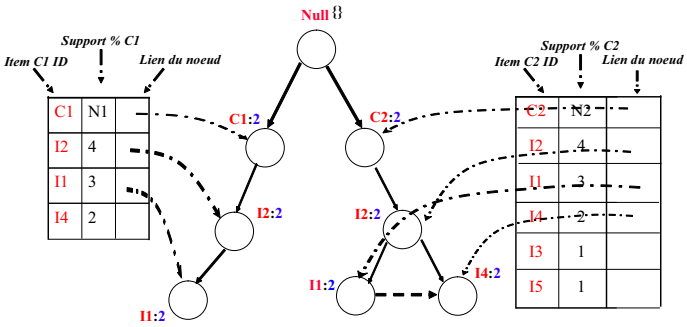


FIG. 2: Construction de FCP-tree

4. Fouille du FCP-Tree et génération des items fréquents : On procède de la même façon que FP-Growth. On extrait les motifs suffixes et on construit pour chacun ses motifs conditionnels. On extrait alors tous les itemsets de classe plus fréquents que le seuil. Les itemsets fréquents sont obtenus par l'association de chaque item avec ses motifs fréquents correspondants. Pour notre exemple, on est sûr d'éliminer l'itemset $I4I1$ trouvé dans la table 3 grâce à FCP-Growth et à la modification de construction du FCP-Tree où chaque item doit être attaché à une branche qui contient un item de classe

5. Elagage : Nous avons introduit dans FCP-Growth la méthode d'élagage précitée de Li (Li, 2006) qui repose sur le fait que toute spécialisation d'une règle qui ne diminue pas le nombre de contre-exemples est moins intéressante que la règle de départ. Cette méthode a été proposée dans le cadre d'Apriori, un algorithme d'extraction d'itemsets fréquents avec génération de candidats. Pour l'introduire dans FCP-Growth qui est un algorithme d'extraction d'itemsets fréquents sans génération de candidats, nous avons dû procéder à quelques adaptations. A chaque transaction et la construction d'un noeud (item $I$), on met à jour et on calcule le nombre de transactions qui valident l'association des items depuis le début de la branche jusqu'au noeud de l'item $supp(Ic)$ et le nombre de transactions qui contredit cette association $Supp(I\bar{c})$.

   Soit l'itemset de classe $Ac$, la règle de classe associée est $A \rightarrow c$. Les contre-exemples de cette règle correspondent à l'itemset $A\bar{c}$. Considérons l'item $x$ auquel on associe la spécialisation $Ax \rightarrow c$. La condition d'élagage est la suivante : pour une mesure $m$ optimale au sens de Li, si $supp(Ax\bar{c}) = Supp(A\bar{c})$, alors $m(Ax \rightarrow c) \leq m(A \rightarrow c)$. Pour intégrer cette contrainte dans un algorithme, il faut être capable de tenir à jour le nombre de contre-exemples de la règle de classe associée à l'itemset de classe courant.

   Dans FCP-Growth, chaque fils de la racine vide correspond à une modalité de classe. Par exemple, si on a les itemsets de classe $ca_1a_2$, puis $ca_1a_2a_3$, la condition d'élagage conduit à comparer les supports des contre-exemples de chacune des deux règles de classe $a_1a_2 \rightarrow c$ et $a_1a_2a_3 \rightarrow c$, c'est-à-dire $p_{\bar{c}a_1a_2}$, et $p_{\bar{c}a_1a_2a_3}$.

FCP-Growth tient à jour les supports $p_{ca_1a_2}$, et $p_{ca_1a_2a_3}$. Pour calculer $p_{\bar{c}a_1a_2} = p_{a_1a_2} - p_{ca_1a_2}$ et $p_{\bar{c}a_1a_2a_3} = p_{a_1a_2a_3} - p_{ca_1a_2a_3}$, il faut connaître $p_{a_1a_2}$ et $p_{a_1a_2a_3}$. De façon générale, à chaque noeud $cA$ de l'arbre il suffit de calculer non seulement $p_{cA}$ mais aussi $p_A$, ce qui permet d'en déduire $p_{\bar{c}A} = p_A - p_{cA}$. Dans l'exemple proposé, si $p_{\bar{c}a_1a_2a_3} = p_{\bar{c}a_1a_2}$, on élague ce noeud, car ni la règle spécialisée $a_1a_2a_3 \rightarrow c$ ni aucune de ses sur-spécialisations ne sont plus intéressantes que la règle $a_1a_2 \rightarrow c$ au sens de la confiance.

# 4 Expérimentation et résultats

FCP-Growth et FCP -Growth-P, les algorithmes que nous proposons pour construire directement les itemsets de classe fréquents et les règles qui en découlent, reposent sur plusieurs principes :

- FCP-Growth est une adaptation de FP-Growth, réputée plus rapide qu'Apriori.
- au cours de la construction du FCP-Tree, FCP-Growth élimine les itemsets non pertinents, ceux qui ne contiennent pas d'item de classe, alors que les méthodes usuelles construisent toutes les règles et éliminent ensuite les règles qui ne sont pas des règles de classe. Cette caractéristique devrait améliorer le temps d'exécution et permettre de diminuer le seuil de support, ce qui est particulièrement important en classification associative, où l'on est très intéressé par les pépites.
- FCP-Growth utilise un seuil de support adaptatif au sens où le seuil de support utilisé dans chaque classe n'est pas le même ; dans le but de ne pas pénaliser les petites classes, ce seuil est choisi proportionnel à la taille de la classe, ce qui revient à fixer comme seuil le même pourcentage de chaque classe.
- FCP-Growth-P résulte de l'adjontion à FCP-Growth d'une procédure d'élagage qui applique la condition de Li sur le support des contre-exemples, ce qui permet d'éliminer les spécialisations des règles qui n'ont pas d'intérêt.

Pour évaluer l'efficacité de FCP-Growth et FCP-Growth-P, nous les avons testés sur 7 bases de données réelles volumineuses (tableau 6) et nous avons calculé (tableaux 7 et 8), avec un seuil de support de 1%, le nombre total d'itemsets (# total Itemsets), le nombre d'itemsets de la classe $Ci$ (# Itemsets Ci) et le nombre d'itemsets non pertinents (# Itemsets non pertinents).

| Bases de données | #Exp | #Attr | #Classes | Poids des classes |
|---|---|---|---|---|
| D100T20I6 | 100000 | 20 | 2 | 58%-42% |
| D100T20I2 | 100000 | 20 | 2 | 55%-45% |
| Retail data | 63000 | 10 | 2 | 69%-31% |
| Adult | 48842 | 14 | 2 | 75,22%-24,78% |
| Connect-4 | 67557 | 42 | 3 | 65,83%-24,62%-9,55% |
| Abalone | 4177 | 8 | 3 | 53%-25%-22% |
| Census | 48842 | 14 | 2 | 75.22%-24.78% |

TAB. 6: Description des bases de données

La comparaison des résultats de FCP-Growth et de sa variante FCP-Growth-P, avec ceux de la version d'origine FP-Growth (tableaux 7 et 8) est menée sur la base de 4 critères : la diminution du nombre total d'itemsets construits, l'augmentation du nombre d'itemsets de classe, la bonne représentation de la classe minoritaire et l'amélioration du taux de couverture (c'est-à-dire la proportion d'exemples couverts par un itemset de classe intéressant).

Les résultats montrent, tout d'abord, l'importance du poids des itemsets non pertinents (les itemsets ne contenant pas la classe). Sur l'ensemble des sept bases traitées, il apparaît qu'entre le tiers et la moitié des itemsets générés par FP-Growth ne sont pas pertinents, ce qui alourdit inutilement la procédure. De plus, la mesure Variation1(obtenu en divisant # Itemsets FCP-Growth par # Itemsets FP-Growth) montre bien que, grâce à sa procédure d'élimination des itemsets qui ne sont pas de classe et à son seuil adaptatif, FCP-Growth ainsi que sa variante FCP-Growth-P permettent d'extraire plus de règles de classes que FP-Growth, tout en évitant que la classe minoritaire soit trop défavorisée. Sur les sept bases de données, ce gain varie entre 17% et 23%.

Pour évaluer les performances de FCP-Growth et FCP-Growth-P, nous avons retenu le taux de couverture qui indique la proportion d'exemples qui sont couverts par au moins un itemset de classe, c'est-à-dire qui vérifient la règle de classe correspondant à cet itemset. Les résultats du tableau 8 montrent très clairement la supériorité de FCP-Growth. En effet, pour chacune des bases, le taux de couverture est nettement augmenté pour arriver environ à plus de 96% de couverture. De plus, pour une meilleure qualité et grâce à sa procédure d'élagage, la variante FCP-Growth-P, héritant déjà des avantages de FCP-Growth en termes d'augmentation d'itemsets de classes, permet de réduire la quantité des règles de classe extraites en améliorant la qualité. En effet, la mesure variation2 du tableau 7 (obtenu en divisant # Itemsets FCP-Growth-P par # Itemsets FCP-Growth)montre que FCP-Growth-P diminue le nombre de règles de classe extraites de 6,67% à 11,37% suivant les bases, tout en assurant le même taux de couverture.

# 5 Conclusion et perspectives

Nous avons proposé FCP-Growth une adaptation de FP-Growth à la recherche des itemsets de classe et des règles d'association prédictives dans une perspective de classification associative, ainsi que FCP-Growth-P, sa variante avec élagage (au sens de Li). Tant FCP-Growth que FCP-Growth-P ne construisent que des itemsets fréquents de classe. En outre, le seuil de support utilisé dans chaque classe est proportionnel à la taille de chaque classe pour éviter de défavoriser les classes minoritaires. Les résultats trouvés montrent que FCP-Growth et plus encore FCP-Growth-P permettent un gain de temps grâce à la génération des seuls itemsets fréquents de classe. Ce gain peut aussi être exprimé en termes de facilité de construction et de stockage. De plus, l'examen des taux de couverture montre que nous obtenons une qualité de règles supérieure à celle des règles générées par FP-Growth. Au vu des résultats obtenus, nous allons utiliser FCP-Growth-P comme algorithme de génération de règles pour la première phase d'une procédure de classification associative performante.

# Références

Agrawal, R. et R. Srikant (1994). Fast algorithms for mining association rules. In J. B. Bocca, M. Jarke, et C. Zaniolo (Eds.), *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, pp. 487–499. Morgan Kaufmann.

Bahri, E. et S. Lallich (2009). FCP-Growth, une adaptation de FP-Growth pour générer des règles d'association de classe. *Poster, 9èmes Journées Francophones d'Extraction et Gestion des Connaissances, EGC 2009, Strasbourg*.

Guillet, F. et H. Hamilton (2007). Quality measures in data mining. *Springer-Verlag*.

Han, J., J. Pei, et Y. Yin (2000). Mining frequent patterns without candidate generation. In W. Chen, J. Naughton, et P. A. Bernstein (Eds.), *2000 ACM SIGMOD Intl. Conference on Management of Data*, pp. 1–12. ACM Press.

Li, J. (2006). On optimal rule discovery. *IEEE Transformation on Knowldge and Data Engeneering. 18*(4), 460–471.

Li, J., H. Shen, et R. Topor (2001b). Mining the smallest association rule set for predictions. *IEEE International Conference on Data Mining*, 361–368.

Li, J. et Y. Zhang (2003). Direct interesting rule generation. *IEEE International Conference on Data Mining*, 155–162.

Li, W., J. Han, et J. Pei (2001a). CMAR : Accurate and efficient classification based on multiple class-association rules. In *ICDM*, pp. 369–376.

Liu, B., W. Hsu, et Y. Ma (1998). Integrating classification and association rule mining. In *Knowledge Discovery and Data Mining*, pp. 80–86.

Quinlan, J. R. et R. M. Cameron-Jones (1995). Induction of logic programs : FOIL and related systems. *New Generation Computing 13*, 287–312.

Rakotomalala, R. (2005). Arbres de décision. *Revue MODULAD 33*, 163–187.

Yin, X. et J. Han. CPAR : Classification based on predictive association rules. In *3rd SIAM International Conference on Data Mining (SDM'03)*, San Francisco, CA, pp. 331–335.

# Summary

In this search, we focused on supervised learning task using association rules algorithms (association based classification). These algorithms, developed in unsupervised learning, extract all the rules whose the support and confidence exceed a prefixed threshold support. The extraction of class association rules, using these algorithms, have several problems, because of the rules' a posteriori filtering. In the first stage, one extracts useless frequent itemsets, those which do not contain class, whereas the second stage can be simplified, since an itemset containing the class gives place only to only one class rule. In order to be able to work with a low threshold support, we propose FCP-Growth and an alternative FCP-Growth-P, an adaptation of FP-Growth which have three main advantages. First, it eliminates the frequent itemsets not containing a class. Secondly, to make the minority class be in advantage during the construction of the class itemsets, we adapt the threshold support, in order to use the same threshold support inside each class. Thirdly, or a better pruning, FCP-Growth-P joined the pruning's condition of Li on the rules' specialization.

| Bases de données | Variation1 | FP-Growth | FCP-Growth | FCP-Growth-P | Variation2 |
|---|---|---|---|---|---|
| *Seuil du support absolu* | *1,00%* | *1,00%* | *1,00%* | *1,00%* | *1,00%* |
| ***D100T20I6*** | | | | | |
| # Itemsets | -24,77% | 1090 | 820 | 730 | -10,97% |
| # Itemsets C1 | 12,22% | 452 | 508 | 438 | -13,72% |
| # Itemsets C2 | 28,46% | 243 | 312 | 292 | -6,48% |
| # Itemsets non pertinents | - | 395 | 0 | 0 | - |
| # Total itemsets de classe | 17,98% | 695 | 820 | 730 | ,-10,97% |
| ***D100T20I2*** | | | | | |
| # Itemsets | -24,62% | 1415 | 1075 | 967 | -10,04% |
| # Itemsets C1 | 23,78% | 565 | 699 | 610 | -12,73% |
| # Itemsets C2 | 23,06% | 306 | 376 | 357 | -6,48% |
| # Itemsets non pertinents | - | 545 | 0 | 0 | - |
| # Total itemsets de classe | 23,56% | 870 | 1075 | 967 | -10,04% |
| ***Retail data*** | | | | | |
| # Itemsets | -25,77% | 330 | 245 | 224 | -8,57% |
| # Itemsets C1 | 5,13% | 121 | 127 | 117 | -8,51% |
| # Itemsets C2 | 92,16% | 40 | 118 | 107 | -8,59% |
| # Itemsets non pertinents | - | 169 | 0 | 0 | - |
| # Total itemsets de classe | 52,17% | 161 | 245 | 224 | -8,57% |
| ***Abalone*** | | | | | |
| # Itemsets | -14,19% | 665 | 562 | 518 | -7,89% |
| # Itemsets C1 | 7,25% | 209 | 224 | 206 | -12,43% |
| # Itemsets C2 | 36,50% | 144 | 196 | 180 | -5,19% |
| # Itemsets C3 | 12,89% | 126 | 142 | 132 | -4,14% |
| # No relevant itemsets | - | 176 | 0 | 0 | - |
| # Tolal class itemsets | 17,57% | 478 | 562 | 518 | -7,89% |
| ***Adult*** | | | | | |
| # Itemsets | -14,60% | 890 | 760 | 675 | -11,18% |
| # Itemsets C1 | 2,8% | 392 | 403 | 398 | -14,53% |
| # Itemsets C2 | 43,33% | 249 | 357 | 277 | -7,82% |
| # No relevant itemsets | - | 249 | 0 | 0 | - |
| # Total class itemsets | 18,56% | 641 | 760 | 675 | -11,18% |
| ***Connect-4*** | | | | | |
| # Itemsets | -11,04% | 6543 | 5820 | 5184 | -11,37% |
| # Itemsets C1 | 11,12% | 2094 | 2328 | 2160 | -15,80% |
| # Itemsets C2 | 14,15% | 1439 | 2037 | 1657 | -8,84% |
| # Itemsets C3 | 17,03% | 1243 | 1455 | 1341 | -7,82% |
| # No relevant itemsets | - | 1767 | 0 | 0 | - |
| # Total class itemsets | 21,85% | 4776 | 5820 | 5184 | -11,37% |
| ***Cencus*** | | | | | |
| # Itemsets | -15,51% | 780 | 659 | 615 | -6,67% |
| # Itemsets C1 | 10,63% | 328 | 362 | 335 | -11,67% |
| # Itemsets C2 | 52,07% | 195 | 297 | 280 | -4,55% |
| # No relevant itemsets | - | 257 | 0 | 0 | - |
| # Total class itemsets | 26% | 523 | 659 | 615 | -6,67% |

TAB. 7: Tableau récapitulatif des résultats des 7 bases de données

| Bases de données | FP-Growth | FCP-Growth et FCP-Growth-P |
|---|---|---|
| D100T20I6 | 63% | 91% |
| D100T20I2 | 75% | 93% |
| Retail Data | 67% | 89% |
| Adult | 69% | 92% |
| Connect-4 | 73% | 90% |
| Abalone | 79% | 96% |
| cencus | 76% | 92% |

TAB. 8: Taux de couverture