

REVUE DE STATISTIQUE APPLIQUÉE

M. CHAVENT

M. TOUATI

Recodage et classification symbolique d'un tableau de données temporelles. Application à l'étude du comportement d'utilisateurs

Revue de statistique appliquée, tome 45, n° 3 (1997), p. 73-88

http://www.numdam.org/item?id=RSA_1997__45_3_73_0

© Société française de statistique, 1997, tous droits réservés.

L'accès aux archives de la revue « *Revue de statistique appliquée* » (<http://www.sfds.asso.fr/publicat/rsa.htm>) implique l'accord avec les conditions générales d'utilisation (<http://www.numdam.org/conditions>). Toute utilisation commerciale ou impression systématique est constitutive d'une infraction pénale. Toute copie ou impression de ce fichier doit contenir la présente mention de copyright.

NUMDAM

Article numérisé dans le cadre du programme
Numérisation de documents anciens mathématiques

<http://www.numdam.org/>

RECODAGE ET CLASSIFICATION SYMBOLIQUE D'UN TABLEAU DE DONNÉES TEMPORELLES. APPLICATION À L'ÉTUDE DU COMPORTEMENT D'UTILISATEURS

M. Chavent (1, 2) et M. Touati (2)

(1) INRIA Rocquencourt.

(2) LISE CEREMADE, Université Paris IX Dauphine & URA 749-CNRS.

RÉSUMÉ

Il s'agit d'une méthode permettant de recoder un tableau incomplet de données temporelles. L'objectif de ce codage est de réduire la complexité du tableau, en regroupant et en résumant au mieux les différentes mesures initiales sur des intervalles de temps cohérents.

Pour trouver ces intervalles de temps, on applique un algorithme de classification optimale sous contrainte d'ordre total (capable de gérer les données manquantes).

Plusieurs méthodes sont proposées pour résumer au mieux les valeurs observées sur ces intervalles.

Ce recodage a été développé dans le cadre d'une application dont le but est l'étude du comportement d'utilisateurs. A partir du recodage des données de l'application, une méthode originale de classification des utilisateurs est développée et plusieurs types de comportements sont identifiés.

Mots-clés : *Données temporelles, codage symbolique, classification optimale, classification hiérarchique symbolique, profil d'utilisateurs.*

ABSTRACT

This paper begins with a coding method applied to a temporal data table containing missing data. The complexity of this data table is reduced by grouping and summarizing the initial values on "good" times period.

In order to find those periods, we apply an optimal clustering algorithm under total order's constraints (this algorithm deals with missing data).

Several methods are proposed to summarize the initial values on each period.

This coding has been developed within an application in order to study the user's behaviour. From the coding of those data, a new clustering method has been applied to the users of this application, and some typical behaviours have been identified.

Keywords : *Temporal data, symbolic coding, optimal clustering, symbolic and hierarchical clustering, user's profile.*

1. Introduction

Le but de l'étude est de trouver des classes d'utilisateurs qui ont le même type de comportement, afin de pouvoir identifier ultérieurement la classe à laquelle appartient un nouvel utilisateur grâce à la connaissance de ses habitudes. Dans l'application présentée ici, le comportement d'un utilisateur est mesuré par deux types de variables qui rendent compte d'une part de l'environnement dans lequel se trouve l'utilisateur, et d'autre part de l'ambiance choisie par l'utilisateur dans cet environnement. Les variables dites d'environnement sont caractérisées par le fait que les utilisateurs ne peuvent pas les influencer. Les variables d'ambiance par contre peuvent être contrôlées par les utilisateurs puisque qu'elles traduisent les réactions d'un utilisateur dans un environnement. Le comportement d'un utilisateur sera donc caractérisé par l'ambiance qu'il associe à un environnement donné et l'on cherchera à obtenir des classes d'utilisateurs qui à un même environnement, associent une même ambiance.

La difficulté de cette classification vient du caractère temporel des données. En effet le comportement des utilisateurs est mesuré tout au long de la journée, et le caractère aléatoire des variables d'environnement que l'on ne peut pas contrôler, impose le traitement d'un tableau ternaire utilisateurs \times variables \times temps. De plus, la taille importante du tableau et la présence d'un grand nombre de données manquantes va nécessiter une étape préalable de recodage. Finalement, on va chercher une interprétation simple des classes d'utilisateurs obtenues, cette interprétation fournissant des comportements types.

Dans une première partie, nous présentons plus précisément les données de l'application et les transformations tabulaires nécessaires pour la suite du traitement. Dans une deuxième partie, nous proposons une méthode de recodage du tableau de données initiales, ce recodage permettant de réduire le nombre de données manquantes, le nombre de variables, et de pallier le manque de fiabilité des données. Dans une troisième partie, nous proposons une méthode de classification des utilisateurs. La principale difficulté de cette étape réside dans le fait qu'un même utilisateur se retrouve dans plusieurs lignes du tableau initial, cet utilisateur étant observé sur des jours différents.

2. Description des données

Les données ont été fournies par les Laboratoires d'Électronique Philips S.A.S. Le tableau initial est ternaire puisqu'il met en relation les utilisateurs, les variables et le temps. Plus précisément, les données décrivent 130 utilisateurs distincts observés sur une durée de 500 jours, chaque utilisateur n'étant pas forcément présent tous les jours. Les mesures sont enregistrées 10 heures par jour toutes les 6 minutes (soit 101 fois par jour) pour chaque utilisateur présent. Le nombre de variables ainsi mesurées est de 47. Le tableau des données initiales décrit donc en ligne les 130 utilisateurs et en colonne les 47 variables mesurées sur 101 unités de temps pour 500 jours d'observations.

TABLEAU 1 : *Tableau initial*

	Jour 1			...	Jour 500		
	unité de temps 1		unité de temps 101		unité de temps 1		unité de temps 101
	$V_1 \dots V_{47}$...	$V_1 \dots V_{47}$...	$V_1 \dots V_{47}$...	$V_1 \dots V_{47}$
u_1							
\vdots			?				
u_{130}					?	?	

Les 47 variables sont binaires, nominales ou numériques et sont structurées en deux groupes :

- les variables d'environnement (au nombre de 24) sur lesquelles les utilisateurs n'ont aucune influence.
- les variables d'ambiance (au nombre de 23) qui font état des actions des utilisateurs pour s'adapter à l'environnement.

Le tableau initial fait état du mode de recueil des données mais les utilisateurs n'étant pas présents tous les jours, le tableau comporte beaucoup de données manquantes et sera traité sous une nouvelle forme où les jours sont déployés en ligne.

TABLEAU 2 : *Tableau déployé*

	V_1			...	V_{47}		
	t_1	...	t_{101}		t_1	...	t_{101}
w_1					?		
\vdots		?					
w_{2600}							?

Une ligne du tableau 2 correspond donc à une nouvelle unité statistique appelée utilisateur/jour et notée w_i , c'est à dire à un utilisateur observé durant une journée particulière. Ce tableau ne comporte que 2600 lignes (un utilisateur donné n'est observé que sur quelques jours parmi les 500 jours où l'expérience a été menée).

3. Codage des données

Le traitement direct du tableau de données déployé (voir tableau 2) est difficile. En effet, il comporte encore un grand nombre de données manquantes, les utilisateurs n'étant pas observés à tous les instants de la journée (absents, défaillance du capteur).

De plus, on remarque le nombre important de variables (47×101) par rapport au nombre d'utilisateur/jour (2600).

La solution proposée est de regrouper les 101 unités de temps en intervalles pendant lesquels les utilisateurs ont eu des comportements «cohérents». Il s'agit là d'un problème de classification sous contrainte d'ordre total. On recode ensuite chaque utilisateur/jour par un résumé des valeurs prises par cet individu sur l'intervalle de temps correspondant.

Par la suite, l'ensemble des traitements seront effectués sur un tableau réduit de 165 utilisateurs/jours et 9 variables dont 4 variables d'environnement et 5 variables d'ambiance (voir tableau 4). Les 165 utilisateurs/jour sont issus des 9 utilisateurs choisis par l'expert. Ces 9 utilisateurs sont ceux ayant été observés sur le plus grand nombre de jours (voir tableau 3). L'expert a également sélectionné les 9 variables les plus représentatives.

TABLEAU 3
Nombre de jours observés par utilisateur

Nom de l'utilisateur	Nombre de jours observés
u_1	12
u_2	43
u_3	6
u_4	3
u_5	5
u_6	6
u_7	21
u_8	56
u_9	13
TOTAL	165

TABLEAU 4 : *Tableau réduit*

	V_1			...	V_{47}		
	t_1	...	t_{101}		t_1	...	t_{101}
w_1			?				
\vdots							
w_{165}	?				?		

3.1. Choix des intervalles de temps : classification sous contrainte d'ordre total

Il s'agit ici de trouver une classification optimale des 101 unités de temps en respectant leur ordre. Pour résoudre ce problème, on a appliqué un algorithme de classification optimale sous contrainte d'ordre total sur le tableau transposé (voir tableau 5), ce tableau mettant en correspondance les 101 unités de temps avec les 5 variables d'ambiance et les utilisateurs du tableau réduit (tableau 4). Ici, seules les variables d'ambiance sont prises en compte. C'est un tableau classique (une seule valeur par variable) de 5 variables quantitatives, incomplet, contenant beaucoup de données manquantes.

Le problème des données manquantes a été traité de deux manières différentes. La première approche consiste à remplacer les valeurs manquantes par la moyenne calculée sur les valeurs présentes. La deuxième approche consiste à modifier l'algorithme de classification, afin qu'il ne tienne compte lors du calcul du critère que des données réellement observées. Cette deuxième approche sera développée un peu plus loin.

TABLEAU 5 : *Tableau transposé*

	V_1		V_5
	$w_1 \dots w_{165}$...	$w_1 \dots w_{165}$
t_1	?	?	
...			
t_{101}			

Le découpage optimal du temps s'écrit de manière générale sous la forme suivante : trouver une partition $P = (P_1, \dots, P_K)$ d'un ensemble $E = \{t_1, \dots, t_n\}$ de n individus totalement ordonnés décrits par p variables. Cette partition doit être optimale au sens du critère choisi, et admissible, c'est à dire que chaque classe de la partition doit vérifier la propriété d'ordre :

$$\forall k = 1, \dots, K, \forall i, j, l \text{ tels que } t_i, t_j \in P_k \text{ et } t_i < t_l < t_j \text{ alors } t_l \in P_k$$

Cette problématique s'apparente à celle du découpage optimal d'une variable quantitative (voir tableau 6), dont une des solutions a été apportée par Fisher (Fisher 1958).

Pour résoudre notre problème (voir tableau 7) nous avons donc appliqué un algorithme de programmation dynamique (Jensen, 1969), sous contrainte d'ordre total (Lechevallier, 1990), appelé algorithme de Fisher «généralisé».

La «généralisation» se situe à deux niveaux :

- au niveau de la propriété d'ordre qui est donnée par un ordonnancement à priori des individus et non plus par un ordonnancement obtenu à partir des valeurs prises par les individus.

- au niveau du critère qui mesure l'inertie intra-classe sur les p variables décrivant les individus au lieu d'une seule variable.

TABLEAU 6 : *Problème de Fisher*

ind	ordre $\overbrace{X_1}$	X_2 X_3
e_1 e_2 ...	← inertie →	

TABLEAU 7 : *Notre problème*

ordre $\overbrace{\text{ind}}$	X_1 X_2 X_3
e_1 e_2 ...	← inertie →

3.1.1. Présentation de l'algorithme de Fisher «généralisé»

Les n éléments e_1, \dots, e_n totalement ordonnés sont représentés dans \mathfrak{R}^p par x_1, \dots, x_n , avec $x_i = (x_i^1, \dots, x_i^j, \dots, x_i^p)$ ($1 \leq i \leq n$) et munis des poids p_1, \dots, p_n .

Le critère de qualité d'une partition $P = (P_1, \dots, P_K)$ que nous voulons optimiser est :

$$W(P) = \sum_{k=1}^K \sum_{e_i \in P_k} p_i d^2(x_i, \bar{x}_k)$$

où $\bar{x}_k = (\bar{x}_k^1, \dots, \bar{x}_k^j, \dots, \bar{x}_k^p)$ est le centre de gravité de la classe P_k

$$\bar{x}_k^j = \frac{1}{\mu_k} \sum_{e_i \in P_k} p_i x_i^j \quad \text{et} \quad \mu_k = \sum_{e_i \in P_k} p_i$$

et $d^2(x_i, \bar{x}_k)$ le carré de la distance euclidienne entre x_i et \bar{x}_k .

W étant la somme des inerties des K classes, ce critère est additif. W est l'inertie intra-classe de la partition P .

L'algorithme utilisé est un algorithme de programmation dynamique basé sur le principe suivant : comme le critère $W(P)$ est additif, alors toute sous-partition d'une partition optimale est optimale sur son ensemble de définition (voir figure 1).

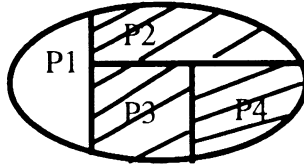


FIGURE 1

Partition optimale en 4 classes et sous-partition optimale en 3 classes

Ainsi le nombre de partitions à envisager pour obtenir la partition optimale est fortement réduit, mais la complexité est encore en 2^{2n} . Ici, la contrainte d'ordre total permet de réduire cette complexité à n^2 car les classes admissibles doivent vérifier la propriété d'ordre.

On note :

- $Q_j(k)$ est la meilleure partition en k classes de l'ensemble $\{e_j, \dots, e_n\}$.
- $F_j(k) = W(Q_j(k))$ est l'inertie de la meilleure partition en k classes de l'ensemble $\{e_j, \dots, e_n\}$.
- $C_{i,j}$ est l'inertie de l'ensemble $\{e_i, \dots, e_j\}$.
- $v_i(k)$ est l'indice du dernier individu de la première classe de gauche de $Q_j(k)$ (la meilleure partition en k classes à partir de e_i).

Le principe est le suivant :

si pour tout j , $Q_{j+1}(k - 1)$, la meilleure partition en $k - 1$ classes à partir de e_{j+1} est connue (donc $F_{j+1}(k - 1)$ connu), alors $Q_{i+1}(k)$ la meilleure partition en k classes à partir de e_{i+1} est la partition $(\{e_{i+1} \dots e_j\}, Q_{j+1}(k - 1))$ telle que la valeur $C_{i+1,j} + F_{j+1}(k - 1)$ soit minimale. On a alors $v_{i+1}(k) = j$.

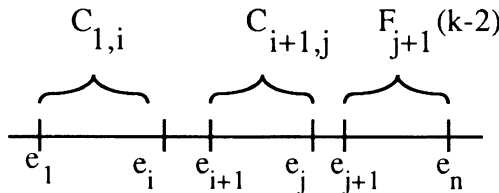


FIGURE 2

Partition en k classes

L'algorithme est donc le suivant :

Initialisation :

Pour $j = 1$ à $n - 1$, calcul de l'inertie des partitions en une classe à partir de j , soit $F_{j+1}(1) = C_{j+1,n}$.

A chaque étape k ($k = 3$ à K nombre de classes désirées) :

Pour i variant de 1 à $n - k - 1$

trouver j^* qui minimise $C_{i+1,j} + F_{j+1}(k - 2)$

$$F_{i+1}(k - 1) = C_{i+1,j^*} + F_{j^*+1}(k - 2)$$

$$v_{i+1}(k - 1) = j^*$$

trouver i^* qui minimise $C_{1,i} + F_{i+1}(k - 1)$

On obtient donc à chaque étape les deux premières coupures i^* et j^* de la partition optimale en k classes. Les autres coupures de la partition optimale en k classes ont été calculées et stockées aux étapes précédentes. Ainsi, la troisième coupure a été stockée dans $v_{j^*+1}(k - 2)$ et ainsi de suite. De même, à l'étape k , pour tout i , on stocke $v_{i+1}(k - 1)$ et $F_{i+1}(k - 1)$ l'inertie de la meilleure partition en $k - 1$ classes à partir de e_{i+1} afin de les utiliser aux étapes suivantes. L'algorithme fournit donc toutes les partitions optimales de 3 à K classes.

3.1.2. Gestion des données manquantes par l'algorithme

Pour résoudre le problème des données manquantes, nous avons étudié plusieurs méthodes. Pour cette application, celle qui a fourni les meilleurs résultats consiste à ne prendre en compte que les données présentes dans le calcul des inerties intra-classe $C_{i,j}$ des classes $\{e_i, \dots, e_j\}$ et donc dans le calcul du centre de gravité de ces classes.

Le centre de gravité de la classe $\{e_i, \dots, e_j\}$ est le vecteur $\bar{x} = (\bar{x}^1, \dots, \bar{x}^q, \dots, \bar{x}^p)$ et la q -ième coordonnée du centre de gravité est le centre de gravité de la q -ième variable. Lorsqu'on tient compte des données manquantes on a :

$$\bar{x}^q = \frac{1}{\mu^q} \sum_{e_m \in \{e_i, \dots, e_j\}} \delta_m^q p_m x_m^q$$

avec

$\delta_m^q = 1$ si l'élément e_m a été observé sur la variable q , 0 sinon (donnée manquante)

$$\mu^q = \sum_{e_m \in \{e_i, \dots, e_j\}} \delta_m^q p_m = \text{poids de la classe } \{e_i, \dots, e_j\}.$$

3.2. Le codage

L'application de l'algorithme de Fisher «généralisé» sur nos données a permis de trouver un découpage en 10 classes qui correspondent chacune à un intervalle de temps.

Le découpage en 10 intervalles de temps I_1, \dots, I_{10} est le suivant : [8h-9h06], [9h12-9h54], [10h-10h54], [11h-12h48], [12h54-13h12], [13h18-13h54], [14h-14h48], [14h54-15h54], [16h-16h48], [16h54-18h]

A partir du découpage optimal obtenu à l'étape précédente, on construit un nouveau tableau mettant en relation l'ensemble des utilisateurs/jour w_1, \dots, w_{165} avec l'ensemble des variables et l'ensemble des intervalles de temps. Pour chaque utilisateur/jour, les variables vont être recodées par un résumé des valeurs prises par cet individu sur l'intervalle de temps correspondant. Le tableau ainsi recodé possède encore des données manquantes, ces valeurs manquantes correspondant au cas où aucune valeur n'a été observée sur l'intervalle de temps considéré. Cependant, le pourcentage de données manquantes a été réduit de 50,3% à 18,5%.

Nous proposons trois résumés possibles :

– une **moyenne** des valeurs observées sur l'intervalle de temps. On obtient alors un nouveau tableau de données numérique classique. Les données manquantes dans ce tableau peuvent être traitées classiquement en estimant sur les données présentes les paramètres d'un modèle (souvent la loi normale) et en remplaçant les données manquantes par des données simulées à partir de loi estimée précédemment (Afi et Elashoff, 1966).

– un **intervalle**, défini par le min et le max des valeurs observées sur l'intervalle. Chaque case du tableau de données correspond alors à un intervalle. Les données manquantes dans ce tableau peuvent être traitées de la manière suivante : on estime sur tous les intervalles [min, max] du tableau, les paramètres de la loi des min, et les paramètres de la loi des max (souvent la loi normale), et on remplace les données manquantes par des intervalles dont les min et les max sont simulés à partir des deux lois estimées précédemment.

– des **fréquences**. Si l'espace de définition de la variable est continu, il est discrétisé en classes de valeurs par l'expert. On calcule les fréquences d'apparition dans chaque classe de valeurs sur l'intervalle de temps.

Si la variable j est divisée en r_j classes, la case (i, j) du tableau de données associé à la $i^{\text{ème}}$ ligne et à la $j^{\text{ème}}$ variable, est alors un vecteur de $r + 1$ nombres, positifs ou nuls et de somme égale à 1. Le $r_j + 1 - i^{\text{ème}}$ nombre correspond au pourcentage de données manquantes dans l'intervalle de temps considéré. En effet pour traiter le problème des données manquantes, on ajoute aux r_j modalités correspondants aux r_j classes de valeurs, la modalité «manquante».

Exemple de recodage par fréquences : discrétisation de l'espace de définition des variables :

v_1 :	[2000, 3000] = faible	v_2 :	[400, 800]
	[3000, 4000] = moyen		[800, 1000]
	[4000, 5500] = fort		[1000, 1200]
	[5500, 8000] = très fort		«manquante»
	«manquante»		

Dans le tableau 8, (0,4; 0,2; 0,3; 0,1; 0) signifie que l'utilisateur/jour w_i , dans l'intervalle de temps I_{10} , prend pour la variable V_1 , 40% de ses valeurs dans l'intervalle [2000, 3000] caractérisant les valeurs faibles, 20% de ses valeurs dans l'intervalle [3000, 4000] caractérisant les valeurs moyennes, ... et 0% de valeurs manquantes.

TABLEAU 8
Tableau des données recodées par vecteurs de fréquences

	V_1			...	V_9		
	I_1	...	I_{10}		I_1	...	I_{10}
w_1			(0,4; 0,2; 0,3; 0,1; 0)				
\vdots							
w_{165}							

Dans cette application, ce dernier type de résumé a été choisi pour recoder les données. Ce choix a été guidé par le fait que cette représentation est celle qui résume le plus fidèlement les données et qui intègre directement les données manquantes. Par contre, elle est plus difficile à lire et à traiter que le codage par intervalles.

On obtient alors un tableau où chaque case contient un vecteur de fréquences qui peut être représenté sous la forme d'histogramme (voir tableau 8).

4. Classification des utilisateurs

Il s'agit ici de trouver des classes d'utilisateurs qui ont les mêmes réactions pour les mêmes environnements, c'est à dire le même comportement.

L'originalité de la méthode présentée ici consiste en la combinaison de deux algorithmes de classification. En effet, dans un premier temps on va classer les utilisateurs/jour simultanément sur les deux groupes de variables (environnement et ambiance), dans un deuxième temps, on utilisera cette classification des utilisateurs/jour pour obtenir une classification des utilisateurs ayant le même comportement.

Cette méthode comprend les trois étapes suivantes :

- classification symbolique hiérarchique des utilisateurs/jour
- description de chaque utilisateur par ses fréquences d'apparition dans chaque classe d'utilisateurs/jour obtenue à l'étape précédente.
- classification hiérarchique des utilisateurs ainsi redécrits.

Première étape : *classification symbolique hiérarchique des utilisateurs/jour*

Le formalisme des objets probabilistes (Diday 91, 95) permet de prendre en compte la particularité du recodage des utilisateurs/jour, cette particularité venant du fait que les utilisateurs sont décrits sur chaque variable, non pas par une valeur unique, mais par un vecteur de fréquences. La classification symbolique hiérarchique sur objets probabilistes nous permet d'obtenir un partitionnement des 165 utilisateurs/jour. Cette méthode est développée dans la thèse de P. Brito (Brito 1991) et implémentée par le logiciel APYROS (Mfoumoune 1994).

Cet algorithme fournit un arbre hiérarchique ou une pyramide, autorisant, dans ce dernier cas, des classes recouvrantes bien que non nécessairement incluses les unes dans les autres.

L'originalité de cette première démarche par rapport aux classifications hiérarchiques classiques tient principalement en ce que :

a) Elle permet de traiter en entrée des objets probabilistes, ce qui est nécessaire car chaque case de notre tableau contient un vecteur de fréquences.

b) Elle fournit directement une description sous la forme d'objets histogrammes qui ont une propriété particulière dite de «complétude» de chaque palier de la hiérarchie ou de la pyramide.

A partir de la hiérarchie obtenue par le logiciel APYROS, sur le tableau des 165 utilisateurs/jours recodés, on a sélectionné la partition en 5 classes qui est la partition la plus intéressante. On obtient ainsi des classes d'utilisateurs/jour qui ont été observés dans le même environnement, avec la même ambiance. Il est donc normal qu'un même utilisateur puisse appartenir à deux classes différentes du fait qu'il est observé sur différents jours, donc dans différents environnements avec des ambiances différentes. On ne peut donc pas en déduire le comportement d'un utilisateur. C'est pourquoi, on a été amené à procéder à une seconde classification.

Deuxième étape

Si à l'issue de la première étape on choisit une partition en K classes, on construit un tableau décrivant les n utilisateurs u_1, \dots, u_n , par leur fréquence d'apparition dans chacune des K classes. On obtient alors un tableau de n lignes et K colonnes.

Ici, les 165 utilisateurs/jours sont partitionnés en 5 classes, et les 9 utilisateurs (voir tableau 3) sont ainsi recodés dans un tableau de 9 lignes et 5 colonnes (voir tableau 9).

TABLEAU 9
*Tableau des utilisateurs recodés par leurs fréquences
d'apparition dans 5 classes*

N° util	C_1	C_2	C_3	C_4	C_5
1	0.910	0	0	0.090	0
2	0.325	0.045	0	0.165	0.465
3	0.333	0	0.166	0	0.500
4	0.333	0	0.666	0	0
5	0.200	0	0.200	0	0.600
6	0.333	0	0.666	0	0
7	0.040	0.195	0.285	0.480	0
8	0.135	0.446	0.142	0.142	0.135
9	0.538	0	0.386	0	0.076

Troisième étape : Classification hiérarchique avec l'indice de WARD

Deux individus ayant une même répartition dans les classes sont des utilisateurs qui pour un même environnement réagissent par une même ambiance. Il s'agit donc de faire une classification du tableau de fréquence des individus pour obtenir une partition des utilisateurs.

Dans cette application, on a mis en œuvre la procédure WARD du logiciel de classification SICLA qui effectue une classification hiérarchique des 9 utilisateurs par la méthode des voisins réciproques avec l'indice de Ward. Cette étape permet de retrouver et de mettre dans la même classe (par classification hiérarchique) les utilisateurs qui se retrouvent régulièrement dans les mêmes classes (obtenues par la première classification symbolique hiérarchique) et qui ont donc le même type de comportement.

L'arbre de classification hiérarchique ainsi obtenu nous permet de déterminer nettement 4 classes d'utilisateurs choisissant les mêmes ambiances pour les mêmes environnements; par exemple la classe 1 contiendra les utilisateurs 2, 3 et 5.

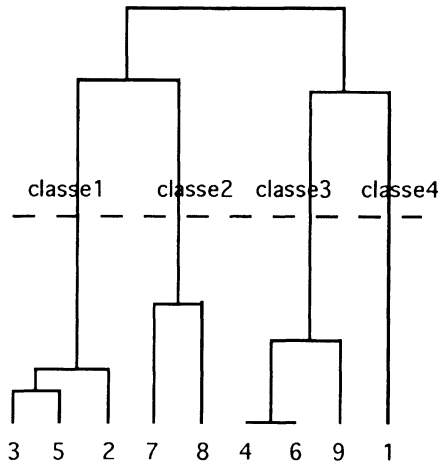


FIGURE 3

Arbre de classification hiérarchique des 9 utilisateurs

Interprétation des classes obtenues

Il est possible de donner une première interprétation des 5 classes d'utilisateurs/jour obtenues par la première classification hiérarchique grâce à la description symbolique des paliers donnée par le logiciel APYROS (Mfoumoune 1994). La description symbolique d'un palier consiste en une liste d'histogrammes, chaque histogramme décrivant les valeurs prises sur une variable, par les individus du palier. En fait, chaque histogramme résulte de l'union probabiliste des histogrammes des individus du palier (Diday 1995).

Une deuxième interprétation experte est alors effectuée à partir des descriptions symboliques des 5 paliers correspondant aux 5 classes choisies. Sur chacune des 5

descriptions symboliques, on regarde si l'ensemble des variables d'environnement ou d'ambiance prennent majoritairement des valeurs faibles, moyennes ou fortes, et on choisit de réinterpréter ces classes respectivement par un environnement ou une ambiance dits FAIBLE, MOYEN ou FORTE.

Les résultats de l'interprétation experte des 5 classes d'utilisateurs/jour sont donnés dans le tableau suivant :

TABLEAU 10
Interprétation des classes d'utilisateurs/jour

	ENVIRONNEMENT		
AMBIANCE	FORT	MOYEN	FAIBLE
FORT	C_3	C_4	C_2
MOYEN		C_1	
FAIBLE			C_5

Par exemple, on lit dans le tableau 10 que la classe C_2 a un environnement faible et une ambiance forte, ce qui veut dire que les individus de cette classe ont été observés dans un environnement faible, et qu'ils ont réagi par une ambiance forte.

En fonction de cette première interprétation experte des classes d'utilisateurs/jour, nous avons pu déduire celle des 4 classes d'utilisateurs. Ainsi, le comportement des utilisateurs d'une même classe est interprété en termes de réactions d'ambiance (faible, fort, moyen) en fonction des environnements possibles (faible, fort, moyen). Pour cela on croise les informations contenues dans les tableaux 9 et 10 de la manière suivante :

Pour chaque classe on calcule l'utilisateur moyen, c'est à dire la moyenne des utilisateurs de la classe. Par exemple le tableau 11 nous donne l'utilisateur moyen de la classe 1 et on sait ainsi comment les utilisateurs de la classe 1 se répartissent dans les 5 classes d'utilisateurs/jours C_1, \dots, C_5 .

TABLEAU 11
Moyenne des fréquences d'appartenance des individus de la classe 1

N° util	C_1	C_2	C_3	C_4	C_5
2	0.325	0.045	0	0.165	0.465
3	0.333	0	0,166	0	0.500
5	0.200	0	0.200	0	0.600
Moyenne	0.286	0.015	0.122	0.055	0.522

Ensuite, à partir du tableau 10 on lit pour chaque type d'environnement (fort, moyen, faible) les classes et les types d'ambiance associés (fort, moyen, faible) et on en déduit par exemple que seuls les utilisateurs/jours de la classe C_3 ont été observés dans un environnement fort, et qu'ils ont réagi par une ambiance forte. Ainsi les utilisateurs de la classe 1 qui appartiennent en moyenne à 12,2% à la classe d'utilisateurs/jour C_3 , ont donc été observés dans un environnement fort et ont toujours réagi par une

ambiance forte. On lit également dans le tableau 10 que seuls les utilisateurs/jour des classes C_4 et C_1 ont été observés dans un environnement moyen. On en déduit que les utilisateurs de la classe 1 qui ont été observés beaucoup plus dans la classe C_1 que dans la classe C_4 ($28,6 \gg 5,5$), réagissent à un environnement moyen presque toujours par une ambiance moyenne.

De même on trouve que les classes d'utilisateurs/jour correspondant à un environnement faible sont les classes C_2 et C_5 , les utilisateurs de la classe 1 ayant été plus souvent observés dans la classe C_5 (52,2%) que dans la classe C_2 (1,5%). On en déduit que dans un environnement faible, les utilisateurs de la classe 1 ont presque toujours réagi par une ambiance faible. On en déduit 4 tableaux résumant les comportements des 4 classes d'utilisateurs :

TABLEAU 12
Interprétation de la classe 1 = {2, 3, 5}

	ENVIRONNEMENT		
AMBIANCE	FORT	MOYEN	FAIBLE
FORT	×		
MOYEN		×	
FAIBLE			×

On lit dans le tableau 12 que les utilisateurs de la classe 1 = {2, 3, 5} réagissent par une ambiance forte à un environnement fort, par une ambiance moyenne à un environnement moyen et par une ambiance faible à un environnement faible.

On construit de même les 3 tableaux décrivant le comportement des 3 autres classes d'utilisateurs :

TABLEAU 13
Interprétation de la classe 2 = {7, 8}

	ENVIRONNEMENT		
AMBIANCE	FORT	MOYEN	FAIBLE
FORT	×	×	×
MOYEN			
FAIBLE			

TABLEAU 14
Interprétation de la classe 3 = {4, 6, 9}

	ENVIRONNEMENT		
AMBIANCE	FORT	MOYEN	FAIBLE
FORT	×	×	
MOYEN			
FAIBLE			

TABLEAU 15
Interprétation de la classe 4 = {1}

	ENVIRONNEMENT		
AMBIANCE	FORT	MOYEN	FAIBLE
FORT		×	
MOYEN			
FAIBLE			

On déduit des 4 tableaux précédents deux types de comportement. Il y a d'abord les utilisateurs qui ne modifient pas leur ambiance, qui correspond alors à l'environnement dans lequel ils sont observés. C'est le cas des utilisateurs de la classe 1 pour qui l'ambiance est toujours du même type que l'environnement. Il y a ensuite les utilisateurs de la classe 2 qui, quelque soit l'environnement, lui associent une ambiance forte. Dans le premier cas les utilisateurs sont passifs face à leur environnement, tandis que dans le deuxième cas, ils sont actifs, mais cette action ne s'adapte pas à l'environnement, puisqu'ils choisissent toujours la même action.

Les classes 3 et 4 correspondent à des individus n'ayant pas été observés dans tous les environnements, on ne peut donc pas en déduire leur comportement, même s'ils semblent passifs.

5. Conclusion

Le recodage du tableau de données temporelles par la méthode présentée précédemment permet de réduire d'une part le nombre de variables de 4747 à 470, d'autre part le pourcentage de données manquantes de 50,3% à 18,5%. De plus, l'utilisation des résumés de valeurs, sur chaque intervalle de temps, permet de réduire le bruit des données initiales. Pour l'application, on a pu identifier, grâce à ce travail, les moments de la journée correspondant à des comportements cohérents d'utilisateurs, soit 10 intervalles de temps.

La classification des utilisateurs nous a permis de déterminer non seulement des classes de comportement d'utilisateurs mais également leur interprétation symbolique. Ainsi, on a déduit deux types de comportement, chaque type donnant pour chaque classe, l'ambiance associée à un environnement donné. Nous travaillons actuellement à l'élaboration de méthodes de validation des différents choix réalisés dans cette première approche : la distance entre objets probabilistes, les classes, le nombre de classes retenues et les règles obtenues.

Remerciements

Nous tenons à remercier le Laboratoire d'Électronique Philips-SAS et en particulier Jean-Philippe Laurent non seulement pour la mise à disposition des données particulièrement intéressantes mais également pour les nombreux conseils, l'aide à la réflexion et la coopération qui nous ont aidées durant les différentes étapes de ce travail.

Bibliographie

- AFIFI A. et ELASHOFF R.M. (1966), "Missing observations in Multivariate Statistics : review of the litterature", J.A.S.A., 61, pp. 595-604.
- BRITO P. (1991), «Analyse de Données Symboliques. Pyramides d'Héritage», Thèse Université Paris IX-Dauphine.
- DIDAY E. (1991), *Des objets de l'Analyse des données à ceux de l'Analyse des connaissances*, Induction symbolique et numérique à partir de données, eds Y.Kodratoff, E. Diday, cepadues.
- DIDAY E. (1995), "Probabilist, possibilist and belief objects for knowledge analysis", Annals of Operations Research. 55, pp. 227-276.
- FISHER W.D. (1958), *On grouping for maximum homogeneity*, J.A.S.A 53, pp. 789-98.
- JENSEN R.E. (1969), *A dynamic programming algorithm for cluster analysis*, Operat. Research Vol. 12.
- LECHEVALLIER Y. (1990), *Recherche d'une partition optimale sous contrainte d'ordre totale*, Rapport de recherche INRIA 1247.
- MFOUMOUNE E. (1994), «Classification pyramidale : aspect algorithmique», 2^e journées «symbolique-numérique», Orsay.