Sélection de variables secondaires de données multi-tables pour la classification

Nicolas Voisine*, Lou-Anne Quellet**
Marc Boulle*, Fabrice Clerot*, Anais Collin*

*Orange Innovation Lannion
prenom.nom@orange.com,
**Grenoble INP
lou-anne.quellet@grenoble-inp.pro

Résumé. Les données multi-tables sont courantes dans les organisations et leur analyse est cruciale pour des applications telles que la détection de fraudes, l'amélioration des services ou la relation client. L'utilisation de ces données nécessite une mise à plat, transformant la structure multi-tables en une table à plat, en créant des agrégats à partir des variables originales. Des outils de propositionnalisation proposent d'automatiser ce processus, mais l'augmentation de la complexité des données par leur nombre et leurs relations réduit l'efficacité de la mise à plat. Pour améliorer la qualité de la propositionnalisation, il est essentiel de développer des systèmes de prétraitement automatique qui optimisent la construction d'agrégats en se concentrant sur les variables qui contiennent le plus d'information. L'objectif de cet article est de proposer une méthode de sélection de variables secondaires et de démontrer que cette méthode permet de trier et filtrer les variables non informatives par une approche univariée. Pour finir nous montrerons sur un ensemble de bases de données académiques qu'en réduisant le nombre de variables secondaires aux seules informatives, la qualité de la classification peut s'améliorer.

1 Introduction

Les données multi-tables constituent une part significative des structures de données utilisées au sein des organisations. Leur analyse permet d'obtenir des informations essentielles pour les entreprises, telles que la détection de fraudes, l'amélioration des services ou la relation client. La modélisation de ces structures multi-tables nécessite une étape de mise à plat (propositionnalisation, Lachiche (2017)) transformant la structure multi-tables en une table de données. La mise à plat construit de nouvelles variables à partir des variables originelles et des primitives de construction. Ce cadre multi-tables offre une richesse informationnelle, mais pose également des défis supplémentaires en raison de la structure relationnelle des données et de la présence de nombreuses variables redondantes ou non informatives. Divers outils, comme featuretools, getML ou Khiops (Kanter et Veeramachaneni, 2015; GetML, 2024; Khiops, 2024), automatisent ce processus. Néanmoins, l'augmentation de la quantité et de la complexité des

données réduit l'efficacité de la mise à plat par la présence de variables non informatives. Il est donc nécessaire de développer des systèmes automatiques de sélection de variables de données multi-tables afin de construire une mise à plat qui améliore la qualité de la modélisation finale.

La sélection de variables est une étape cruciale dans le processus de construction de modèles de classification, particulièrement lorsque l'on traite des ensembles de données complexes et volumineux (CRISP-DM, 2000). Dans le cadre de l'analyse de données multi-relationnelles (MRDM, Multi-Relational Data Mining), les données sont organisées en plusieurs tables reliées par des clés. La classification dans le contexte de MRDM exige une approche qui prenne en compte les relations entre les entités de différentes tables. Traditionnellement, la sélection de variables s'est principalement concentrée sur les variables des tables principales, c'est-à-dire celles contenant les objets cibles de la classification (Guyon et Elisseeff, 2003). Toutefois, les tables secondaires (reliées à la table principale), contenant des variables que nous nommerons secondaires peuvent également fournir des informations précieuses pour améliorer la performance des modèles de classification. Par exemple, considérons un problème de classification supervisée dans le domaine du credit scoring où l'objectif est de classifier les clients en fonction de leur probabilité de défaut de paiement. La table principale pourrait contenir des informations personnelles des clients (revenu, emploi, âge, etc.), tandis que les tables secondaires pourraient contenir des données de transactions bancaires, des historiques de remboursements de crédits antérieurs, ou des informations sur les relations avec d'autres institutions financières. La sélection de variables permettrait de déterminer quelles variables, provenant par exemple des historiques de transactions ou des interactions passées avec d'autres créanciers, ajoutent une valeur prédictive significative pour l'évaluation du risque de crédit (Lessmann et al., 2015). L'utilisation efficace des variables des tables secondaires nécessite des techniques avancées de sélection de variables capables de gérer les relations multi-tables. En effet, la sélection de variables pour le MRDM ne consiste pas seulement à identifier les attributs pertinents dans une table, mais aussi à déterminer quelles relations et attributs dans les tables secondaires apportent une contribution significative à la tâche de classification (Hu et al., 2008). Une approche naïve consistant à utiliser toutes les variables disponibles peut entraîner un sur-apprentissage du modèle (overfitting), ce qui diminue la capacité de généralisation du modèle (Modi et al., 2011). Plusieurs méthodes ont été développées pour aborder ces défis, allant des techniques de filtrage et wrapper aux approches "embedded" adaptées au contexte multi-relationnel (Guyon et Elisseeff, 2003). Par exemple, les méthodes de filtrage utilisent des mesures statistiques pour évaluer l'importance des variables sans construire de modèle explicite, ce qui peut être avantageux pour réduire le coût computationnel. D'autre part, les méthodes embedding construisent et évaluent des modèles prédictifs pour chaque sous-ensemble de variables, ce qui peut offrir une meilleure performance prédictive au prix d'un coût de calcul accru.

Dans cet article, nous nous focaliserons sur les techniques de sélection de variables filtres qui exploitent les variables des tables secondaires pour la classification dans le contexte du MRDM. Nous examinerons comment ces variables secondaires peuvent être identifiées et intégrées efficacement dans le processus de modélisation, tout en minimisant le risque d'inclusion de bruit. Nous proposerons une mesure d'importance des variables secondaires permettant de filtrer les variables non informatives et nous montrerons son utilité pour détecter des variables secondaires de bruit.

2 Construction de variables dans les données multi-tables

Les approches pionnières de la mise à plat viennent de la propositionnalisation (Lachiche, 2017), Lavrač et al. (1991) avec le systeme LINUS et Krogel et Wrobel (2001) utilisée pour automatiser la construction d'agrégats à partir de données relationnelles. Ces techniques incorporent des méthodes d'optimisation communément trouvées dans les bases de données relationnelles, comme l'indexation, pour agréger et résumer les données provenant de relations non ciblées sur des entrées spécifiques de la table cible. Cependant, ces méthodes présentaient des limitations, telles que l'interdiction de la récursivité. L'approche issue du Machine Learning introduite par Deep Feature Synthesis (Kanter et Veeramachaneni, 2015) construit automatiquement des agrégats simples à partir d'ensembles de données relationnelles. Elle utilise également une méthode de sélection non supervisée des variables à l'aide de la décomposition des valeurs singulières (SVD) et classe les variables en fonction de leur corrélation à la classe cible. Cette méthode s'inscrit dans le cadre d'un effort plus large visant à automatiser l'ensemble du processus de machine learning, l'ingénierie des variables relationnelles étant un aspect clé. GetML (GetML, 2024) et Khiops (Khiops, 2024) utilisent des implémentations efficaces de la propositionnalisation employant des opérations simples basées sur les agrégations SQL, qui transforment une structure de données relationnelle en une table à plat. La règle SQL (figure 1) permet de construire une variable $feature_1$ comme une fonction agrégation de la variable var_k sur une sélection de la table secondaire ts_1, la sélection étant le produit d'une condition sur l'ensemble des variables de la table ts_1.

GetML permet la génération non supervisée très rapide d'un nombre substantiel d'agrégats basés sur des agrégations simples de données continues, catégorielles ou temporelles. Khiops

```
SELECT AGG(ts1.vark) AS feature1
FROM table ts1
WHERE CONDITION({ts1.vari})
```

FIG. 1 – *Règle SQL de création d'une variable à partir d'une table secondaire.*

ajoute une analyse des distributions de chaque variable des tables secondaires pour construire des primitives de sélection prenant en compte les données. Dans cet article nous utiliserons Khiops pour construire les variables multi-tables et évaluer les variables créées. Khiops est un outil open source d'apprentissage automatique conçu pour la fouille de grandes bases de données multi-tables. Khiops repose sur une approche bayésienne MODL, ayant démontré son intérêt académique pour la construction d'agrégats, le sélection de variables et la classification.

2.1 Construction de variables

Pour Khiops, une primitive de construction d'un agrégat est similaire à une fonction dans un langage de programmation. Elle est désignée par son nom, la liste de ses opérandes et sa valeur de retour. Les opérandes et la valeur de retour sont typés. Les types standard, numérique ou catégoriel, peuvent être étendus à d'autres types spécialisés, tels que la date ou l'heure. Les opérandes peuvent être une variable d'une table secondaire, la sortie d'une autre primitive, c'est-à-dire un autre agrégat construit, ou une constante provenant des données d'ap-

prentissage. Les primitives de construction utilisées par Khiops dans les expériences sont les suivantes :

- *Selection(Table, selection criterion)→Table* : sélection des enregistrements de la table en fonction d'une conjonction de termes de sélection (appartenance à un intervalle numérique ou à un groupe de valeurs catégorielles),
- *Count(Table)→Num* : nombre d'enregistrements dans une table,
- *Mode(Table, Cat)→Cat* : valeur la plus fréquente d'une variable dans une table,
- *CountDistinct(Table, Cat)→Num* : nombre de valeurs distinctes,
- *Mean/median/StdDev(Table, Num)→Num* : valeur moyenne, médiane ou écart type,
- *Min/Max(Table, Num)→Num* : valeur minimale/maximale,
- *Sum(Table, Num)→Num* : somme des valeurs.

En utilisant la structure de données présentée dans la Figure 2 et les primitives de construction précédentes (plus la primitive *YearDay* pour les variables de date), on peut construire les agrégats suivants pour enrichir la description d'un client :

- *MainProduct = Mode(Usages, Produit)*,
- *LastUsageYearDay = Max(Usages, YearDay(useDate))*,
- *NbUsageProd1FirstQuarter = Count(Selection(Usages, YearDay(useDate) ∈ [1 ;90] and Product = "Prod1"))*.

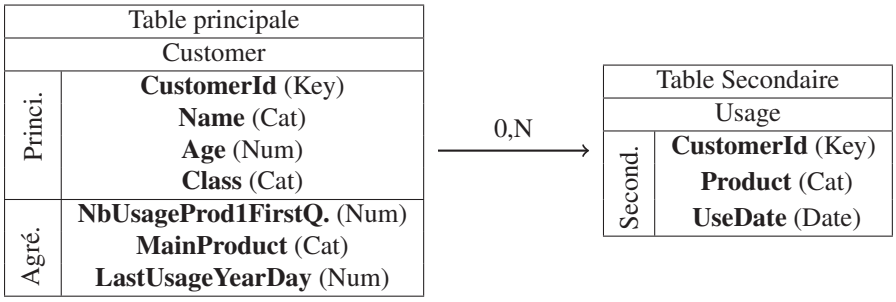

FIG. 2 – *Diagramme de la base de données relationnelle avec une table secondaire Usage avec ses variables secondaires, une table primaire Customer avec ses variables principales et ses agrégats. la clé de jointure est CustomerId.*

2.2 Évaluation de variables

Les agrégats construits peuvent être plus ou moins complexes, c'est-à-dire, construits avec une ou plusieurs variables secondaires et une ou plusieurs primitives de construction. La construction d'agrégats complexes peut amener à un phénomène de sur-apprentissage, si un agrégat est trop spécifique aux données d'entraînement il ne pourra être généralisable au jeu de données de test. Khiops, implémente l'approche MODL pour pallier le risque de construction d'agrégats trop spécifiques en incluant une régularisation des agrégats (Boullé et al., 2019). Le coût de construction de l'agrégat est considéré comme un critère de sélection de modèle. Le modèle M est divisé en deux parties, M_a correspondant à l'agrégat construit et M_e au modèle

de préparation associé.

$$cout(M) = L(M_a) + L(M_e) + L(D|M_a, M_e)$$

Le terme $L(M_e)$, est le coût soit de discrétisation pour des agrégats continus, soit de groupement pour des agrégats catégoriels. Le terme $L(M_a)$ est le coût de construction. Un agrégat complexe aura un coût de construction important, ce qui pénalisera le critère de sélection. Le terme de vraisemblance $L(D|M_a, M_e)$ mesurant la pertinence d'un agrégat sur la variable cible devra alors compenser le coût de construction pour que l'agrégat construit soit sélectionné par le modèle. Pour donner un exemple, un agrégat construit à partir d'une unique variable et d'une seule primitive a un coût de construction plus faible qu'un agrégat construit à partir de deux variables secondaires et de deux primitives. Le premier agrégat sera alors prioritairement pris en compte si les deux agrégats possèdent la même pertinence. Khiops construit alors, en définissant un coût nul, $cout(M_\emptyset)$ une métrique normalisée, le Level.

$$Level(M) = 1 - \frac{cout(M)}{cout(M_\emptyset)}$$

Le Level mesure l'importance d'un agrégat vis-à-vis de la cible et mesure sa corrélation à la cible. Les valeurs du Level sont comprises entre 0 et 1. Une valeur de 0 signifie que l'agrégat construit n'est pas corrélé à la cible et une valeur de 1 correspond à un agrégat totalement corrélé à la cible. Le cout du modèle nul $cout(M_\emptyset)$ correspond au coût d'un modèle avec un seul intervalle ou un seul groupe de valeurs. Ainsi, si un modèle de discrétisation a un coût supérieur au modèle nul il est moins probable que le modèle nul. Il n'y a donc pas d'information dans la variable ou agrégat. Le Level Khiops est une mesure sur laquelle s'appuieront les études réalisées.

3 Étude de l'influence des variables de bruit sur les performances

3.1 Objectif

Dans le cas de données à plat, l'ajout d'une variable de bruit n'apporte pas d'information pour la prédiction de la variable cible. Par contre dans le cas de données multi-tables, il y a peu d'études sur l'ajout de variables de bruit dans les tables secondaires. Elles pourraient engendrer une baisse des performances due à la construction d'une multitude d'agrégats bruités ou au contraire produire des projections aléatoires informatives. Dans cette étude, nous cherchons à répondre à deux questions : (1) Est-ce que les variables créées à partir de variables de bruit dans les tables secondaires peuvent apporter de l'information utile à un classifieur supervisé ? et (2) Est-ce que la présence de ces variables de bruit dans les tables secondaires impacte négativement les performances du classifieur ? Nous utilisons l'algorithme de classification Khiops et mesurons l'AUC (aire de la courbe ROC) dans différents scénarios. Khiops utilise un classifieur Bayésien naïf avec sélection de variables et apprentissage direct des poids par variable. Quatre jeux de données synthétiques seront utilisés, chacun avec une table secondaire présentant une distribution du nombre d'instances corrélée avec la variable cible (entre 80 et 100 instances pour la classe 1 et entre 70 et 90 pour la classe 0), donc le nombre d'instances

dans chaque table dépend de la classe cible. Les tables secondaires seront les suivantes : (i) une table sans bruit, (ii) une table avec une seule variable de bruit ajoutée, (iii) une table avec 10 variables de bruit ajoutées, et (iv) une table composée uniquement de 10 variables de bruit. Toutes les tables, excepté (iv), disposent de 10 variables catégorielles corrélées à la cible VAR_i. Plus i est grand plus les valeurs dans la table secondaire seront corrélées à la cible. On peut construire avec VAR_9 un agrégat parfaitement corrélé avec la cible.

Dans cet article les bruits seront soit catégoriels soit numériques. Les bruits de type numérique (N_i) sont des variables numériques de type gaussien. Les bruits de type catégoriel (C_i) sont des variables dont les valeurs sont construites à partir d'un dictionnaire de labels tirés uniformément. La taille du dictionnaire est égal à 2^{i+1} pour i allant de 0 à 5.
À partir de ces tables, nous générerons 100, 1 000, 10 000 et 100 000 variables à inclure dans le modèle.

3.2 Résultats

La figure 3 montre que la courbe rouge, correspondant à la table constituée uniquement de bruit, présente une AUC supérieure à 0,75. Cela prouve que, lorsque la distribution du nombre des instances est corrélée avec la cible, même des agrégats construits avec des variables de bruit peuvent apporter de l'information. Donc l'information portée par une table secondaire ne réside pas seulement dans les variables secondaires mais aussi dans la distribution des instances secondaires. Cependant, on observe également que les performances des courbes orange et verte, qui incluent du bruit, sont inférieures à celles obtenues sans bruit. De plus on constate que plus le nombre de variables de bruit est important plus les performances baissent. Cela indique que la présence de bruit diminue les performances du classifieur en construisant une multitude d'agrégats bruités. Il est donc crucial de détecter et d'éliminer les variables non informatives issues des tables secondaires, même lorsque la distribution des instances est corrélée à la cible. Enfin, on observe dans certains cas de création d'un très grand nombre d'agrégats que l'ajout du bruit réduit très peu les performances. Dans ce cas, le classifieur compense l'effet de l'ajout du bruit.

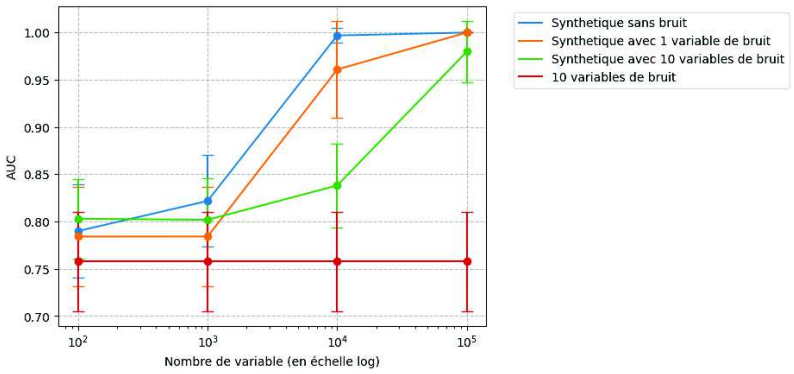

FIG. 3 – *Évolution de l'AUC en fonction du nombre de variables construites pour les données synthétiques sans bruit, avec 1 variable de bruit, avec 10 variables de bruit et uniquement composées de 10 variables de bruit*

4 La méthode de sélection proposée

Pour répondre à la problématique des variables non informatives dans les tables secondaires nous proposons une approche univariée, c'est-à-dire une mesure où chaque variable secondaire est étudiée indépendamment des autres. L'approche univariée pour la création d'une mesure d'importance permet de mesurer le potentiel d'une variable seule sur le problème de classification. La mesure d'importance par analyse univariée s'effectue par la création d'agrégats univariés et par l'analyse du level Khiops (section 2.2). Les agrégats univariés sont des agrégats construits à partir d'une unique variable secondaire et de primitives de construction. La capacité maximale d'apport d'information d'une variable sur la cible est considérée comme le level maximum obtenu sur l'ensemble des agrégats créés par Khiops.

L'estimation d'une variable s'effectue en deux étapes détaillées ci-dessous :

— Étape 1 : création de k agrégats univariés sur la variable à estimer ; avec k à définir.
— Étape 2 : extraction du level Khiops maximal dans l'ensemble des agrégats construits. Ce level maximal est considéré comme la mesure d'importance de la variable.

L'importance de la variable i se définit par :

$$Importance(Var_i) = Max(level(agregat_1), ..., level(agregat_k))$$

Les deux étapes sont répétées pour analyser l'ensemble des variables secondaires. Pour donner un exemple du fonctionnement de cette approche, soit une table secondaire contenant deux variables Var_1 et Var_2 sur lesquelles sont construits k = 3 agrégats. Le processus d'estimation est schématisé à la figure 4.

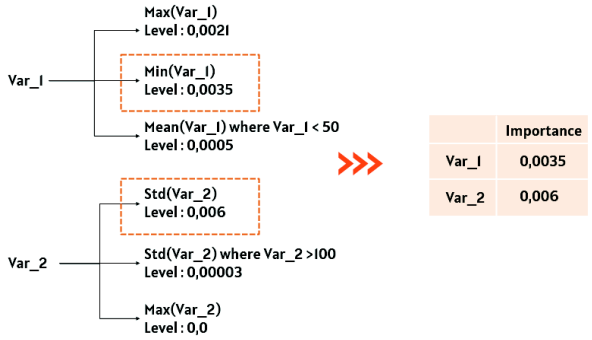

FIG. 4 – *Schéma de l'estimation par approche univariée*

La variable Var_1 génère donc trois agrégats univariés et chacun est associé à un level. L'agrégat possédant le level maximal est $Min(Var_1)$. L'importance de la variable Var_1 est donc prise égale à ce level, soit $0,0035$. De même pour la variable Var_2, l'agrégat avec le level maximal est $Std(Var\_2)$ et donc l'importance de la variable Var_2 est estimée à 0,006.

Pour pallier l'apport possible d'information par des variables de bruit et avoir une mesure de l'apport univarié non dépendant de la distribution des instances on applique des techniques de stratification (Martens, 2007) pour réduire voire éliminer l'influence du facteur de confusion (confounder) c'est-à-dire le nombre d'instances de la table secondaire. Notre méthode consiste alors à :

— Construire une stratification de la table principale conditionnée par le nombre d'instances dans la table secondaire. La solution utilisée est d'appliquer la discrétisation optimale de Khiops sur l'agrégat $Count(Table)$ qui va donner les intervalles du nombre d'instances où le comportement de la cible est uniforme.
— Evaluer pour chaque intervalle D_c la mesure d'importance $Importance_{D_c}(Var)$ où la distribution des individus n'a donc plus d'influence sur la modélisation.
— L'importance finale $LevelMT(var_i)$ d'une variable var_i est donc prise comme étant le maximum de ses estimations.
$LevelMT(var_i) = Max(Importance_{D_1}(Var_i), ..., Importance_{D_C}(Var_i))$

Où $Importance_{D_c}(Var_i)$ est l'importance de la variable var_i pour un sous-ensemble des données dont le nombre d'instances de la table secondaire est inclus dans un intervalle c. Les intervalles sont choisis de façon supervisée en optimisant la discrétisation du nombre d'instances des tables secondaires.

Les valeurs de nos deux mesures [1] sont comprises entre 0 et 1. Une valeur de 0 signifie que la variable secondaire n'a aucun agrégat construit corrélé à la cible et au contraire, une valeur de 1 montre que la variable secondaire a au moins un agrégat totalement corrélé à la cible.

4.1 Apport à la sélection de variables multi-tables

Nous illustrons les apports de notre méthode en montrant sur les données synthétiques la différence entre l'approche univariée initiale ($Importance$) et l'approche univariée finale ($LevelMT$). On constate tout d'abord dans le tableau 1 que $LevelMT$ estime les variables de bruit N_i et C_i à 0 contrairement à $Importance$. On constate aussi que les 2 méthodes rangent les variables faiblement corrélées à la cible (VAR_0 à VAR_4) avec le bruit. Pour conclure, la mesure $LevelMT$ permet une estimation des variables non informatives proches de 0 et donc proches d'une mesure d'estimation de non-information. La détection des variables de bruit ou non informatives semble donc plus fiable. Toutefois, le paramètre k est à déterminer, car une faible valeur de ce dernier peut entraîner une estimation faussée. Par la suite nous utiliserons $k = 50$. Son choix a été établi suite à des expériences de détection de variables de bruit.

4.2 Apport à la classification

Dans cette expérience, nous cherchons à évaluer l'apport d'une mesure d'importance des variables secondaires pour la classification supervisée avec des données multi-tables. L'objectif est de comparer l'efficacité d'un filtrage de variables secondaires sur la performance du classifieur, en utilisant trois types de méthodes de classification, évaluées sur 11 jeux de données, à la fois réelles et synthétiques. Les méthodes comparées sont : (1) une classification sans filtrage de variables, utilisant uniquement les données d'origine, (2) une classification sans filtrage de variables, mais incluant des variables de bruit ajoutées aux données d'origine, et (3) une classification avec filtrage de variables ($LevelMT = 0$), incluant également des variables de bruit ajoutées aux données d'origine. Les bases de données académiques contrairement aux problèmes réels incluent peu de variables de bruit ou non informatives. Pour simuler des cas réels,

1. Notre implémentation est fournie sur `https://github.com/UData-Orange/feature_selection_multi_table`

Variable	Importance estimée		Variable	Importance estimée	
	$Importance$	$LevelMT$		$Importance)$	$LevelMT$
N_0	0.0943	0	C_2	0.0951	0
N_1	0.0943	0	C_3	0.0951	0
N_2	0.0943	0	C_4	0.0951	0
VAR_0	0.0951	0	N_3	0.1001	0
VAR_1	0.0951	0	N_4	0.1014	0
VAR_2	0.0951	0	VAR_5	0.1052	0.0668
VAR_3	0.0951	0	VAR_6	0.1297	0.04844
VAR_4	0.0951	0	VAR_7	0.2382	0.2216
C_0	0.0951	0	VAR_8	0.3416	0.3149
C_1	0.0951	0	VAR_9	0.7552	0.6348

TAB. 1 – *Importance des variables des données synthétiques estimée par analyse univariée pour* $k = 50$ *avec et sans réduction de l'effet du Count.*

nous avons ajouté sur chaque table secondaire 5 variables de bruit numérique et 5 variables de bruit catégoriel. Pour chaque méthode et chaque jeu de données, nous évaluerons l'AUC (aire de la courbe ROC) en faisant varier le nombre de variables créées, en testant des configurations avec 100, 1 000 et 10 000 variables. Cette approche permettra de quantifier l'impact du bruit et de mesurer l'utilité du filtrage de variables pour améliorer les performances du classifieur dans un contexte de données multi-tables.

Data	Instances	enregistrements	Cat.cols	Num.cols	Classes	Maj.
Accident	57,783	146,949	28	6	2	0.945
Auslan	2,565	146,949	1	23	96	0.011
Credit Scoring	1,526,659	241,938,537	194	391	2	0.9686
Fox	2,565	146,949	1	23	96	0.011
Medical data	1240	3205	41	4	22	0.158
Musk1	92	476	1	166	2	0.511
Musk2	102	6,598	1	166	2	0.618
Synthétique 1	20,000	2,420,221	10	0	2	0.99
Synthétique 2	20,000	2,055,870	10	0	2	0.99
SpliceJunction	3,178	191,400	2	1	3	0.521
20newsgroups	18,846	2,435,219	1	0	20	0.054

TAB. 2 – *Ensembles de données multi-table : nombre d'instances, d'enregistrements dans les tables secondaires, de colonnes catégorielles et numériques, de classes et précision de la classe majoritaire.*

Onze ensembles de données relationnelles sont pris en compte dans ces expériences. Ils ont été choisis parce qu'ils contiennent un mélange de colonnes numériques et catégorielles et sont susceptibles de nécessiter des agrégats complexes. Ces ensembles de données séquentielles ou temporelles sont représentés par une table principale et une ou plusieurs tables secondaires en relation zéro à plusieurs. Les ensembles de données Auslan, SpliceJunction et Twenty Newsgroups proviennent du dépôt UCI (Bache et Lichman, 2013) et sont liés à la reconnaissance

de la langue des signes australienne, des caractères issus des trajectoires des pointes de stylo et des limites entre intron et exon dans les séquences de gènes (ADN). Twenty Newsgroups est une table de textes de 20 groupes de discussion sur laquelle on a construit une table secondaire faite des relations textes-mots. Les jeux de données Musk1 Musk2 (De Raedt, 1998) et les données médicales sont liés à la chimie moléculaire. Credit Scoring (Herman et al., 2024) vient d'une compétition Kaggle. Le tableau 2 donne les principales caractéristiques de ces ensembles de données : le nombre d'instances dans la table principale, le nombre total d'enregistrements dans les tables secondaires, le nombre de variables catégorielles et numériques, le nombre de classes et la proportion de la classe majoritaire. Les données Synthétique 2 sont identiques à celles (iii) utilisées en section 3.1 alors que Synthétique 1 est proche de (iii) mais sans corrélation de la distribution du nombre d'instances.

Le tableau 3 présente les AUC des expériences réalisées. On remarque tout d'abord que toutes les expériences avec bruit et sans filtre donnent des résultats inférieurs à celles sans bruit ou avec filtre. Cela confirme que l'ajout de bruit diminue les performances du classifieur, et souligne ainsi l'intérêt de détecter et de filtrer les variables secondaires de bruit ou non informatives dans un processus de propositionnalisation. De plus, on constate que le filtrage permet non seulement de maintenir les performances des classifieurs, mais aussi, dans de nombreux cas, de les améliorer (cf. Accident, Credit Scoring). En revanche, pour la base SpliceJunction, le filtrage dégrade considérablement les performances. Cela est dû à l'élimination d'une des deux variables informatives, à savoir le rang de l'acide aminé. Le caractère univarié de la méthode de sélection a filtré le motif multi-variables : le rang de base nucléique dans le brin d'ADN, combiné à son nom, apporte en effet une information précieuse. Enfin, on observe dans certains cas que la création d'un grand nombre d'agrégats compense l'effet de l'ajout du bruit. Dans ce cas le classifieur filtre lui-même le parasitage du bruit.

5 Conclusions

Cet article a pour cadre l'amélioration de la création automatique d'agrégats à partir des données multi-tables. La mise à plat transforme les données multi-tables en une table à plat. Tout d'abord nous avons montré que les agrégats créés à partir de variables de bruit dans les tables secondaires peuvent apporter de l'information dans le cas où le nombre d'enregistrements secondaires est corrélé avec la classe cible, mais que dans le cas général, la présence de ces variables de bruit dans les tables secondaires impacte négativement les performances du classifieur. Ensuite on a proposé une méthode de sélection de variables secondaires permettant de trier les variables les plus corrélées à la cible et celles qui apportent peu d'information. L'évaluation de notre méthode a montré que nous détectons parfaitement les variables de bruit. De plus on a montré qu'une réduction de l'espace des variables non informatives peut engendrer une amélioration de la qualité de la modélisation. Notre approche permet d'estimer l'importance d'une variable secondaire et de réduire l'espace de recherche pour la mise à plat en affectant faiblement la qualité de la modélisation. Toutefois, notre approche ne tient pas compte des primitives utilisées. En perspective, l'amélioration du système pourra être effectuée par l'ajout de la sélection des primitives de construction. Le système pourra ensuite être testé sur d'autres grands jeux de données académiques. Par la suite, il pourrait être intéressant de comparer le système d'exploration globale avec l'utilisation d'un second outil de mise à plat GetML et de divers algorithmes de classification supervisée.

Dataset	Nombre d'agrégats	AUC sans bruit	AUC avec bruit	AUC aprés filtrage
Accident (étoile)	100	0.8291 ± 0.0094	0.8090 ± 0.0109	$\mathbf{0.8322 \pm 0.0088}$
	1000	0.8444 ± 0.0072	0.8345 ± 0.0083	$\mathbf{0.8465 \pm 0.0069}$
	10000	0.8513 ± 0.0073	0.8468 ± 0.0072	$\mathbf{0.8536 \pm 0.0073}$
Auslan	100	$\mathbf{0.9996 \pm 0.0004}$	0.9988 ± 0.0006	0.9995 ± 0.0005
	1000	$\mathbf{0.9997 \pm 0.0004}$	0.9996 ± 0.0006	0.9996 ± 0.0007
	10000	$\mathbf{0.9999 \pm 0.0002}$	0.9997 ± 0.0006	0.9998 ± 0.0004
Credit Scoring	100	0.67 ± 0.0022	0.6775 ± 0.0025	$\mathbf{0.6923 \pm 0.0032}$
	1000	0.8215 ± 0.002	0.8026 ± 0.0021	$\mathbf{0.8261 \pm 0.0021}$
	10000	0.8325 ± 0.0018	0.8304 ± 0.0022	$\mathbf{0.8361 \pm 0.0019}$
Fox	100	0.5 ± 0.0	0.5 ± 0.0	$\mathbf{0.6470 \pm 0.0895}$
	1000	0.4998 ± 0.0324	0.5068 ± 0.0216	$\mathbf{0.6688 \pm 0.0857}$
	10000	0.5110 ± 0.0478	0.5068 ± 0.0216	$\mathbf{0.6688 \pm 0.0857}$
MedicalData	100	0.5377 ± 0.0355	0.5017 ± 0.0055	$\mathbf{0.546 \pm 0.0351}$
	1000	0.5377 ± 0.0355	0.5377 ± 0.0355	$\mathbf{0.546 \pm 0.0351}$
	10000	0.5377 ± 0.0355	0.5377 ± 0.0355	$\mathbf{0.546 \pm 0.0351}$
Musk1	100	$\mathbf{0.5 \pm 0}$	0.4972 ± 0.018	0.4972 ± 0.018
	1000	$\mathbf{0.5197 \pm 0.1451}$	0.5142 ± 0.1538	0.5142 ± 0.1538
	10000	0.5031 ± 0.1316	$\mathbf{0.5142 \pm 0.1538}$	$\mathbf{0.5142 \pm 0.1538}$
Musk2	100	$\mathbf{0.5286 \pm 0.1788}$	0.5286 ± 0.1788	0.5163 ± 0.2066
	1000	0.6601 ± 0.2241	0.6773 ± 0.1953	$\mathbf{0.7203 \pm 0.2128}$
	10000	0.6601 ± 0.2241	0.6773 ± 0.1953	$\mathbf{0.7646 \pm 0.1371}$
synthétique 1	100	0.5 ± 0.0	0.5 ± 0.0	0.5 ± 0.0
	1000	$\mathbf{0.5408 \pm 0.0365}$	0.5 ± 0.0	0.5 ± 0.0
	10000	0.9984 ± 0.0025	0.5408 ± 0.0365	$\mathbf{0.9988 \pm 0.0035}$
synthétique 2	100	0.7899 ± 0.0492	$\mathbf{0.8030 \pm 0.0421}$	0.7863 ± 0.0516
	1000	0.8218 ± 0.0486	0.8018 ± 0.0438	$\mathbf{0.8914 \pm 0.0613}$
	10000	0.9969 ± 0.0078	0.8382 ± 0.0446	$\mathbf{1.0 \pm 0.0}$
SpliceJunction	100	$\mathbf{0.8145 \pm 0.0245}$	0.6675 ± 0.0256	0.6838 ± 0.0206
	1000	$\mathbf{0.9926 \pm 0.0034}$	0.7389 ± 0.0294	0.6838 ± 0.0206
	10000	$\mathbf{0.9939 \pm 0.0039}$	0.9198 ± 0.0159	0.6838 ± 0.0206
20newsgroups	100	$\mathbf{0.7268 \pm 0.0069}$	0.6716 ± 0.0110	$\mathbf{0.7268 \pm 0.0069}$
	1000	$\mathbf{0.7866 \pm 0.0075}$	0.7080 ± 0.0079	$\mathbf{0.7866 \pm 0.0075}$
	10000	$\mathbf{0.9514 \pm 0.003}$	0.7455 ± 0.0086	$\mathbf{0.9514 \pm 0.003}$

TAB. 3 – *Mesure de l'AUC pour de multiples bases de données sans bruit, avec 10 variables de bruit et avec filtrage des variables non informatives. On utilise une validation croisée à 10 sous-échantillons*

Références

Bache, K. et M. Lichman (2013). UCI machine learning repository.

Boullé, M., C. Charnay, et N. Lachiche (2019). A scalable robust and automatic propositionalization approach for bayesian classification of large mixed numerical and categorical data. *Machine Learning 108*, 229–266.

CRISP-DM, C. P. (2000). 1.0 : Step-by-step data mining guide. *SPSS Inc*.

De Raedt, L. (1998). Attribute-Value Learning Versus Inductive Logic Programming : The Missing Links (Extended Abstract). In D. Page (Ed.), *Proceedings of the 8th International*

Workshop on Inductive Logic Programming, ILP '98, pp. 1–8. Springer-Verlag.

GetML (2024). Getml documentation. `https://docs.getml.com/latest/`.

Guyon, I. et A. Elisseeff (2003). An introduction to variable and feature selection. *Journal of machine learning research 3*(Mar), 1157–1182.

Herman, D., T. Jelinek, W. Reade, M. Demkin, et A. Howard (2024). Home credit - credit risk model stability.

Hu, B., H. Liu, J. He, et X. Du (2008). Fars : A multi-relational feature and relation selection approach for efficient classification. In *Advanced Data Mining and Applications : 4th International Conference, ADMA 2008, Chengdu, China*. Springer.

Kanter, J. M. et K. Veeramachaneni (2015). Deep feature synthesis : Towards automating data science endeavors. In *2015 IEEE international conference on data science and advanced analytics (DSAA)*, pp. 1–10. IEEE.

Khiops (2024). Khiops documentation. `https://khiops.org/learn/understand/`.

Krogel, M.-A. et S. Wrobel (2001). Transformation-based learning using multirelational aggregation. In *Inductive Logic Programming : 11th International Conference, ILP 2001 Strasbourg, France, September 9–11, 2001 Proceedings 11*, pp. 142–155. Springer.

Lachiche, N. (2017). Propositionalization. *Encyclopedia of Machine Learning and Data Mining*, 1025–1031.

Lavrač, N., S. Džeroski, et M. Grobelnik (1991). *Learning nonrecursive definitions of relations with LINUS*. Springer.

Lessmann, S., B. Baesens, H.-V. Seow, et L. C. Thomas (2015). Benchmarking state-of-the-art classification algorithms for credit scoring : An update of research. *European Journal of Operational Research 247*(1), 124–136.

Martens, E. P. (2007). *Methods to adjust for confounding : propensity scores and instrumental variables*. Utrecht University.

Modi, S., A. Thakkar, et A. Ganatra (2011). A survey on approaches of multirelational classification based on relational database. *IJEAT 1*, 77–81.

Summary

This article discusses the significance of multi-table data analysis in organizations for applications like fraud detection, service improvement, and customer relations. To utilize this data, it must be flattened into a single table by creating new variables from the original ones. While propositionalization tools can automate this process, the complexity of the data can hinder efficiency. The aim of this paper is to propose a secondary feature selection method and to demonstrate that this method can sort and filter out uninformative variables using a univariate approach. Finally, we will show on a set of academic databases that by reducing the number of secondary variables to only those that are informative, the quality of the classification can be improved.