Journées de méthodologie statistique de l'Insee 2022

Quelques limites de l'algorithme implémenté dans l'outil Sicore

Théo LEROY (*)

(*) Insee, Direction de la méthodologie et de la coordination statistique et internationale

theo.leroy@insee.fr

Mots-clés. : traitement du langage naturel, arbres de décision, moteur de règles, entropie

Domaines. Codification automatique

Résumé

Sicore (Système Informatique de Codage de Réponse aux Enquêtes) est, depuis 1993, le principal logiciel de codification automatique de l'Insee. Il permet de coder des libellés dans des nomenclatures. Son utilisation s'est imposée sur une large variété de données : professions, activités d'entreprise, diplômes, communes, nationalités, produits... Sicore intervient dans de nombreuses productions de l'institut (le recensement de la population, les enquêtes ménages, le répertoire Sirene...). En dépit, de ces applications, les choix méthodologiques adhérents à cet outil ne sont pas toujours évidents. Sans remettre en cause le rôle précieux de Sicore, cet article discute de deux potentielles limites autour de la méthodologie mise en oeuvre.

La codification avec Sicore se déroule en 3 parties. Une première étape consiste à normaliser le libellé à coder pour simplifier sa manipulation par la suite. Des caractères sont alors supprimés et des règles de synonymisation sont appliquées. Une seconde phase vise à retrouver le libellé normalisé au sein d'un index de libellé de référence selon un arbre de décision questionnant des groupes de caractères consécutifs appelés « atomes » qui composent le libellé. Enfin, si le libellé est retrouvé dans l'index et si le libellé à lui seul n'est pas suffisant pour coder dans la nomenclature, des variables annexes catégorielles sont prises en compte au travers de règles logiques. Sicore est un système expert c'est à dire que tous les composants digérés par l'algorithme (règles de normalisation, index de références, règles logiques) doivent être spécifiés en intégralité par un ou plusieurs experts de la nomenclature.

La première partie de l'article interroge la pertinence des arbres de décision Sicore pour rechercher un libellé dans un index. Ce type de classifieur malgré sa grande simplicité ne va pas nécessairement de soi pour répondre à un besoin codification dans une nomenclature car il est souvent associé à des risques de sur-apprentissage ou d'instabilité. Les arbres de décision Sicore sont construits à partir de l'index sur des critères d'entropie locaux. Ainsi, chaque noeud de l'arbre est créé de telle sorte que le désordre dans les codes de la nomenclature possibles depuis

ce noeud soit minimal. Ce choix d'optimum conduit à des arbres assez peu profonds où, parfois, seuls quelques atomes interviennent dans la codification. Afin d'augmenter la fiabilité (pour que la décision d'attribuer un code ne soit pas fondée sur un nombre trop réduit de caractères), l'expert de la codification peut spécifier des atomes additionnels (dits de « redondance ») où la conformité devra être aussi vérifiée. Ce phénomène est très fréquemment employé en pratique mais tend à rendre l'utilisation d'arbre pour l'identification du libellé dans un index inutile en ajoutant des contrôles. Il s'agira d'évaluer pour différents types de libellés (communes, professions, activités...) à quel point l'arbre de décision avec prise en compte des atomes de redondance est plus efficace c'est à dire qu'il permet de coder plus de libellé qu'une méthode naïve d'appariement strict avec l'index.

Dans une deuxième partie, il s'agira de mettre en évidence des limites de l'approche système expert pour la codification automatique de libellés issus d'enquêtes ou de sources administratives. Le bon fonctionnement de Sicore repose sur la capacité à extraire les connaissances d'experts et à les formaliser sous forme de règles. Ce type de solution peut entraîner des problèmes de maintenance ou de gestion du volume. Au fil du temps, malgré les compétences et la volonté des experts, il devient difficile d'ajouter des nouvelles règles sans engendrer des effets de bord car le moteur de règles devient trop complexe. Par ailleurs, il est impossible pour l'expert d'intégrer toute la reprise manuelle sous forme de nouvelles règles. Le but sera de donner des ordres de grandeur de l'évolution du nombre de règles et du nombre de libellé dans l'index de référence d'une version à l'autre d'une base de connaissances et d'essayer d'évaluer la répercussion de ces mises à jour sur l'efficacité de la codification automatique.

Abstract

At Insee, an automatic coding tool called Sicore exists for more than 25 years. This software is a general system based on rules and is used for a lot of variables such as occupations, towns or countries. However, this method has some drawbacks. This article attempts to hightlight two of them : the lack of efficiency of the pattern recognition algorithm and the difficulty to maintain the knwoledge base.

Introduction - Principe général de la codification Sicore

Sicore [1] permet à partir d'un libellé reçu en entrée et éventuellement quelques variables annexes de renvoyer un code si la codification est possible.

L'algorithme de codification Sicore se déroule en 3 étapes :

1. Le libellé est d'abord normalisé. Au cours de cette étape, des caractères (usuellement des caractères de ponctuation) peuvent être supprimés ou remplacés par des espaces. Le libellés est ensuite tokenizé en mots. Des expressions de synonymie sont alors appliquées successivement sur ces mots. Ce traitement permet par exemple de supprimer des mots jugés insignifiant ou homogénéiser certaines syntaxes pour simplifier la reconnaissance du libellé par la suite. Enfin, le libellé est « calibré » afin d'uniformiser la longueur en vue de l'étape suivante. Seul les n premiers mots sont retenus et, pour chacun d'entre-eux, des espaces sont rajoutés ou des caractères sont tronqués afin de fixer la longueur de chacun de ces mots.
2. Le libellé normalisé est découpé en groupe successif de caractères. Il s'agit, en général, de groupes de deux caractères appelés bigrammes. Sur ces derniers, s'applique une règle de décision sous forme d'arbre afin de tenter de reconnaître le libellé.
3. Enfin, et de manière facultative si le libellé est reconnu mais est insuffisant pour coder dans la nomenclature, des règles de décision s'appuyant sur les valeurs d'autres variables (numériques ou catégorielles) sont alors parcourues afin d'attribuer un code.

Cet article s'intéresse à la décomposition de chacune de ces étapes et tente de dégager leurs apports et leurs limites sur la codification. Ces travaux ont pu être conduit en grande partie grâce une opération de rétro-documentation de Sicore menée à l'automne 2020. Cette dernière a permis d'avoir une vision plus précise de l'implémentation et de la structure des bases de connaissances pour en rendre compte ici.

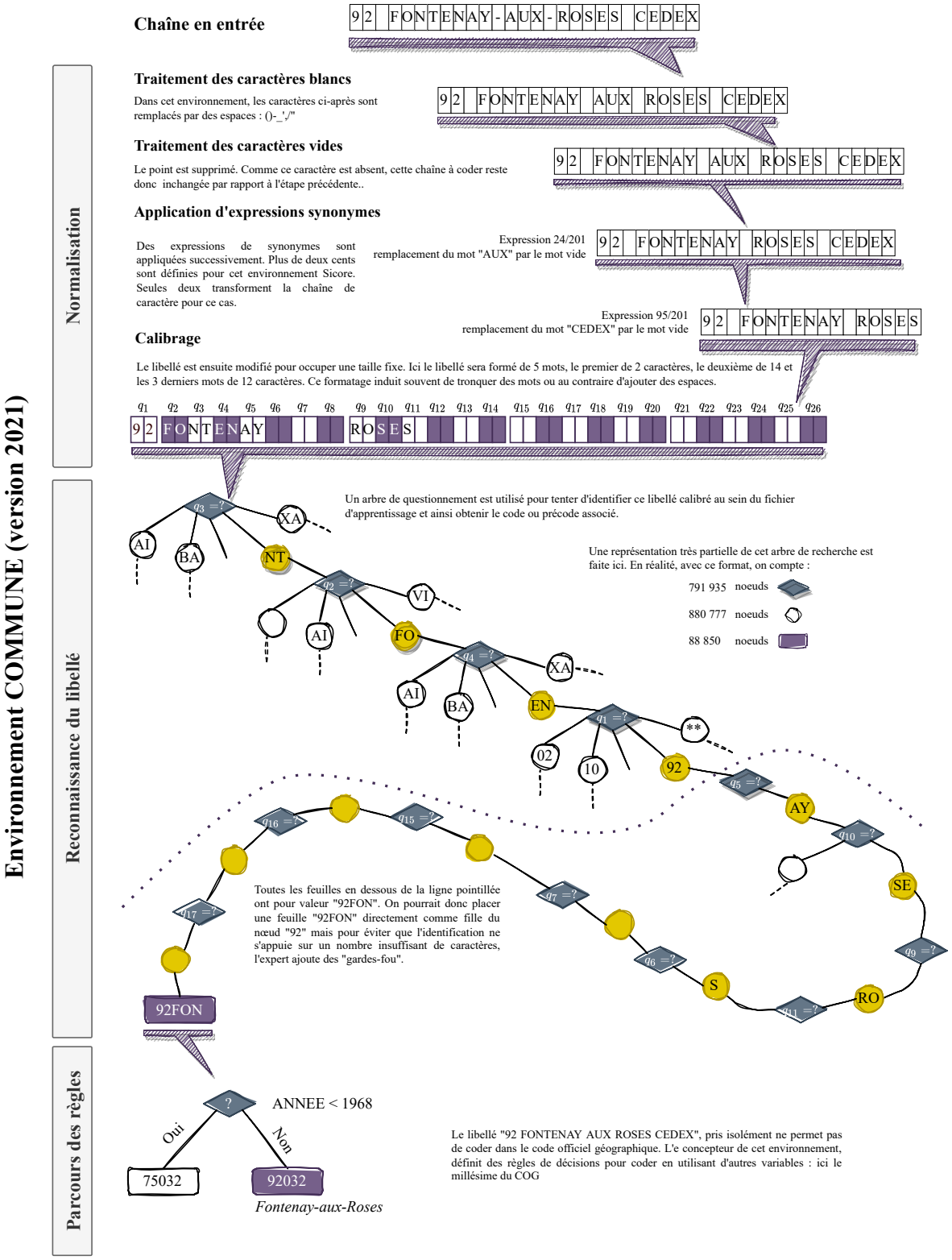


FIGURE 1 – Exemple détaillé de la codification du libellé « 92 FONTENAY-AUX-ROSES CEDEX » avec l'environnement Sicore Commune 2021

1 Étude de la pertinence de la reconnaissance du libellé sous forme d'arbre de questionnement

L'arbre de bigramme de caractères constitue la structure fondamentale au coeur de l'outil Sicore capable d'interpréter le libellé. On s'interroge donc dans la suite sur les performances induites par ce type d'objet.

1.1 Un programme d'optimisation privilégiant la rapidité de codage

L'arbre de questionnement est construit selon l'algorithme QUID [2] aussi connu dans la littérature plus générale sur les arbres de décision sous le nom d'ID3 [3]. Une liste de couple (libellé normalisé,code) appelé fichier d'apprentissage sert de base pour la construction.

Libellé d'origine	**Libellé normalisé**	**Code**
01 ABERGEMENT CLEMENCIAT	01ABERGEMECLEMEN	01001
⋮	⋮	⋮
08 REGNIOWEZ	08REGNIOWE_____	08355
08 REMAUCOURT	08REMAUCOU_____	08356
08 REMILLY AILLICOURT	08REMILLY_____	08357
08 REMILLY POTHEES	08REMILLY_POTHEE	08358
08 RENNEVILLE	08RENNEVIL_____	08360
08 RENWEZ	08RENWEZ_____	08361
08 RETHEL	08RETHEL_____	08362
08 BOIS BRYAS	08BOIS____BRYAS_	08363
08 LA BOUVERIE	08LA_____BOUVER	08363
08 LES BROUTAYS	08LES_____BROUTA	08363
08 ORZY	08ORZY_____	08363
⋮	⋮	⋮
98 ILE CLIPERTON	98ILE_____CLIPPE	98CLI

TABLE 1 – Représentation de la structure du fichier d'apprentissage brut et du fichier d'apprentissage normalisé

Dans le cas particulier de QUID, le libellé normalisé n'est pas utilisé tel quel mais est subdivisé en groupe de caractères successifs appelé « atomes ». Comme les libellés normalisés sont calibrés pour occupé une longueur fixe de ℓ caractères et sont partitionnés en groupe de caractère de longueur a (avec généralement $a = 2$), $m = \ell/a$ variables sont alors formées.

Ce sont ces potentielles m variables qui sont exploitées dans l'arbre de décision pour reconnaître un nouveau libellé et le cas échéant lui affecter un code. L'objectif lors de l'apprentissage est de déterminer pour chaque noeud la variable $q_1,\ q_2, \ldots,$ ou q_m la plus pertinente pour coder. L'algorithme choisi pour transformer cette liste en une structure de décision hiérarchique vise à sélectionner récursivement l'attribut ($q_1,\ q_2, \ldots,$ ou q_m) apportant le plus d'information selon l'entropie de Shannon. En d'autre termes, il s'agit de déterminer de localement l'atome le plus judicieux pour segmenter les données afin que les codes soient les plus homogènes dans les noeuds fils.

Code	**Bigramme**							
(T)	q_1	q_2	q_3	q_4	q_5	q_6	q_7	q_8
01001	01	AB	ER	GE	ME	CL	EM	EN
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
08355	08	RE	GN	IO	WE	__	__	__
08356	08	RE	MA	UC	OU	__	__	__
08357	08	RE	MI	LL	Y_	__	__	__
08358	08	RE	MI	LL	Y_	PO	TH	EE
08360	08	RE	NN	EV	IL	__	__	__
08361	08	RE	NW	EZ	__	__	__	__
08362	08	RE	TH	EL	__	__	__	__
08363	08	BO	IS	__	__	BR	YA	S_
08363	08	LA	__	__	__	BO	UV	ER
08363	08	LE	S_	__	__	BR	OU	TA
08363	08	OR	ZY	__	__	__	__	__
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
98CLI	98	IL	E_	__	__	CL	IP	PE

TABLE 2 – Représentation de la structure des données (variables explicatives $q_1,\ q_2, \ldots$ et de la variable à prédire T) pour l'apprentissage de l'arbre de reconnaissance du libellé

On note :

— T : la variable correspondant aux codes. Elle contient K modalités $t_1, t_2, \ldots, t_K$. Par exemple, t_1 peut être 01001 le code correspondant à la commune de Abergement-Clémenciat.

— x : un noeud dans l'arbre de décision. Par exemple, x peut être le noeud qui rassemble toutes les observations où le 1er bigramme du 2ème mot est « PA » et le 2ème bigramme du premier mot vaut « RI ». Dans ce cas, les libellés « 75 PARIS » ou « 50 PARIGNY » appartiennent au noeud. La proportion du code t_k parmi les observations correspondant aux caractéristiques du noeud x est notée $P(t_i|x)$. L'entropie $H(T|x)$ de la variable T au sein du noeud x est alors définie par :

$$H(T|x) = -\sum_{\substack{k\in\{1,\ldots,K\}\\ P(t_k|x)>0}} P(t_k|x)\ln\left(P(t_k|x)\right)$$

Plus le noeud concentre des codes homogènes et plus l'entropie sera faible (peu de désordre).

— q_j : la variable correspondant au j-ème bigramme

— $\Gamma_{q_j}(x)$: l'ensemble des noeuds fils de x en choisissant le bigramme q_j, c'est ensemble contient autant d'éléments que de modalités possibles pour la variable q_j contionnellement à x. Pour tout noeud, $y \in \Gamma_{q_j}(x)$ la proportion des libellés du fichier d'apprentissage appartenant au noeud y conditionnellement à x est notée $P(y|x)$

Ainsi, on sélectionne récursivement, à chaque noeud x, le bigramme q_j pouvant encore être choisi qui minimise l'entropie moyenne de l'ensemble des fils :

$$\underset{q_j}{\operatorname{argmin}} \sum_{y\in\Gamma_{q_j}(x)} P(y|x)H(T|y)$$

Il est également possible pour le concepteur de la base de connaissance de définir de atomes « prioritaires » et d'imposer ainsi le tout début du questionnement autour de certains bigrammes

plutôt que de laisser l'algorithme d'apprentissage faire la sélection sur l'intégralité des noeuds. Cette option réduit le temps pour l'apprentissage de l'arbre de décision mais peut augmenter la profondeur de l'arbre car il n'est plus construit partout sur un choix optimal.

En pratique, même si les bigrammes prioritaires sont employés dans la grande majorité des environnements Sicore, les arbres Sicore restent très peu profonds. Par exemple, la reconnaissance du libellé s'appuie en moyenne sur un peu moins de 4 bigrammes pour la codification des communes de résidence des Bases Tous Salariés 2017 alors le fichier d'apprentissage contient près de 90 000 observations. La phase de reconnaissance du libellé au sein du fichier d'apprentissage est donc très efficace. Sicore est un outil rapide.

Il faut donc voir cet arbre QUID comme un moyen de minimiser le temps de recherche au sein de la base d'apprentissage grâce à une structure hiérarchique et donc plutôt sous l'angle informatique et donc moins sous l'angle statistique d'arbre de classification. C'est d'ailleurs pour cette raison calculatoire que cette structure est justifiée dans l'article fondateur de J. Lorigny. Dans une approche plus statistique, on aurait sans doute choisit d'autres variables explicatives comme la présence ou non de certains mots ou de certains n-grammes. De plus, les fichiers d'apprentissage seraient constitués plus directement à partir de données réelles issues d'enquêtes ou de sources administratives et non des exemples ajouter au cas par cas par des experts qui enrichissent les bases de connaissances Sicore.

1.2 Des atomes de redondance pour gagner en fiabilité mais perdre en efficacité

En conséquence, des arbres peu profonds construits par réduction d'entropie, la décision peut parfois être fondée sur un nombre très réduit de caractères. Par exemple, dans l'arbre présenté figure 1, le noeud $q_3 = \text{KW}$ conduit immédiatement à coder 67119 pour la commune d'Eckwersheim ou $q_3 = \text{CG}$ mène directement sur 64358 (Lucgarier). Plus précisément, c'est le cas pour 48 des 465 valeurs possibles de q_3 du premier noeud dans l'environnement Commune.

Pour se prémunir d'une codification trop hâtive, Sicore autorise la définition de contrôles supplémentaires entre le libellé du fichier d'apprentissage et le libellé à coder sur des atomes considérés comme important par l'expert. En général, ce sont les premiers atomes des premiers mots qui sont retenus. Les erreurs sont ainsi fortement réduites. Par exemple, pour coder les communes de résidence dans la base tous salariés 2017, les codages avec un échec du contrôle de redondance sont inexacts à 24 % contre environ 1 % d'erreur pour ceux qui passent le contrôle de redondance avec succès.

Toutefois, lorsque le nombre de bigramme de redondance augmente, on peut s'interroger sur la pertinence de ces arbres de reconnaissance du libellé et en particulier évaluer le gain sur le taux de libellé codé automatiquement (efficacité) par rapport à une recherche strict sur le libellé au sein du fichier d'apprentissage. La table 3 montre que dans la plupart des cas testés, le contrôle de cohérence est tel que l'arbre de reconnaissance du libellé n'est pas beaucoup plus efficace qu'une méthode naïve d'appariement strict sur un libellé normalisé. Par exemple, pour coder le pays de naissances, on code automatiquement 0,8 point d'observations en plus que lorsqu'on recherche strictement le libellé au sein du fichier d'apprentissage normalisé. Le gain est un peu plus important sur les environnements pour les professions en PCS 2003 ou les liasses Sirene provenant de certains type de centre de formalité des entreprises. La table 3 permet d'isoler l'apport des différentes étapes de l'algorithme Sicore sur l'efficacité et indique que l'essentiel de l'identification provient de la phase de preprocessing des données et non de cette structure originale d'arbre de décision portant sur des valeurs de bigrammes.

	Nombre d'observations	Base de connaissances Sicore	Pourcentage d'identification (efficacité de la méthode)		
			Appariement strict sur le libellé non normalisé	Appariement strict sur le libellé normalisé	Arbre de reconnaissance Sicore
Commune de résidence (Base tous salariés 2017)	27 000 000	Commune version 2021	69,5	97,4	98,2
Pays de naissance (EAR 2020)	900 000	PAYS version 2020	94,1	98,4	99,2
Profession actuelle des salariés (EAR 2020)	1 500 000	PCS2003R et REFNC version 2020	33,0	76,3	87,5
Profession actuelle des indépendants (EAR 2020)	220 000	PCS2003R et REFNC version 2020	50,4	77,8	88,7
Profession antérieure (EAR 2020)	1 200 000	PCS2003R et REFNC version 2020	33,7	75,7	88,7
Activité économique pour les liasses provenant des chambres de commerce et d'industrie (Sirene 2020)	180 000	NR21C version 2021	22,1	33,2	48,6
Activité économique pour les liasses provenant des greffes des tribunaux de commerce (Sirene 2020)	500 000	NR21C version 2021	26,8	39,2	55,0
Activité économique pour les liasses provenant des chambres de métiers et de l'artisanat (Sirene 2020)	200 000	NR21C version 2021	35,3	49,0	63,3
Activité économique pour les liasses provenant des chambres des URSSAF (Sirene 2020)	500 000	NR21U version 2021	6,3	81,1	81,1
Activité économique pour les liasses provenant des chambres d'agriculture (Sirene 2020)	30 000	NR21X version 2018	52,3	89,3	89,9

TABLE 3 – Comparaison du taux de reconnaissance du libellé par Sicore par rapport à des méthodes naïves d'appariement strict

2 Un enrichissement difficile du moteur de règles

Sicore est un un système expert. Ainsi, pour accroître les performances des bases de connaissances, un expert doit ajouter de nouvelles règles de normalisation ou d'exploitation des variables annexes et compléter le fichier d'apprentissage de nouveaux exemples. Il doit transposer ces connaissances dans le système de règles Sicore afin que son raisonnement soit reproduit.

2.1 Risque d'explosion du conditionnement logique dans Sicore

Pour concevoir l'étape de questionnement des variables annexes, l'expert doit déclarer des règles de type "SI ... ALORS ... , SINON SI ... ALORS ..., SINON ..." dans la syntaxe de Sicore. Par exemple, pour gérer l'historique du COG pour le libellé Petit-Mercey, le raisonnement logique déclaré dans Sicore Commune repose sur au plus 3 conditions.

Sicore

```
39PET DATE   ;
REGLE < 1973 ; 39414 ; /* 01/01/1973 : Le Petit-Mercey est rattachée à Louvatange (39302) */
REGLE < 1985 ; 39302 ; /* 01/01/1985 : Le Petit-Mercey est rétablie. */
REGLE < 2019 ; 39414 ; /* 01/01/2019 : Le Petit-Mercey devient commune déléguée au sein de
↪ Dampierre  (39190) (commune nouvelle) */
REGLE        ; 39190 ;
```

On peut représenter dans la figure 2 cet enchaînement de conditions sous forme de graphe orienté avec 3 noeuds pour les conditions et 4 pour les codes en sortie.

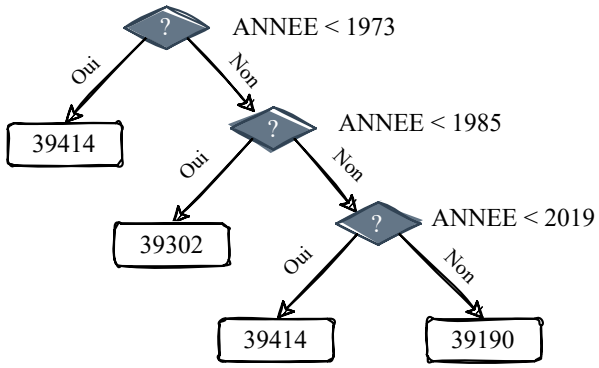


FIGURE 2 – Représentation sous forme de graphe orientée de la règle logique pour le libellé Petit-Mercey dans l'environnement Sicore Commune

Ce cas est relativement simple car il ne tient compte que d'une variable annexe et au pire 3 contraintes à combiner. Toutefois, certains graphes sont très complexes car il est possible avec Sicore de créer des noeuds où la contraintes porte sur plusieurs variables mais aussi de définir des noeuds de condition en incidence des arêtes « non » et « oui » comme sur la figure 3. Dans cette figure, le nombre de conditions pour balayer tous les cas resterait de trois mais le nombre de conditions à parcourir pour coder serait ici toujours de 2. La table 4 permet de se rendre compte de la taille du graphe des règles pour différents environnements Sicore. Certains graphes, bien que grand, comme ceux des environnements ISCO2020 ou PCS2020R, ne sont pas nécessairement compliqués à interpréter car on peut isoler l'ensemble en petits sous-graphes indépendants pour chaque entrée du graphe provenant de l'étape de reconnaissance du libellé. Dans d'autres cas, tels que les environnements pour la PCS 2003 ou pour la PCS-ESE, les tailles de ces sous-graphes suivent une distribution asymétrique à gauche avec des valeurs extrêmes de plusieurs centaines de conditions. Ces environnements comportent de nombreuses variables annexes avec beaucoup de modalités et incitent alors à générer un questionnement très riche.

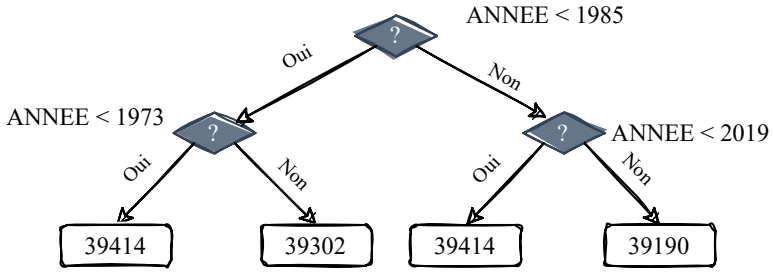
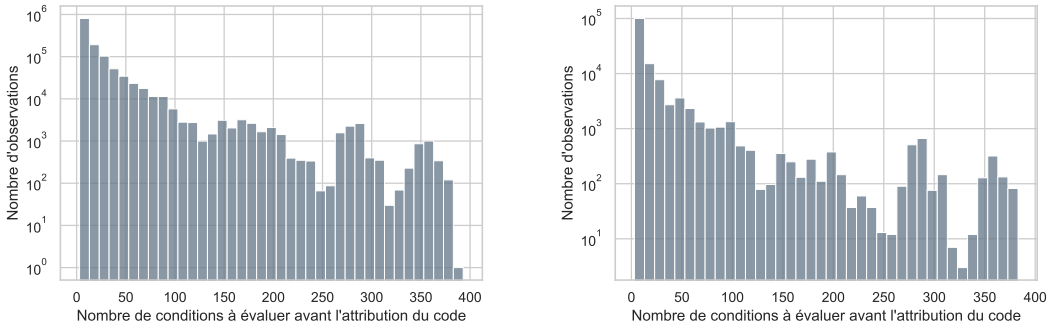


FIGURE 3 – Alternative de règle pour le libellé Petit-Mercey

Environnement Sicore	Nombre de noeuds de conditions	Nombre de noeuds entrants	Statistiques de la distribution du nombre de noeuds de conditions accessibles depuis chaque noeud entrant			
			Minimum	Médiane	Moyenne	Maximum
PCS2003R						
codification de la PCS 2003 dans les enquêtes ménages et les EAR	14 775	2 446	4	14	194	2 332
REFNC						
codification en PCS 2003 en complément de PCS2003R pour les EAR uniquement	14 775	1 402	4	16	214	2 328
PCSDSN						
codification en PCS-ESE lors du traitement des DSN	17 346	934	0	11	278	1 152
ISCO						
codification en ISCO-08 à partir de la PCS 2003	263 689	2 442	0	29	108	1 159
ISCO2020						
codification en ISCO-08 à partir de la liste des professions	69 432	5 786	12	12	12	12
PCS2020R						
codification en PCS 2020 des professions dans la liste des professions	98 362	5 786	17	17	17	17
COMMUNE						
codification dans le COG	4 420	4 124	1	1	1	3
NR21C						
codification en NAF5 des liassses Sirene provenant des CCI ou greffes	547	148	1	4	4	10
NR21U						
codification en NAF5 des liasses Sirene provenant des Urssaf	120	37	1	3	4	8

TABLE 4 – Indicateurs de complexité des graphes de règles dans différents environnements Sicore

La complexité soulevée ici n'est pas exclusivement théorique en étudiant des tailles de composantes connexes. Cet empilement de conditions intervient en pratique pour le codage de jeux de données. Par exemple, lors du codage des professions actuelles de l'EAR 2020, près de 90 % des données sont codées et, parmi elles, un quart nécessite de parcourir au moins 20 conditions lors du parcours des règles. Cette accumulation de conditions survient principalement pour des libellés assez flous qui requièrent pas mal d'information annexes pour préciser la codification comme « AGENT DE PRODUCTION », « EMPLOYE DE BUREAU », etc. La figure 4 quantifie la fréquence de ces évènements dans l'EAR 2020. Parfois, le parcours de plus de 100 conditions est nécessaire avant l'attribution d'un code.



(a) EAR 2020 - Profession actuelle des salariés

(b) EAR 2020 - Profession actuelle des non-salariés

FIGURE 4 – Histogrammes du nombre de conditions testées dans les règles avant d'aboutir à un code (échelle logarithmique pour les fréquences)

2.2 Incidence de la complexité cognitive des règles sur l'amélioration des connaissances Sicore

La complexité s'accroissant au fil des mises à jours, il est difficile pour les experts variable Sicore d'enrichir les bases de connaissances. Prendre en compte une suite de plusieurs conditions devient abscons à partir d'un certain stade pour le cerveau humain. Certains environnements Sicore connaissent alors une sorte de point de saturation avec peu de modification chaque année malgré la bonne volonté et les compétences des experts de la codification d'une variable.

	Mise à jour 2017	Mise à jour 2018	Mise à jour 2019
Nombre d'ajouts de règles logiques	17	9	12
Nombre de suppressions de règles logiques	12	0	3
Nombre de modifications des règles logiques existantes	11	9	4

TABLE 5 – Nombre de modifications au sein du fichier de règles lors des mises à jour des environnements pour la PCS 2003

La table 5 indique que de l'ordre d'une dizaine de modifications sont réalisées sur le fichier de règles des environnements PCS 2003 chaque année. Toutefois, ce travail fastidieux ne permet pas de capitaliser pleinement sur la reprise manuelle où plus de 140 000 libellés de professions actuelles uniques sont repris chaque année.

Conclusion

Sicore est un outil très pratique qui répond à un grand nombre de besoins mais présente quelques limites. Il peut être difficile de maintenir les bases de connaissances. De plus des fonctionnalités sont absentes comme une capacité à manipuler des libellés incomplets en autocompletion ou proposer des recommandations à un gestionnaire plutôt qu'un seul code lors des reprise manuelle.

Comme nous avons pu le voir en première partie, Sicore repose sur une structure d'arbre originale et très performante qui rend l'outil très rapide pour coder. Cependant, le contexte a changé avec l'accroissement des capacités de calcul des ordinateurs. Il ne semble plus vraiment nécessaire d'optimiser la classification selon le temps de calcul mais sur un arbitrage plus statistique entre efficacité et fiabilité du codage automatique. Des approches par *machine learning* sont peut être préférables dans certaines situations. Elles permettent de profiter plus facilement de la reprise manuelle et enterrent les insertions des connaissances au cas par cas comme aujourd'hui. Sicore intègre malheureusement une partie des défauts du *machine learning* sans avoir les avantages évoqués. En effet, même si Sicore est un système expert construit à partir de règles, l'interprétation de la décision qui mène au code est difficile à avoir vu l'intrication des règles.

Au vu de la situation, il semblerait nécessaire de diversifier les offres et de méthodes de codifications pour répondre au mieux aux besoins propres de chaque variable tout en gardant une maîtrise méthodologique de l'ensemble.

Références

[1] RIVIERE Pascal, Sicore, un outil et une méthode pour le chiffrement automatique à l'Insee", Courrier des statistiques, n°74 - août 1995

[2] LORIGNY Jacques, QUID, une méthode générale de chiffrement automatique, Techniques d'enquêtes, Vol 14, Décembre 1988

[3] QUINLAN, J.. Ross . Induction of decision trees. Machine learning, 1986, vol. 1, no 1, p. 81-106