Editorial

The 10th Meeting of the French-speaking Classification Society (SFC) took place in Neuchâtel. The University of Neuchâtel and its Statistics Group hosted this reunion between 10th and 12th September 2003. During the meeting more than fifty papers were presented.

A fruitful state of the art in clustering methods, classification algorithms, correspondence analysis as well as factor analysis, demonstrated the dynamic research which is carried out by the members and the friends of the French-speaking Classification Society.

Furthermore, the numerous applications of classification algorithms in various fields of real life problems proved once more again the capacity to capitalise on the existing methods in a creative and rigorous way.

The present issue of *Student* could not evidently include all of the papers presented during the 10th reunion of the French-speaking Society of Classification. Five refereed and original papers are given in this issue as a sample of the interesting topics presented during this meeting.

The Editor

Student (2004), Vol. 5 No. 1, 1-12

Some Measures of Agreement Between Close Partitions

Genane Youness

CEDRIC CNAM
BP 11 - 4661
Beirut, Lebanon

Gilbert Saporta

Chaire de Statistique Appliquée
CEDRIC, CNAM, 292 rue Saint Martin
75003 Paris, France

Abstract: In order to measure the similarity between two partitions coming from the same data set, we study extensions of the RV-coefficient, the kappa coefficient proposed by Cohen (in case of partitions with same number of classes), and the D2 coefficient proposed by Popping. We find that the RV-coefficient is identical to the Janson and Vegelius index. We compare the result coming from kappa's coefficient to the ordination given by correspondence analysis. We study the empirical distribution of these indices under the hypotheses of a common partition. For this purpose, we use data coming from a latent profile model to formulate the null hypothesis.

Key words : RV, Janson and Vegelius index, Cohen's Kappa, Popping' D2, Correspondence analysis, latent class

1 Introduction

The problem of measuring the agreement between two partitions of a same data set has attracted interest and reappears continually in the literature of classification.

The most useful agreement measured proposed by Rand (1971) has been rediscovered and modified by [2] . Based on the comparison of object triples, a measure of partition correspondence was introduced by [4] . A generalized Rand-index method for consensus clustering was proposed by [3] for finding an amalgamated clustering

of a set of contributory partitions. By using mathematical and statistical aspects in the standardization of the comparison coefficients, [8] shows how to take into account the relational constraint, which results from the partition structure.

A simple way to compare partitions is to study the empirical distribution of a measure of agreement under some null hypothesis. It is necessary to define measures of agreement between partitions, before testing the agreement. In previous papers [13, 14, 15] , we have examined the following indices: Rand, Mc Nemar, and Jaccard. In this paper, we present new approaches to find the similarity between clustering. The RV-coefficient introduced by Robert and Escoufier [12] is proposed and written in term of paired comparisons. We prove that RV-coefficient is identical to the JV index [9] .

The kappa coefficient proposed by Cohen [1] is another way to measure the agreement between partitions with equal number of classes. Kappa can be used only in situations where categories are specified in advance, which is not the case of partitions where the labels of the classes are arbitrary: for that, we identify the permutation of classes of one of the partitions by maximizing the kappa values. We compare the optimal permutation with the ranking given by correspondence analysis: the results are the same.

Finally we also study the D_2 index proposed by Popping [10] . This less known index is based on comparison of pairs of subjects. We find the empirical distribution of this agreement measure when the partitions only differ by chance.

2 Notations

Let P_1 and P_2 be two partitions (or two categorical variables) of the same subjects with p and q classes. If K_1 and K_2 are the disjunctives $n \times p$ and $n \times q$ tables and N the corresponding contingency table with elements n_{ij}, we have: $N = K_1'K_2$.

Each partition P_k is also characterized by the $n \times n$ paired comparison table C^k with general term $c^k_{ii'}$:

$$c^k_{ii'} = \begin{cases} 1 & \text{if } i \text{ and } i' \text{ are in the same class of } P_k \\ \\ 0 & \text{otherwise} \end{cases}$$

Note that $c^k_{ii} = 1$, for every $i \in \mathbb{N}$. We have $C_1 = K_1K_1'$ and $C_2 = K_2K_2'$. Given n subjects, $n(n-1)/2$ pairs of subjects can be compared. When both partitions assign pairs of subjects to the same classes, we consider this as an agreement.

$P_1 \backslash P_2$	**Same class**	**Different class**
Same class	**a** Agreement (same)	**c** Disagreement
Different class	**d** Disagreement	**b** Agreement (Different)

Table 1: The four cases

3 RV-coefficient

Robert and Escoufier [12] have derived a measure of similarity of two configurations, called RV-coefficient or the vector correlation coefficient; it allows to measure the similarity between two data tables X_1 and X_2 of the same observations by comparing the scalar product between individuals associated to the two tables. If the scalar matrices product $X_i X_i'$ is noted as W_i with dimension $n \times n$, the RV-coefficient is defined as:

$$RV(X_1, X_2) = \frac{\text{tr } (W_1 W_2)}{\text{tr } (W_1^2)\text{tr } (W_2^2)}.$$

When applied to the disjunctive tables associated to two partitions, we find:

$$RV(P_1, P_2) = \frac{\text{tr } (C^1 C^2)}{\text{tr } (C^1)^2 \text{tr } (C^2)^2} = \frac{\sum_{i \neq i'} \left(c^1_{ii'} - \frac{1}{p}\right)\left(c^2_{ii'} - \frac{1}{q}\right)}{\sqrt{\sum_{i \neq i'} \left(c^1_{ii'} - \frac{1}{p}\right)^2 \sum_{i \neq i'} \left(c^2_{ii'} - \frac{1}{q}\right)^2}}. \quad (1)$$

A high value of RV may lead to the conclusions that partitions are almost identical.

Lazraq, A. and Cleroux, R. [6] have proposed a test for the null hypothesis that the theoretical (population) RV is zero, but only in the case of numerical data, which cannot be applied to indicator variables.

3.1 JV index

The JV index is a measure of association proposed by Janson and Vegelius [9] . Its initial expression is:

$$JV(P_1, P_2) = \frac{pq \sum_u \sum_v n_{uv}^2 - p \sum_u n_{u\cdot}^2 - q \sum_v n_{\cdot v}^2 + n^2}{\sqrt{[p(p-2) \sum_u n_{u\cdot}^2 + n^2]\,[q(q-2) \sum_v n_{\cdot v}^2 + n^2]}}.$$

It has been proved that, in terms of paired comparisons:

$$JV(P1, P2) = \frac{\text{tr } (C^1 C^2)}{\text{tr } (C^1)^2 \text{tr } (C^2)^2} = \frac{\sum_{i,i'} \left(c^1_{ii'} - \frac{1}{p}\right)\left(c^2_{ii'} - \frac{1}{q}\right)}{\sqrt{\sum_{i,i'} \left(c^1_{ii'} - \frac{1}{p}\right)^2 \sum_{i,i'} \left(c^2_{ii'} - \frac{1}{q}\right)^2}}.$$

Idrissi [5] used this formula to study the probability distribution of JV under the hypothesis of independence. If the k classes are equiprobable, one finds that $\sum_{i \neq i'} \left(c^1_{ii'} c^2_{ii'}\right)$ follows a binomial distribution $B(n(n-1), 1/k^2)$. Idrissi claims that the JV index between two categorical variables with k equiprobable modalities has an expected value equal to:

$$E(JV) = \frac{k-1}{n}.$$

The RV-coefficient and the JV index are identical, in terms of paired comparisons.

4 Cohen's Kappa

The kappa coefficient for computing nominal scale agreement between two raters was first proposed by Cohen [1] . He defined his index as "the proportion of agreement after chance agreement is removed from consideration".

$$\kappa = \frac{P_o - P_e}{1 - P_e},$$

where

$$P_o = \frac{1}{n}\sum_{i=1}^{k} n_{ii} \qquad \text{and} \qquad P_e = \frac{1}{n^2}\sum_{i=1}^{k} n_{i.}n_{.i}.$$

In this formula, P_o is the amount of observed agreement. In case of independence, given the marginals, the expected amount of agreement is P_e. Therefore, a correction is made for this amount agreement. In order that the index assumes the value 1 in case of perfect agreement, the correction is also made in the denominator. It measures the deviation of objects on the diagonals in the agreement table.

We use the equivalent formula of the kappa coefficient to compute the agreement between two partitions having the same number of classes:

$$\kappa = \frac{n\sum_{i=1}^{k} n_{ii} - \sum_{i=1}^{k} n_{i.}n_{.i}}{n^2 - \sum_{i=1}^{k} n_{i.}n_{.i}}.$$

An important condition to use the kappa coefficient is for the situation in which the identity of classes per observations is known in advance. In our case, when we compare partitions coming from the same data set and found by the clustering methods, the numbering of classes is totally arbitrary: so we propose to identify the classes of the partitions from the maximum value of κ. We permute the columns or rows in the cross table and we calculate κ at each time; we take the numbering of permutation with the maximum κ.

Another method to find the optimal ordering is to use the order of the categories of both variables given by the first factor in the simple corresponding analysis. This method maximizes the weight of the diagonal of the cross table by permuting their lines and columns.

We compare the result found by the simple corresponding analysis to our method. Often, the same value of κ can be observed.

5 Popping's D2

The D_2 index was proposed by Popping [10] and is based on the same principles of kappa coefficient. It is used to study the agreement between the nominal classifications by two judges who independently categorize the same subjects in case that the categories are not known in advance. The D_2 index is based on the comparison of pairs of subjects and start from the situation of same agreement (Table 1).

The index contains a correction for agreement that can be expected on the basis of chance given the marginals of the original classification.

Under the null hypothesis of independence, the D_2 is a transformation of the Russel and Rao's coefficient

$$\frac{a}{a+b+c+d}.$$

The D_2 index is defined as:

$$D_2 = \frac{D_o - D_e}{D_m - D_e}$$

where

$$D_o = \frac{\sum_{i=1}^{p}\sum_{j=1}^{q} n_{ij}(n_{ij}-1)}{n(n-1)},$$

$$D_e = \frac{2\sum_{i=1}^{p}\sum_{j=1}^{q} c_{ij}}{n(n-1)},$$

$$c_{ij} = g_{ij}(h_{ij} - 0.5g_{ij} - 0.5) \quad \text{where } h_{ij} = \frac{n_{i.}n_{.j}}{n} \quad \text{and } g_{ij} = \text{integer } (h_{ij}),$$

$$D_p = \frac{\sum_{i=1}^{p} n_{i.}(n_{i.}-1)}{n(n-1)},$$

$$D_q = \frac{\sum_{i=1}^{p} n_{.j}(n_{.j}-1)}{n(n-1)},$$

and

$$D_m = \max(D_p, D_q).$$

Note that in D_2 one considers De as a reasonable minimum [10] , so for situations where we use the g_{ij} in D_e, we have no empirical proof to get an expected value smaller than the minimum. This is in favor of g_{ij} instead of the fractional number h_{ij}. However, this results in the fact that D_e is biased: the results are too high. The consequence for D_2 is that the index will take conservative values.

The D_2 index has been compared to the kappa coefficient to the JV index, in the particular following case:

Categories	+	−	Total
+	u	$v-u$	v
−	$v-u$	u	v
Total	v	v	$2v$

Table 2: The particular case found by Popping.

Popping obtained the results:

$$\kappa = \frac{2u-v}{v},$$

$$D_2 = \left[\frac{2u-v}{v}\right]^2 = JV.$$

In the general case we found that there are a strong correlation between these indices in the simulation study.

We find a relation between the Jaccard's index known as measure of similarity between objects described by presence-absence attributes [14] to the D_2 coefficient, we find the following relation:

In case of Integer$(hij) = h_{ij}$

$$D_2 = \frac{(c_n^2 - b)J}{\max(a+c, a+d) - C_n^2}$$

where

$$J = \frac{a}{a+c+d} \qquad \text{and } C_n^2 = \frac{n(n-1)}{2}.$$

6 Empirical distribution

We use the algorithm presented in [13] for finding the empirical distribution of the indices of association when the two partitions only differ by chance. The difficulty consists in conceptualising a null hypothesis of "identical" partitions and a procedure to check it. Note that departure from independence does not mean that there exists a strong enough agreement.

Now we have to define the sentence "two partitions are identical". Our approach consists to say that the units come from a same partition, where the two observed partitions are noised. The latent class model is well adapted to this problem for getting partitions. Note that Green and Krieger [3] have used it in their consensus partition research. More precisely, we use the latent profiles models because we have numerical variables.

For getting "near-identical partitions", we suppose the existence of a common partition for the population according to a latent profile model. The basic hypothesis is the independency of observed variables conditional to the latent classes that gives similar partitions from one or another groups of variables:

$$f(x) = \sum_k \pi_k \prod_j f_k(x_j/k).$$

The π_k are the class proportions and x is the random vector of observed variables, where the component x_j are independent in each class.

We use here the latent class model to generate the data and not to estimate parameters. Once having choosen the number of classes, we first generate their frequencies from a multinomial distribution with probabilities π_k, and then we generate observations in each class according to the local independence model in other words a normal mixture model with independent components in each class. Then we split arbitrarily the p variables into two sets and perform a partitioning algorithm on each set. The two partitions should thus differ only by random. We calculate the indices for the two partitions: we repeat the procedure N times to find a sampling distribution of kappa, RV (or J index), and D_2. Our algorithm has the following steps:

1. Generate the sizes n_1, n_2,..., n_k of the clusters according to a multinomial distribution $M(n; \pi_1 \dots \pi_k)$.

2. For each cluster, generate n_i values from a random normal vector with p independent components.

3. Get two partitions: P_1 of the units according to the first p_1 variables and P_2 according to the last $p - p_1$ variables.

4. Compute association measures (for RV, J and D_2) for P_1 and P_2.

4'. Permute the columns of the cross table (P_1, P_2) to find the maximum value of kappa, so the numbering of classes.

5. Repeat the procedure N times.

7 Numerical Applications

We applied the previous procedure with 4 equiprobable latent classes, 1000 units and 4 variables. We obtained the two partitions P_1 (with X_1 and X_2) and P_2 (with X_3 and X_4) with 4 classes by the k-means methods. We present only one of our simulations (performed with S+ software).

Class 1	Class 2	Class 3	Class 4
X1 N(1.2,1.5)	X2 N(4,2.5)	X3 N(7,3.5)	X4 N(10,4.5)
X1 N(-2,1.5)	X2 N(-4,2.5)	X3 N(-6,3.5)	X4 N(-10,4.5)
X1 N(-5,1.5)	X2 N(-10,2.5)	X3 N(-13,3.5)	X4 N(-20,4.5)
X1 N(-8,1.5)	X2 N(-15,2.5)	X3 N(-20,3.5)	X4 N(-30,4.5)

Table 3: The normal mixture model.

The following figure shows the spatial distribution of one of the 1000 iterations for the two partitions P_1 and P_2.

The cross table of the two partitions in one of the simulations is represented as follows:

1	**2**	**3**	**4**
248	0	0	2
1	198	27	9
2	6	43	202
0	58	192	12

Table 4: Cross tabulation of P_1 and P_2 for one of the simulation.

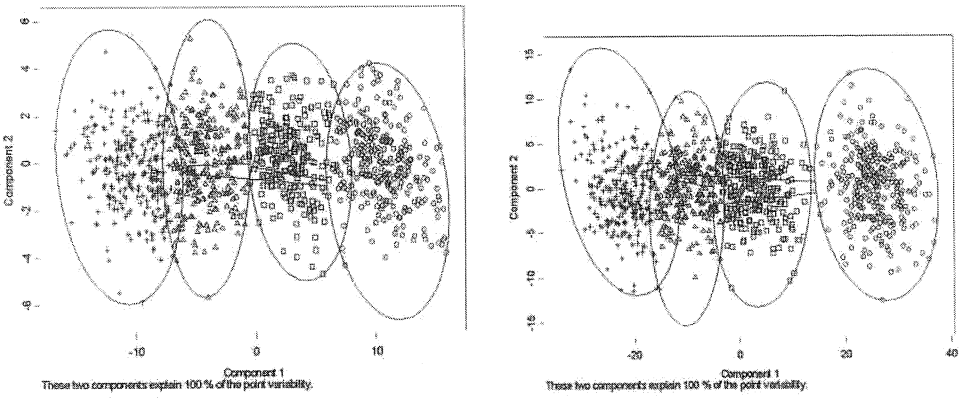


Figure 1. The first two principal components of one of the 1000 samples for the P1 and P2

We find a value of the kappa coefficient equal to 0.335, and a value of JV index (or RV) equal to 0.648. The value of the D_2 index is equal to 0.647.

To identify the labels of classes of P_2 to those of P_1, we permute the columns (4! permutations) of the cross table: The maximum value of the kappa coefficient is 0.787, and the numbering of column's permutation is 1,2,4,3. So the table becomes:

1	**2**	**3**	**4**
1	2	4	3
248	0	2	0
1	198	9	27
2	6	202	43
0	58	12	192

Table 5: Cross tabulation with the new permutation of columns.

Another way of getting the "optimal" ordering is by means of correspondence analysis: categories of both variables are ordered according to their coordinates along the first axis.

198	27	9	1
58	192	12	20
6 43	202	2	
0	0	2	248

Table 6: Order according to the first factor of CA.

If we calculate the kappa coefficient of this last table, we find the value 0.787. Here correspondence analysis ordering gives the same result as our method. The distribution of maximum kappa coefficient values is presented in Figure 3:

With the same choice of normal independent mixtures variables, we find that the kappa coefficient varies between 0.4 and 0.875. Its means is 0.82.

By simulation, the value of the JV index varies between 0.4 and 0.7. The most frequent value is 0.63 and the mean is equal to 0.617. (Fig. 4)

Under the hypothesis of independence $E(JV) = 0.003$, and with 1000 observations, independence should have been rejected for $JV > 0.617$ at 5% level. The 5% critical value is much higher than the corresponding one in the independence case. It shows that departure from independence does not mean that the two partitions are close enough.

The D_2 index takes its values between 0.3 and 0.7 with a mode equal to 0.625. The distribution has a mean equal to 0.610. Bimodality is due to the use of k-means, which gives a local optima [15] .

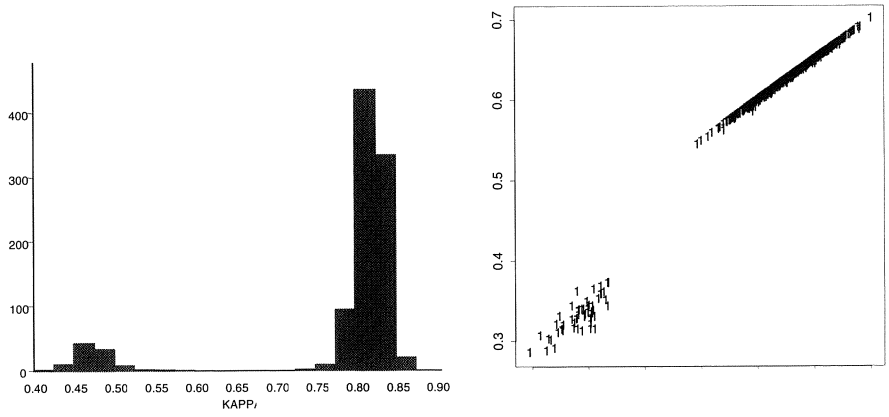


Figure 2. Distribution of kappa for 1000 individuals and 1000 iterations, partitions with 4 classes. The scatter plot of JV against D_2 in the 1000 iterations.

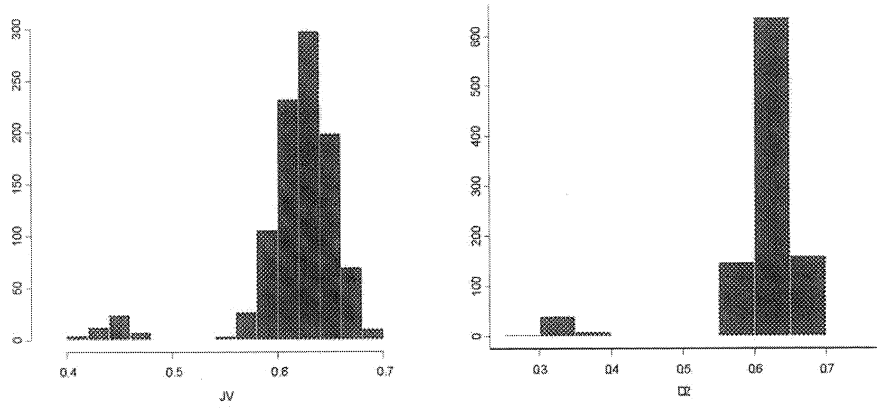


Figure 3. The JV- index distribution and D_2 for the partitions of 4 classes in 1000 iterations.

There is a strong correlation between the JV and D_2 equal to 0.983. So we can say that the two indices give same result in comparing partitions (Fig. 3).

Under the null hypothesis of close partitions, all indices have values around the mean, which is close to 0.6 so that we could say that the two partitions P_1 and P_2 are close enough.

8 Conclusion

In this paper, we have proved the identity between the RV-coefficient and the JV-index for comparing two partitions. A latent class model has been used to solve the problem of comparing close partitions and three agreement indices: JV (or RV), kappa and D_2 have been studied.

The kappa coefficient allows to test the similarity between two partitions after permutation and when we they have the same number of classes. We compare the optimal permutation with the order given by correspondence analysis: By simulation, it gives often the same results.

The D_2 index seems useful for comparing the classification of two partitions in the case that the identification of classes is not known in advance. D_2 gives stress on pairs in the same class of both partitions, as Jaccard's index. We found a relation between them.

Since we found a strong correlation between the JV index and the D_2 in our simulation, we can deduce that JV (or RV) and D_2 lead us to the same result in comparing partitions. The distributions of these proposed indices have been found

very different from the case of independence and are bimodal. The bimodality might be explained by the presence of local optima in the k-means algorithm.

References

[1] Cohen, J. (1960). A Coefficient of Agreement for Nominal Scales., *Educ. Psychol. Meas.*, 20, 27-46.

[2] Fowlkes, E.B. and Mallows, C.L. (1983). A Method for Comparing Two Hiearchical Clusterings, *Journal of American Statistical Association*, 78, 553-569.

[3] Green, P. and Krieger, A. (1999). A Generalized Rand-Index Method for Consensus Clustering of Separate Partitions of the Same Data Base, *Journal of Classification*, 16, 63-89.

[4] Hubert, L., and Arabie, P. (1985). Comparing Partitions, Journal of Classification, 2, 193-218.

[5] Idrissi, A. (2000). Contribution à l'Unification de Critères d'Association pour Variables Qualitatives, *Thèse de doctorat de l'Université de Paris 6.*

[6] Lazraq, A. and Cleroux R. (2002). Inférence Robuste sur un Indice de Redondance, *Revue de Statistique Appliquée*, 4, 39-54.

[7] Lerman, I.C (1973). Etude Distributionnelle de Statistiques de Proximité entre Structures Finies de Memes Types; Application à la Classification Automatique, Cahier 19, *Bureau Universitaire de Recherche Opérationelle, Institut de Statistique des Universités de Paris.*

[8] Lerman, I.C (1988). Comparing Partitions (Mathematical and Statistical Aspects), *Classification and Related Methods of Data Analysis*, H.H Bock Editor, 121-131.

[9] Janson S., Vegelius J. (1982). The J- Index as a Measure of Association For Nominal Scale Response Agreement, *Applied psychological measurement*, 16, 243-250.

[10] Popping, R. (1983). Traces of Agreement. On the Dot-Product as a Coefficient of Agreement, *Quality and Quantity*, 17, 1-18.

[11] Popping, R. (1992). Taxonomy on Nominal Scale Agreement, *Groningen; iec ProGAMMA*, 1945-1990.

[12] Robert, P. and Escoufier, Y. (1976). A Unifying Tool for Linear Multivariate Statistical Methods: The *RV*-Coefficient, *Appl. Statist.*, 25, 257-265.

[13] Saporta G. and Youness, G. (2001). Concordance entre Deux Partitions: Quelques Propositions et Expériences, in *Proceedings SFC 2001, 8èmes rencontres de la Société Francophone de Classification, Pointe à Pitre.*

[14] Saporta G. and Youness, G. (2002) Comparing Two Partitions: some proposals and Experiments, *Proceedings in Computational Statistics edited by Wolfgang Härdle, Physica- Verlag*, Berlin.

[15] Saporta G. and Youness, G. Une Méthodologie Pour la Comparaison de Partitions, *Revue de Statistique Appliquée*, to appear.

Student (2004), Vol. 5 No. 1, 13-28

Binary Decision Trees for Interval and Taxonomic Variables

Chérif Mballo

ESIEA Recherche
38, Rue des Docteurs Calmette et Guérin
F-53000 Laval, France
E-mail: mballo@esiea-ouest.fr

Edwin Diday

LISE-CEREMADE
Place du Maréchal de Lattre de Tassigny
F-75775 Paris 16 ^{ieme}, France
E-mail: diday@ceremade.dauphine.fr

Mounir Asseraf

ESIEA Recherche
38, Rue des Docteurs Calmette et Guérin
F-53000 Laval, France
E-mail: asseraf@esiea-ouest.fr

Abstract: Decision trees have been well studied, with applications in such fields as pattern recognition, taxonomy, expert systems, etc. With the advances in technology, data sets often contain a very large number of observations. The approach of the symbolic objects makes it possible to reduce the size of the data to be treated by transforming the initial classical variables into variables called symbolic variables. The algebraic structure of these variables forces to adapt metric ones to be able to study them. In particular, in the decision trees, it would be interesting to propose the metric in order to extend the mechanism of the construction of these trees. Our contribution in this paper is to adapt the criterion of Kolmogorov-Smirnov to the type of interval and taxonomic variables. We present an example to illustrate this approach.

Key words : Symbolic data analysis, Interval and taxonomic orders, traversing and decision trees, Kolmogorov-Smirnov criterion.

1 Introduction

In the last few years, the extension of the classical data analysis methods to new objects with more complex structure ([4]) has been an investigated topic. The aim of symbolic data analysis ([7]) is to consider the data's particularities in the treatment methods without losing information.

In the case of partition by binary decision trees ([5]), the predictors are usually quantitative or qualitative. A variety of test selection metrics have been proposed, including gain and gain ratio ([14]), Gini ([5]). The Kolmogorov-Smirnov (KS) criterion ([17] ; [1]) require that individuals have to be ordered by their description (variable's values). This metric has been proposed by Friedman ([11]) and it is used as a test selection metric (splitting criterion or attribute selection metric) to build a decision tree induction. We adapt this criterion to symbolic variables with interval or taxonomic data ([2] ; [3]). Its advantage is that it gives the best cutting of each predictor in order that the two probability distributions associated to the two prior classes has the largest KS criterion value. We have been interesting to different possible orders of the predictors. The principal problem is to find a total order in interval or taxonomic values. We dispose a symbolic data table with p variables : one variable to explain and the $(p-1)$ remaining variables are the predictors. In the beginning, the variable to explain is not partitioned. We transform this symbolic variable to explain to a class variable ([8]) with the Fisher algorithm ([10]) to obtain the prior classes. All the variables are interval or taxonomic type. So our application is applied in the case where all the predictors are interval type (for the case where the type of the predictors is taxonomic, see [3]). The prior classes can be from supervised classification.

2 Ordering

2.1 Ordering of interval's variable

We denote the set of closed real intervals by $\Im$. Thus, if $x \in \Im$, then $x = [l(x), r(x)]$ for some real numbers $l(x)$ ("l" as left bound), $r(x)$ ("r" as right bound) such that $l(x) \leq r(x)$.

Taken two intervals $x = [l(x), r(x)]$ and $y = [l(y), r(y)]$, by $x = y$, we mean, of course, that $l(x) = l(y)$ and $r(x) = r(y)$.

Let Ω the set of individuals and α, β two real numbers with $\alpha \leq \beta$. An interval variable Y is an application defined by:

$$Y : \begin{pmatrix} \Omega & \rightarrow & \Im \\ w & \rightarrow & Y(w) = [\alpha, \beta] \end{pmatrix}$$

The description of each individual is an closed real interval.

For studying an order relationship on $\Im$, it is important to distinguish two different situations: the cases in which the interval are disjoint or non disjoint.

Some authors ([9] ; [13]) have been interested in interval orders. Taken two intervals $I = [a; b]$ and $J = [c; d]$, I is considered "before" J if and only if $b < c$. We see that this ordering method is not "true" when intervals are non disjoint. However, the case of non disjoint intervals have been broached by Tsoukias but in the preference modelling area ([15] ; [16]). These authors define three binary relations P (strict preference), Q (weak preference) and I (indifference). With a finite set A, they define the necessary and sufficient conditions to associate, for each element of A, an interval in such a way that:

- if one interval is "completely to the right" of another interval, we obtain the relation P;
- if one interval is "included" into another interval, we obtain the relation I;
- if one interval is " to the right" of another interval but their intersection is not empty, we obtain the relation Q.

We broach this problem of non disjoint intervals but in another direction that permit us to define two relations and each of them is a total interval order.

2.1.1 Disjoint intervals

Let D be the subset of $\Im$ composed of all intervals such that : $x_i \cap x_j = \phi \quad \forall \quad i \neq j$.

Mathematically, the following relation R on D may be defined as follows: $xRy \Leftrightarrow r(x) < l(y)$.

The relation R is transitive and anti-reflexive, so it is a total order on D.

It is clear that, when xRy, the interval x has a position which is *"totally before"* the interval y, this means that each element in x is less than anyone of the set y.

For example, for the case considered, if we want to order two intervals x and y with respect to a time variable, the interval x will be *"totally before"* the interval y because x *"finishes before"* y *"starts"*.

2.1.2 Non-disjoints intervals

Let us consider the case in which two non-disjoint intervals have to be ordered. Different situations may occur according to the position of both the bounds of the considered intervals.

- *Ordering " using lower bounds"*: two possibilities exist:

(i) lower bounds are distinct: the ordering of the intervals is related to the position of the lower bounds;

(ii) lower bounds are equal: the ordering of the intervals is related to the position of the upper bounds.

Mathematically, the following relation I on $\Im$ may be defined as follows:

if $l(x) \neq l(y)$, then $xIy \Leftrightarrow l(x) < l(y)$; if $l(x) = l(y)$, then $xIy \Leftrightarrow r(x) < r(y)$.

The relation I is transitive and anti-reflexive, so it is a total order on $\Im$.

In conclusion, when two intervals x and y are in relation I to one another, for example xIy, we will say that the interval x is *"almost before"* the interval y with respect to I. It is important to notice that the statement *"almost before"* is used just to point out that the intervals are non disjoint and not every element in x is less than or equal to any element in y. For the interval x, we may say only that there is at least one element which is less than or equal to any element in y.

Considering again the case of a time variable, an interval x is *"almost before"* an interval y if x *"starts before"* y but it doesn't matter which one *"finishes first"* ; on the contrary, in the case in which x and y *"start"* at the same time, the interval which *"finishes first"* will be considered *"almost before"* the other one.

- *Ordering "using upper bounds"*: two possibilities exist:

(i) upper bounds are distinct: the ordering of the intervals is related to the position of the upper bounds;

(ii) upper bounds are equal: the ordering of the intervals is related to the position of the lower bounds.

Mathematically, the following relation S on $\Im$ may be defined as follows:

if $r(x) \neq r(y)$, then $xSy \Leftrightarrow r(x) < r(y)$; if $r(x) = r(y)$, then $xSy \Leftrightarrow l(x) < l(y)$.

The relation S is transitive and anti-reflexive, so it is a total order on $\Im$.

In this case, when xSy, we will say that the interval y is *"almost after"* the interval x with respect to S. In analogy to the relation I introduced before, also here it is necessary to use the statement *"almost after"* to point out that the intervals are non disjoint and not every element in y is greater than or equal to any element in x. In the interval y, there is at least one element which is greater than or equal to any element in x.

In the case of a time variable, an interval y is *"almost after"* an interval x if y *"ends before"* x but it doesn't matter which one *"starts first"* ; on the contrary, in the case in which y and x *"end"* at the same time, the interval which *"starts after"* will be considered *"almost after"* the other one.

Let us analyze the following four cases which refer to the possible different positions of the intervals to be ordered. The aim is to understand well the difference between I and S in order to give to the reader the instruments to decide which order is more suitable for treating his own data.

case 1: $l(x) = l(y)$ and $r(y) < r(x)$

$l(x)$[————————-] $r(x)$

$l(y)$[———————-] $r(y)$

yIx " y is *almost before* x "
ySx " x *is almost after* y "

case 2: $l(x) < l(y)$ and $r(y) < r(x)$

$l(x)$[————————]$r(x)$

$l(y)$[——] $r(y)$

xIy " x *is almost before* y "
ySx " x *is almost after* y "

case 3: $l(x) < l(y)$ and $r(y) = r(x)$

$l(x)$[————————] $r(x)$

$l(y)$[———] $r(y)$

xIy " x is *almost before* y "
xSy " y *is almost after* x"

case 4: $l(x) < l(y)$ and $r(x) < r(y)$

$l(x)$[————————]$r(x)$

$l(y)$[————] $r(y)$

xIy " x is *almost before* y "
xSy "y *is almost after* x"

It is very important to point out that, considering two intervals x and y not strictly included one into another as in cases 1, 3 and 4, if the interval x is "*almost before*" the interval y with respect to the relation I, then y is "*almost after*" x with respect to the relation S. In other words, I and S create the same relation of "*precedence*" between two intervals.

On the contrary, when the intervals are strictly included one into another as in case 2, it comes out that x is "*almost before*" y with respect to the relation I, but x is also "*almost after*" y with respect to S. This is the main difference between the two orderings I and S.

For example, in the case of a time variable, ordering two intervals x and y of case2 by I, we will say that x, which "*starts first*", "*precedes*" y. On the contrary, ordering the intervals by S, we will say that y, which "*finishes first*", "*precedes*" x.

Taking into account this difference between orderings I and S, the reader has to decide which one is more suitable for his own data.

In conclusion of this part, an alternative simplest way to order intervals is to take into account the only position of the centers.

2.2 Ordering of taxonomic variable

A taxonomic variable is an organized variable in tree expressing many levels of generality (Figure 1, Figure 2). The modalities of the level k of a taxonomic variable are grouped at the level $(k + 1)$.

A taxonomic variable ([7]) is an application of the set of the individuals in the set of ordered values (ordered totally or partially). The children of a given node are organized according to an increasing "*preference*" order (left to right). The observation domain of a taxonomic variable has a hierarchical structure. A taxonomic variable is

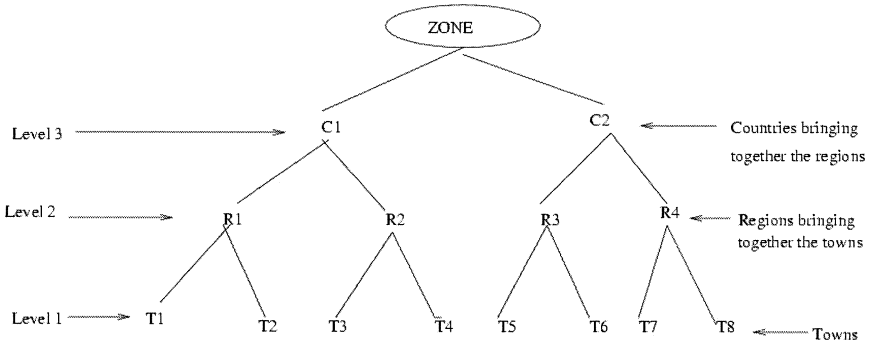


Figure 1. Variable "ZONE"

also called hierarchical variable or a structured variable. For example the domain of the variable "AGE" (Figure 2) is: {Baby, Child, Adolescent , Adult, Old man }.

Here the root corresponds to the name of the taxonomic variable, so it is not a value (description) taken by an individual. The description of an individual is a node of the taxonomic variable. If each node has at most two children, this taxonomy is called binary taxonomy. In the case where a node can have more than two children (n children where n is a natural integer), this taxonomy is called n-ary taxonomy. The problem is to order the individuals by their description to adapt Kolmogorov-Smirnov criterion. To study this problem, we use the principle of traversing trees (*inorder traversal* called *symmetric order* and *level-order traversal*) and binary search trees ([12] ; [6]). The question is how to traverse the taxonomic variable: how to systematically visit every node to obtain an order of the nodes. Any two nodes always have a parent node (at least the root). Let x and y be two nodes of the taxonomic variable. In the following of this paragraph , $x \Re y$ means that the node x "*is before*" the node y. This relation will be defined in various ways according the order by the depth (*inorder traversal* or *symmetric order*) or the order by the widthwise (*level-order traversal*).

2.2.1 Ordering using "inorder traversal or symmetric order"

We consider first of all the case of binary taxonomy. The traversing of the taxonomy, in the aim to find a total increasing order of the individuals by their description, will consist of exploring successively the left and the right subtree in the following way:

- visit first the left subtree (or left child);
- visit the root of this left subtree (or parent node of this left child);
- and, in the end, visit the right subtree (or the right child) in the same way.

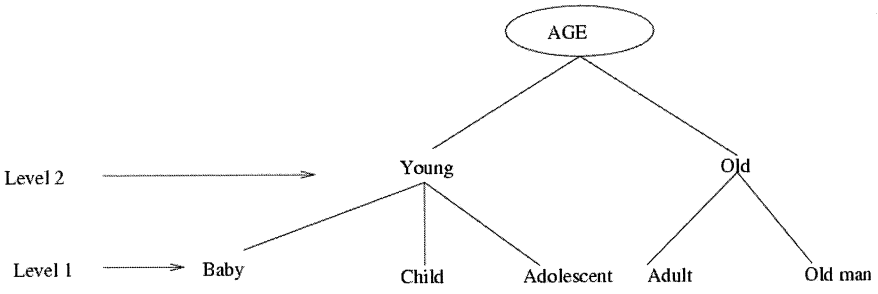


Figure 2. Variable "AGE"

This treatment will respect the following known property of the binary search trees: *each node of the left subtree of a given node has a lower ordering number than it and each node of the right subtree of this given node has a highest ordering number than it.* The principle of the numbering is as follows: we start at the root (without number this root because it corresponds at the name of the variable so it is not a description) and we continue to follow the left splitting until the last node without a child. We number this node and it is thus the first numbered node. We number its parent node and after we go to the right child applying the same principle (the right child can be a splitting). Then we return to the parent node of this preceding parent node and we apply again the same principle. The order obtained by this traversing tree is a total order. This numbering is a bijection ([6]) that assigns each number a unique node. The individuals order by their description will be the one obtained by the numbering of the nodes. In the case where two individuals have the same description (thus corresponding to a unique number for these identical descriptions), their order of the initial data table is maintained at the sorted data table. We illustrate this traversing tree at the figure 3. This principle permits us to define the relation $\Re$ as follows:

$x \Re y \Leftrightarrow$ *Number(x)*< *Number(y)* where Number(x) indicates the number obtained at traversing tree (*inorder traversal* or *symmetric order*).

In the case of n-ary taxonomy, the principle consists of choosing, among the m children nodes of the considered node (with $m > 2$), the p first children from the left as left group child (the $(m-p)$ form then the right group child). After this choice, we apply the same principle that the binary taxonomy on the group child.

The relation $\Re$ is a total order (anti-reflexive and transitive).

2.2.2 Ordering using "level-order traversal"

In this part, we define the levels of the taxonomy variable (from the low). The first level (Level 1) corresponds to the most low node without child. After we go to the following level and so on until the root. But the level corresponding to the root is not numbered because the root is not a description. We illustrate this traversing tree (*level-order traversal*) at the Figure 4. This principle permits us to define the relation $\Re$ as follows:

- if two nodes x and y are at the same level, then:
  $x \Re y \Leftrightarrow$ *x come from the left splitting of the common parent node with y.*
- if two nodes x and y are not at the same level, then: $x \Re y \Leftrightarrow$ *level(x)* < *level(y).*

This relation $\Re$ is a total order.

The reader has to decide which one is more suitable for his own data.

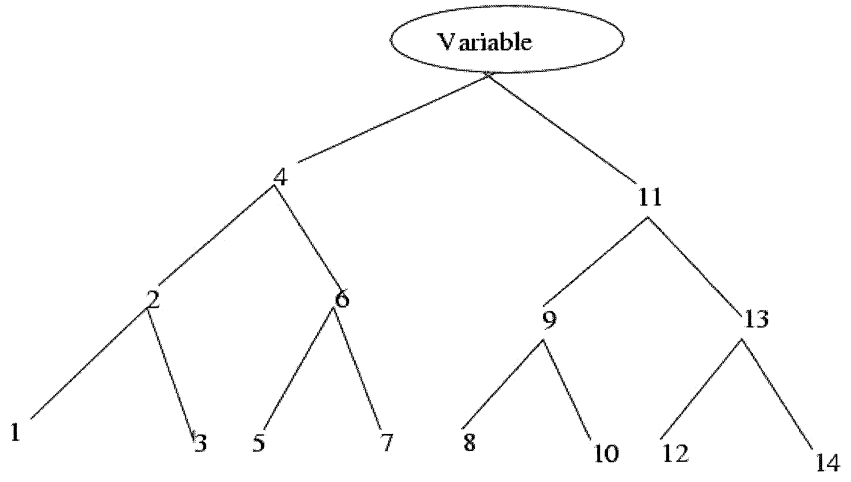


Figure 3. Illustrative example of inorder traversal

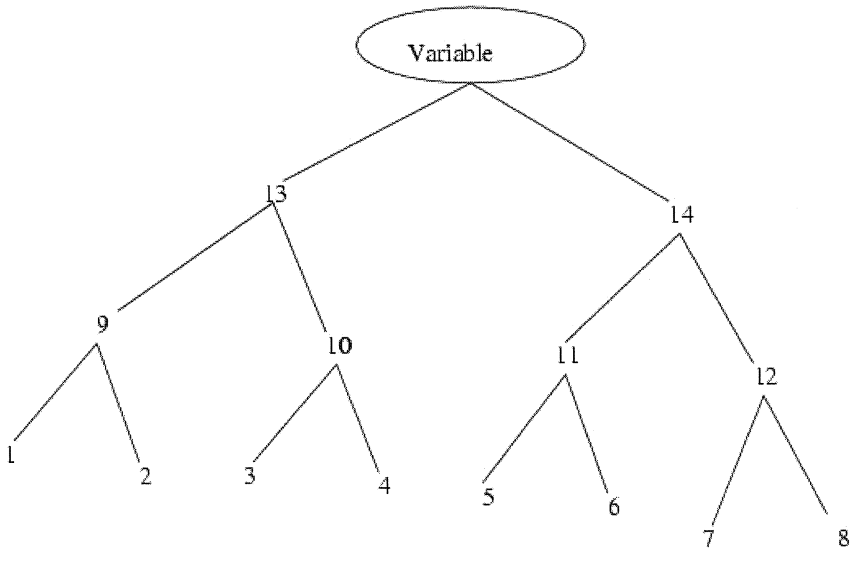


Figure 4. Illustrative example of level-order traversal

3 Decision tree

To build a decision tree, we need a criterion to separate the population of one node, denoted C, into two children nodes (left and right). We use the Kolmogorov-Smirnov (KS) criterion. This criterion allows the separation of a population into two sub-population most homogeneous. It uses the two cumulative distribution functions resulting by the grouping of the m prior classes into two classes called super-classes. The method called "twoing splitting process" ([5]) is used to generate , from the m classes, two super-classes C_1 and C_2 which are associate the two cumulative distribution functions F_1 and F_2 of a symbolic random variable (with interval or taxonomic data). The problem is to find a single cut point x that partitions an interval or taxonomic variable X's values into two block partitions (x is an interval or a node according to the type of the selected predictor): one block (left node) contains those values of X for which XRx (where R is one of the relation defined in Section 2) and the other block (right node) contains the remaining values. Because the calculation of KS distance involves only the cumulative distribution functions, we need only consider approximations (approximated cumulative function). The approximation $\hat{F}_i$ (i=1,2) which estimates F_i is given by:

$$\hat{F}_i(x) = \frac{cardinal\left(\{y \in X/yRx\} \cap C_i\right)}{cardinal(C_i)} \tag{1}$$

where R is one of the relations defined in Section 2. So the Kolmogorov-Smirnov criterion (KS) is defined by:

$$KS(x) = \sup_X \sup_x \left|\hat{F}_1(x) - \hat{F}_2(x)\right| \tag{2}$$

It is a natural extension of this criterion KS, but the selected argument for the cutting is a node or a closed interval and not a real number as the classical case. Thus, we can use all others steps (because these steps are common of all variable's type) in order to build the decision tree. The principle consist of selecting, at each node, the most discriminating predictor (therefore the description of the corresponding individual) maximizing the criterion KS (as indicated in the previous formula).

4 Example of partitioning of the variable to explain

When the variable to explain is not a class variable but a taxonomic or an interval variable, how should we transform it into a class variable? To answer this question in an optimal way, we use the Fisher algorithm ([10]). For example, in the case of two classes, the individuals w are ordered by their description and then each individual is classified in one of the two classes $C_1 = \{i/i \leq w\}$ and $C_2 = \{i/i > w\}$. The partition C=(C_1,C_2) is the one which minimizes the following quantity:

$$W(C_1, C_2) = \sum_{l \in \{1,2\}} \sum_{i,j \in C_l} D(i,j) \tag{3}$$

where D is a numerical measure between the descriptions of the individuals.

We present here an example where the variable to explain is an interval variable. The Table 1 (see the next paragraph for the source of these data) contains the name of the individuals (first column), the initial values (second column) and the order (increasing order) obtained by the "*lower bound*" (third column). Partitioning this interval variable, we obtain Table 2. For the partitioning criterion by the Fisher algorithm, we use the Haussdorff distance between intervals data. Let I=[a,b] and J=[c,d] two intervals, the Haussdorff distance D between the two intervals I and J is defined by:

$$D(I,J) = \max\left(|a-c|, |b-d|\right) \tag{4}$$

Individuals	*Number of Employees*	*Intervals Order*
Various	[87600;416200]	[5292;164744]
Aeronautics	[107100;201400]	[28382;775100]
Energy	[5292;164744]	[44546;311917]
Mechanics	[55500;189413]	[52656;107200]
Chemistry	[52656;107200]	[55500;189413]
Farm Produce	[44546;311917]	[63500;63500]
Automobile	[28382;775100]	[64504;133824]
Electricity	[78900;365000]	[78900;365000]
Computer Science	[95000;383200]	[87600;416200]
Wood-Paper	[63500;63500]	[95000;383200]
Metal-Steel-Metallurgy	[64504;133824]	[107100;201400]

Table 1: Initial values and order

This partitioning enables us to transform a symbolic variable to explain in a class variable as indicated in Table 2. For this example, we choose just two classes. However, this method can be applied to k classes (k<n where n is the number of the individuals).

5 Application

We present here an application (Table 3) of a database of firms in the case where all variables are interval type. This database comes from the free software SODAS (***S****tatistical* ***O****fficial* ***D****ata* ***A****nalysis* ***S****oftware*) available at

www.ceremade.dauphine.fr/~touati/exemples.htm.

The first line of this table indicates the names of the variables and the first column the names of the individuals (the individuals are the name of the firms). This database contains three interval variables which are the predictors: X_1=*Turnover*, X_2=*Number*

Individuals	*Number of Employees*	*Class*
Various	[87600;416200]	2
Aeronautics	[107100;201400]	2
Energy	[5292;164744]	1
Mechanics	[55500;189413]	1
Chemistry	[52656;107200]	1
Farm Produce	[44546;311917]	1
Automobile	[28382;775100]	1
Electricity	[78900;365000]	2
Computer Science	[95000;383200]	2
Wood-Paper	[63500;63500]	2
Metal-Steel-Metallurgy	[64504;133824]	2

Table 2: Class variable

of subsidiaries, X_3=*Disposable Income* and the remaining variable Y=*Number of Employees* is the variable to explain. This variable to explain is partitioned like in Section 4. Therefore, we present just the classes of the individuals for this variable at the initial database (last column of table 3). Ordering these interval variables (ordering the individuals according their description) using for example the order by " *lower bounds*" (see Subsection 2.1.2), we obtain the Table 4.

It remains to calculate the KS criteria of each individual for each predictor (as indicated in Section 3) to deduce the most discriminant predictor. For example, with the predictor *Turnover*, one has:

$$KS([11.54; 49.07]) = \left|\frac{3}{5} - \frac{2}{6}\right| = \frac{4}{15}$$

because there are three intervals of the class 1 before the interval [11.54; 49.07] and two intervals of the class 2 before it (itself counted).

To the root, the program (Visual C++) shows us that it is the predictor X_1 = *Turnover* that is the most discriminating to the interval [11.34; 39.06] (with KS = $0.6 - 0 = 0.6$). This variable separates the root into two nodes: the left node contains three individuals: *Chemistry; Energy; Farm produce* and the right node contains eight individuals: *Wood-Paper; Various; Mechanics; Computer Science; Aeronautics; Electricity; Automobile; Metals-Steel-metallurgy.* The left node is pure because all its individuals are the same class (class 1). As we have a small file, a node is declared terminal (leaf) if it is pure. Therefore, the left node is a terminal node and the process continues the same way for the right node (Figure 5). In the case of a file having several individuals, one can declare for example a node as terminal node when the size

	Turnover	*Number of subsidiaries*	*Disposable Income*	*Number of Employees*
Various	[11.54;49.07]	[3;20]	[0.529;1.861]	2
Aeronautics	[12.02;20.27]	[3;17]	[0.219;0.979]	2
Energy	[11.24;86.65]	[2;24]	[0.013;6.482]	1
Mechanics	[11.87;21.2]	[1;14]	[0.246;0.589]	1
Chemistry	[11.19;35.2]	[5;32]	[0.522;2.487]	1
Farm Produce	[11.34;39.06]	[2;34]	[0.197;2.946]	1
Automobile	[12.37;126.97]	[4;24]	[0.068;4.224]	1
Electricity	[12.02;55.26]	[3;21]	[0.078;3.939]	2
Computer Science	[11.89;63.43]	[6;24]	[0.541;3.758]	2
Wood-Paper	[11.37;11.37]	[14;14]	[0.864;0.864]	2
Metal-Steel-Metallurgy	[13.98;20.76]	[2;21]	[0.408;1.061]	2

Table 3: Initial database

Turnover	*Class*	*Number of Subsidiaries*	*Class*	*Disposable Income*	*Class*
[11.19;35.2]	1	[1;14]	1	[0.013;6.482]	1
[11.25;86.65]	1	[2;21]	2	[0.068;4.224]	1
[11.34;39.06]	1	[2;24]	1	[0.078;3.939]	2
[11.37;11.37]	2	[2;34]	1	[0.197;2.946]	1
[11.54;49.07]	2	[3;17]	2	[0.219;0.979]	2
[11.87;21.2]	1	[3;20]	2	[0.246;0.579]	1
[11.89;63.43]	2	[3;21]	2	[0.408;1.061]	2
[12.02;20.27]	2	[4;24]	1	[0.522;2.487]	1
[12.02;55.26]	2	[5;32]	1	[0.529;1.861]	2
[12.37;126.97]	1	[6;24]	2	[0.541;3.758]	2
[13.98;20.76]	2	[14;14]	2	[0.864;0.864]	2

Table 4: Order and classes of the intervals by predictor

of one class is greatly superior with regard to others (in this case, one can calculate the misclassification rate).

Graphically, we represent an internal node by an ellipsis (containing at the left the number indicating the size of the class 1 and at the right the size of the class 2) and a terminal node by a rectangle containing its size. We note by " $\leq$ " the relation R : " *almost before*" defined in Subsection 2.1.2. For example, we list the individuals for each terminal node. From left to right, the individuals of the two first and the fourth terminal nodes are of class 1, those of the third and fifth terminal nodes are of class 2. Each terminal node corresponds to a decision rule. We obtain five decision rules corresponding to the five terminal nodes (w designates an individual):

- if X_1(w) *is almost before* [11.34;39.06], then w belongs to class 1;
- if X_1(w) *is almost after* [11.34;39.06] and X_2(w) *is almost before* [0.246;0.589] and X_3(w) *is almost before* [1;14], then w belongs to class 1;
- if X_1(w) *is almost after* [11.34;39.06] and X_2(w) *is almost before* [0.246;0.589] and X_3(w) *is almost after* [1;14] and X_3(w) *is almost before* [12.02;55.26], then w belongs to class 2;
- if X_1(w) *is almost after* [11.34;39.06] and X_2(w) *is almost before* [0.246;0.589] and X_3(w) *is almost after* [1;14] and X_1(w) *is almost after* [12.02;55.26], then w belongs to class 1;
- if X_1(w) *is almost after* [11.34;39.06] and X_2(w) *is almost after* [0.246;0.589], then w belongs to class 2;

These decision rules will permit us to classify new observations.

6 Conclusion and perspectives

In this paper, we point out the extension of Kolmogorov-Smirnov criterion to the case where the variables are interval or taxonomic. This criterion require that the variables (the predictors) have to be ordered.

We propose in our future work to :

- adapt the Kolmogorov-Smirnov criterion and the Fisher algorithm to the others symbolic variables;
- examine their complexity.

References

[1] Asseraf, M. (1998) *Extension et Optimisation pour la Segmentation de la distance de Kolmogorov-Smirnov*, Université Paris 9 Dauphine.

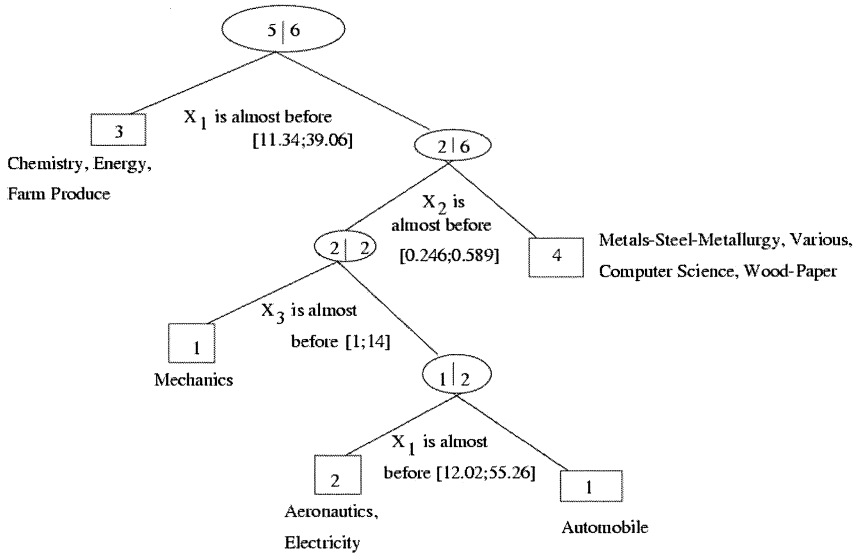


Figure 5. Decision tree.

[2] Asseraf, M. and Mballo, C. (2003), Arbre de décision pour des variables de type intervalle, *Journées de Statistique*, 35.
[3] Asseraf, M. and Mballo, C., and Diday E. (2003), Arbre de décision pour des variables de type taxonomique, in *Méthodes et Perspectives en Classification*, Dodge, Y. and Melfi, G. Eds., Presses Académiques Neuchâtel, 51–55.
[4] Bock, H.H. and Diday, E. (2000), *Analysis of symbolic data: Exploratory methods for extracting statistical information from complex data*, Springer-Verlag, Studies in classification, data analysis and knowledge organisation.
[5] Breiman, L. and Friedman, J.H. and Olshen, R. and Stone, C.J. (1984), *Classification and Regression Trees*, Belmont, CA: Wadsworth.
[6] Castillo, E., Gutierrez, J.M., and Hadi, A.S. (1997), *Expert Systems and Probabilistic Network Models*, Monographs in Computer Science, Springer.
[7] Diday, E. (1998), *L'analyse des données symboliques: un cadre théorique et des outils*, CEREMADE, UMR 7534, Université Paris Dauphine.
[8] Diday, E. Gioia, F. and Mballo, C. (2003), Codage qualitatif d'une variable intervalle, *Journées de Statistique*, 35.
[9] Fisburn, P.C. (1985), *Interval orders and interval graphs: A study of partially ordered sets*, A Wiley-Interscience Publication.
[10] Fisher, W.D. (1958), On grouping for maximum homogeneity, *Journal of the American Statistical Association*, 53, 789–798.
[11] Friedman, J.H. (1977), A recursive partitioning decision rule for non parametric classification, *IEEE transactions on computers*, C-26, 404–408.
[12] Gondran, M. and Minoux, M. (1985), *Graphes et algorithmes*, Eyrolles, Paris.
[13] Pirlot, M. and Vincke, Ph. (1997), *Semi-orders: properties, representations, applications*, Kluwer Academic Publisher.
[14] Quinlan, J.R. (1993), *C4.5: Programs for machine learning*, Morgan kaufmann.
[15] Tsoukias, A. The Ngo, A. and Vincke, Ph. (1999), *PQI interval orders*, LAMSADE, Université Paris Dauphine, 165.
[16] Tsoukias, A. The Ngo, A. (2001), *Numerical representation of PQI interval orders*, LAMSADE, Université Paris Dauphine, 184.
[17] Utgoff, P.E. and Clouse, J.A. (1996), *A kolmogorov-Smirnov metric for decision tree induction*, University of Massachusetts, Amherst, 96-3.

Student (2004), Vol. 5 No. 1, 29-42

On the Comparison and Representation of Fuzzy Partitions

François Bavaud

Université de Lausanne
Informatique et Méthodes Mathématiques
CH-1015 Lausanne, Dorigny
Switzerland

Abstract: A fuzzy partition assigns to each among n objects a distribution over a categories. Elementary linear algebraic methods permit to introduce and investigate concepts and properties such as a) variance and inertia decomposition; b) coarse- and fine-graining (nestedness); c) iteration of fuzzy partitions; d) stability of a group in regard to another partition; e) (Euclidean embeddable) dissimilarities between objects; f) (Euclidean embeddable) dissimilarities between partitions. Unweighted (R) or weighted (T, P) object similarities are further investigated, and found to be related to the chi-square as well as to the indices of Gini, variety and Mirkin-Cherny-Rand. Weighted versions T and P differ for fuzzy partitions, allowing various non-equivalent constructions characterizing differing aspects of fuzzy partitions and possessing no formal analog at the crisp level.

Key words : Fuzzy partitions.

1 Introduction and notations

Partitioning (deterministically) n objects consists in assigning each object i to a group j, among a possible groups; see e.g. Saporta pp. 210-224 (1990) or Mirkin pp. 229-246 (1996) for a classical, formal approach. A *fuzzy* partition consists of a probabilistic assignment of object i to group j, specified with z_{ij} = "probability that object i belongs to group j", obeying $z_{ij} \geq 0$, $\sum_{j=1}^{a} z_{ij} = 1$ and $\sum_{i=1}^{n} z_{ij} > 0$ (absence of empty groups); see e.g. Bezdek (1981) for a presentation of the fuzzy context.

Elementary algebra allows characterizing the combination, iteration or nesting of fuzzy partitions; associated operators, whose projective or Markov-like properties are exploited, possess simple interpretations in terms of dissimilarities between objects, yielding in turn Euclidean embeddable dissimilarities between objects and even between partitions themselves.

The present general framework suggests a certain view of the multivariate analysis of fuzzy partitions (=fuzzy categorical variables), that is of *multiple fuzzy correspondence analysis.*

2 Membership matrices

Definition 1 *A (fuzzy) partition $\mathcal{A}$ of a set of n objects in a groups is defined by a $(n \times a)$ (fuzzy)* membership *or* indicator *matrix such that $z_{ij}^{\mathcal{A}} \geq 0$, $\sum_{j=1}^{a} z_{ij}^{\mathcal{A}} = 1$ for all $i = 1, \ldots, n$ and $n_j^{\mathcal{A}} := \sum_{i=1}^{n} z_{ij}^{\mathcal{A}} > 0$ for all $j = 1, \ldots, a$.*

Definition 2 *A deterministic or* crisp *partition obtains when $z_{ij} = 1$ or $z_{ij} = 0$ for all i, j, or equivalently $z_{ij}^2 = z_{ij}$. In the case of a crisp partition, $j(i)$ will denote the group to which i belongs. A partition is said to be* full *if $Rank(Z) = a$, and* defective *if $Rank(Z) < a$.*

Crisp partitions are full (since $n_j > 0$). The *uniform partition* in a groups $\mathcal{U}(a)$ is defined by the $(n \times a)$ matrix $z_{ij}^{\mathcal{U}(a)} = \frac{1}{a}$ for all i and $j = 1, \ldots, a$. Uniform partitions are defective for $a \geq 2$; the full case $a = 1$ defines the *one-group partition* $\mathcal{O} = \mathcal{U}(1)$, with associated $(n \times 1)$ membership matrix $z_{i1}^{\mathcal{O}} = 1$. The *n-groups partition* $\mathcal{N}$ is defined by the $(n \times n)$ identity matrix $z_{ij}^{\mathcal{N}} = \delta_{ij}$ (or a permutation of it).

2.1 Variance decomposition

Let X be a numerical variable with scores x_i, $i = 1, \ldots, n$. Define the (fuzzy) average for the j-th group as $\bar{x}_j := \sum_{i=1}^{n} \frac{z_{ij}}{n_j} x_i$, and the total average as $\bar{x} = \sum_{j=1}^{a} f_j \bar{x}_j = \frac{1}{n} \sum_{i=1}^{n} x_i$, where $f_j := \frac{n_j}{n}$. Define the total, within- and between-groups variances as

$$\mathrm{var}(x) := \frac{1}{n} \sum_{i=1}^{n} (x_i - \bar{x})^2$$

$$\mathrm{var}_W(x) := \frac{1}{n} \sum_{j=1}^{a} \sum_{i=1}^{n} z_{ij} (x_i - \bar{x}_j)^2 \qquad (1)$$

$$\mathrm{var}_B(x) := \sum_{j=1}^{a} f_j (\bar{x}_j - \bar{x})^2$$

Then the (fuzzy) variance decomposition formula $\mathrm{var}(x) = \mathrm{var}_W(x) + \mathrm{var}_B(x)$ holds. In particular, for fixed values $\{x_i\}$, $\mathrm{var}_W(x)$ is maximum for $\mathcal{A} = \mathcal{U}(a)$ (for any a), and minimum for $\mathcal{A} = \mathcal{N}$.

2.2 Connected components

Define the $(a \times a)$ matrices $B = (b_{jj'})$ and $N = (n_{jj'})$ as

$$B := Z'Z \quad \text{i.e.} \quad b_{jj'} := \sum_i z_{ij} z_{ij'} \qquad N := \mathrm{diag}(\mathbf{1}'Z) \quad \text{i.e.} \quad n_{jj'} := \delta_{jj'}\, n_j \quad (2)$$

where $\mathbf{1}$ is the $(n \times 1)$ unit vector. Note that $Z\ 1 = \mathbf{1}$ and $Z'\mathbf{1} = N1$, where 1 is the $(a \times 1)$ unit vector. Also, B^{-1} exists if and only if $\mathcal{A}$ is full.

$b_{jj'} \geq 0$ constitutes an index of overlapping between groups j and j' and measures their common sharing of objects. Distinct groups j and j' with $b_{jj'} > 0$ are said to be *adjacent*. Distinct groups j and j' related by a path $b_{jk_1}\, b_{k_1k_2} \ldots b_{k_lj'} > 0$ of adjacent groups are *connected*. A set of connected groups constitutes an (irreducible) *component*, indexed by $J = 1, \ldots, c(\mathcal{A})$, where $c(\mathcal{A}) \leq m$ is the number of irreducible components of the partition $\mathcal{A}$, or, equivalently, the number of irreducible blocks of Z. One has:

$$\begin{array}{ccccc} c(\mathcal{A}) = m & \Leftrightarrow & \mathcal{A} \text{ is crisp} & \Leftrightarrow & B = N \\ \mathrm{Rank}(Z) = m & \Leftrightarrow & \mathcal{A} \text{ is full} & \Leftrightarrow & B^{-1} \text{ exists} \end{array}$$

2.3 Iterated partitions

In view of the previous section, the matrix $G := N^{-1}B$ is the identity if and only if is $\mathcal{A}$ crisp. In general, G generates *iterated partitions*:

Definition 3 *The r-th iterated membership $Z^{(r)}$ defining partition $\mathcal{A}^{(r)}$ obtains as*

$$Z^{(r)} := Z\ G^{r-1} \qquad G := N^{-1}B = (g_{jj'}) \qquad g_{jj'} = \frac{1}{n_j} \sum_{i=1}^{n} z_{ij}\ z_{ij'} \quad (3)$$

Indeed, identity $G\ 1 = 1$ ensures the normalization $Z^{(r)}\ 1 = 1$: that is, $Z^{(r)}$ is the membership matrix associated to some (fuzzy) partition denoted $\mathcal{A}^{(r)}$.
$G\ 1 = 1$ with $g_{jj'} \geq 0$ also shows G to be the $(a \times a)$ transition matrix of a Markov chain among groups $j = 1, \ldots, a$: $g_{jj'}$ is the probability that, starting from group j in which one selects an individual i, one precisely gets group j' when further selecting a group from individual i. Identity $\mathbf{1}'ZN^{-1}Z'Z = \mathbf{1}'Z$ ensures $n_j^{(r)} := \sum_i z_{ij}^{(r)} = n_j$: the group sizes are thus unchanged by iteration.

G is doubly stochastic, and made up of $J = 1, \ldots, c(\mathcal{A})$ irreducible doubly stochastic matrices $G^{(J)}$, each with stationary distribution $f_j^{(J)} = n_j/n_J$ where $n_J := \sum_{j \in J} n_j$. Iterating partitions mixes the objects i among the various classes j of the *same* connected component J; for instance, $z_{ij}^{(2)} = \sum_{i'j'} z_{ij'}\, z_{i'j'}\, z_{i'j}/n_{j'}$. In the limit $r \to \infty$,

objects inside the same component J possess the same group membership:

$$g_{jj'}^{(\infty)} = \frac{n_{j'}\, I(j' \in J(j))}{n_{J(j)}} \qquad \text{implying} \qquad z_{ij}^{(\infty)} = \frac{n_j\, I(i \in J(j))}{n_{J(j)}} \tag{4}$$

where $I(E)$ denotes the characteristic function for event E, and $J(j)$ denotes the component to which group j belongs. Partition $\mathcal{A}^{(\infty)}$ thus obtains by

(1) first assigning individuals i to their component $J(i)$; we denote this partition as $\mathcal{A}^{(0)}$, with $(n \times c(\mathcal{A}))$ associated membership matrix $z_{iJ}^{(0)} = I(i \in J)$

(2) then choosing group $j \in J(i)$ with probability $n_j/n_{J(i)} = f_j/f_{J(i)}$; the $(c(\mathcal{A}) \times a)$ membership matrix associated to this *component-group* partition is $z_{Jj}^{\mathrm{cg}} = \frac{n_j}{n_J} : I(j \in J)$.

By construction

$$Z^{(\infty)} = Z^{(0)}\, Z^{\mathrm{cg}} \qquad Z^{(0)} = Z\, Z^{\mathrm{gc}} \qquad \text{where} \quad z_{iJ}^{\mathrm{gc}} := I(j \in J) \tag{5}$$

Partition $\mathcal{A}^{(\infty)}$ is defective if and only if $\mathcal{A}$ is fuzzy (since $\mathrm{Rank}(Z^{(\infty)}) = c(\mathcal{A}) < a$), and full if and only if $\mathcal{A}$ is crisp. Crisp partitions are characterized by $g_{jj'} = g_{jj'}^{(r)} = \delta_{jj'}$ and $z_{ij} = z_{ij}^{(r)} = I(i \in j)$.

Example 1 *Consider the fuzzy partition $\mathcal{A}$ of $n = 5$ objects in $a = 4$ classes with*

$$Z^{\mathcal{A}} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0.2 & 0.8 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0.6 & 0.4 \\ 0 & 0 & 0.2 & 0.8 \end{pmatrix} \qquad N = \begin{pmatrix} 1.2 & 0 & 0 & 0 \\ 0 & 1.8 & 0 & 0 \\ 0 & 0 & 0.8 & 0 \\ 0 & 0 & 0 & 1.2 \end{pmatrix}$$

$$B = \begin{pmatrix} 1.04 & 0.16 & 0 & 0 \\ 0.16 & 1.64 & 0 & 0 \\ 0 & 0 & 0.4 & 0.4 \\ 0 & 0 & 0.4 & 0.8 \end{pmatrix} \qquad G = \begin{pmatrix} \frac{13}{15} & \frac{2}{15} & 0 & 0 \\ \frac{4}{45} & \frac{41}{45} & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & \frac{1}{3} & \frac{2}{3} \end{pmatrix}$$

$$G^{(\infty)} = \begin{pmatrix} 0.4 & 0.6 & 0 & 0 \\ 0.4 & 0.6 & 0 & 0 \\ 0 & 0 & 0.4 & 0.6 \\ 0 & 0 & 0.4 & 0.6 \end{pmatrix} \qquad Z^{(\infty)} = \begin{pmatrix} 0.4 & 0.6 & 0 & 0 \\ 0.4 & 0.6 & 0 & 0 \\ 0.4 & 0.6 & 0 & 0 \\ 0 & 0 & 0.4 & 0.6 \\ 0 & 0 & 0.4 & 0.6 \end{pmatrix}$$

3 Object comparisons

3.1 Object similarities

Let $S = (s_{ii'})$ denote a general $(n \times n)$ similarity matrix between objects, obeying $s_{ii'} \geq 0$, $s_{ii'} = s_{i'i}$ and $s_{ii'} \leq \sqrt{s_{ii}\, s_{i'i'}}$. Three natural candidates for S are provided by the $(n \times n)$ matrices $R := ZZ'$, $T := ZN^{-1}Z'$ and (assuming the partition to be

full, that is B^{-1} exists) $P := ZB^{-1}Z'$, namely[1]

$$r_{ii'} := \sum_{j=1}^{a} z_{ij}\, z_{i'j} \qquad t_{ii'} := \sum_{j=1}^{a} \frac{z_{ij}\, z_{i'j}}{n_j} \qquad p_{ii'} := \sum_{j,j'=1}^{a} z_{ij}\, b_{jj'}^{(-1)}\, z_{i'j'} \quad (6)$$

- $R = (r_{ii'})$ (the *relation* similarity matrix) yields, for a crisp classification, the indicator matrix of the relation "objects i and i' belong to the same group".
- $T = (t_{ii'})$ (the *transition* similarity matrix) satisfies $\sum_{i'} t_{ii'} = 1$: it is thus the transition matrix among objects of a doubly stochastic Markov chain; $t_{ii'}$ is the probability to jump from object i to object i' when first selecting class j with probability z_{ij} and then selecting object i' inside class j with probability $z_{i'j}/n_j$.
- $P = (p_{ii'})$(the *projection* similarity matrix) is a projection matrix (see Theorem 1). Also, $P = T$ if and only if $\mathcal{A}$ is crisp; in other words, similarity matrices can be made simultaneously Markovian and projective for crisp partitions only. Note that $p_{ii'} \geq 0$ can be violated (since B^{-1} possesses negative components); however, $|p_{ii'}| \leq \sqrt{p_{ii}\, p_{i'i'}}$ holds.

Theorem 1 *a) T is a Markov transition matrix, with stationary uniform distribution $\pi_i = 1/n$; its iterate obeys $T^2 = T$ if and only if $\mathcal{A}$ is crisp (that if and only if $T = P$ as well).*
b) In general, however, $1 \leq Tr(T^2) \leq Tr(T) \leq a$, with $Tr(T) = 1$ if and only if $\mathcal{A}$ is the uniform partition $\mathcal{U}(a)$ (for any a), and $Tr(T) = a$ if and only if $\mathcal{A}$ is crisp, in which case $T = P$.
c) P exists if and only if $\mathcal{A}$ is full, in which case $P^2 = P$ and $Tr(P) = a$.

Proof.

a) obtains from c) below; recall $T = P$ in the crisp case.

b) by definition, $\mathrm{Tr}(T) = \sum_{ij} \frac{z_{ij}^2}{n_j}$. $\mathrm{Tr}(T) = a$ holds as a consequence of $z_{ij}^2 \leq z_{ij}$ with equality if and only if $\mathcal{A}$ is crisp. $\mathrm{Tr}(T) = 1$ obtains from Jensen's inequality $\frac{1}{n}\sum_i z_{ij}^2 \geq \{\frac{1}{n}\sum_i z_{ij}\}^2$ with equality if and only if $\mathcal{A}$ is the uniform partition.

Inequality $(z_{ij} - z_{i'j})^2 z_{ij'} z_{i'j'} \geq 0$ holds in general, while $(z_{ij} - z_{i'j})^2 z_{ij'} z_{i'j'} = 0$ for all i, i', j, j' if and only if $\mathcal{A}$ is crisp. Summing the latter yields

$$\mathrm{Tr}(T^2) = \sum_{ii'jj'} \frac{z_{ij} z_{i'j} z_{ij'} z_{i'j'}}{n_j\, n_{j'}} \leq \sum_{ii'jj'} \frac{z_{ij}^2 z_{ij'} z_{i'j'}}{n_j\, n_{j'}} = \sum_{ij} \frac{z_{ij}^2}{n_j} = \mathrm{Tr}(T)$$

which demonstrates that $T^2 \neq T$ if $\mathcal{A}$ is not crisp. On the other hand, $T = P$ if $\mathcal{A}$ is crisp, and thus $T^2 = T$.

c) $P^2 = ZB^{-1}Z'ZB^{-1}Z' = ZB^{-1}BB^{-1}Z' = ZB^{-1}Z' = P$; also,
$\mathrm{Tr}\,(P) = \mathrm{Tr}\,(ZB^{-1}Z') = \mathrm{Tr}\,(B^{-1}Z'Z) = \mathrm{Tr}\,(B^{-1}B) = \mathrm{Tr}\, I = a$.

[1] Equations (2) and (6) are somewhat reminiscent of the "Burt-Condorcet" duality in multiple correspondence analysis (see e.g. Marcotorchino (2000)). Recall however the latter to refer to $p \geq 2$ *crisp partitions*, rather than *one fuzzy partition* as in the present case.

■

3.2 Iterated object similarities

Higher order similarities can be constructed as

$$R^{(r)} := Z^{(r)}\,(Z^{(r)})', \quad T^{(r)} := Z^{(r)}\,(N^{(r)})^{-1}\,(Z^{(r)})'$$

and (for a full partition)

$$P^{(r)} := Z^{(r)}\,(B^{(r)})^{-1}\,(Z^{(r)})',$$

where

$$Z^{(r)} := Z\,G^{r-1}, \quad B^{(r)} := (Z^{(r)})'\,Z^{(r)} \text{ and } N^{(r)} := \mathrm{diag}(\mathbf{1}'Z^{(r)}).$$

Theorem 2 *For $r \geq 0$, $T^{(r)} = T^{2r-1}$ and (for a full partition) $P^{(r)} = P$.*

Proof.

$$\begin{aligned} P^{(r+1)} &= Z^{(r+1)}\,(B^{(r+1)})^{-1}\,(Z^{(r+1)})' \\ &= Z^{(r)}N^{-1}B[BN^{-1}B^{(r)}N^{-1}B]^{-1}BN^{-1}(Z^{(r)})' \\ &= Z^{(r)}N^{-1}BB^{-1}N(B^{(r)})^{-1}NB^{-1}BN^{-1}(Z^{(r)})' \\ &= Z^{(r)}\,(B^{(r)})^{-1}\,(Z^{(r)})' \\ &= P^{(r)} \end{aligned}$$

Using $N^{(r)} = N$, identity $T^{(r)} = T^{2r-1}$ is proved similarly. ■

3.3 Object distances

Matrices R, T and P are three instances of positive-definite similarity matrices $S = (s_{ii'})$ between objects i and i', from which a *squared Euclidean distance* can be constructed as $D^S_{ii'} := (d^S_{jj'})^2 = s_{ii} + s_{i'i'} - 2s_{ii'}$ (Schoenberg 1935; Gower 1982). Explicitly

$$D^R_{ii'} = \sum_j (z_{ij} - z_{i'j})^2$$

$$D^T_{ii'} = \sum_j \frac{(z_{ij} - z_{i'j})^2}{n_j} \tag{7}$$

$$D^P_{ii'} = \sum_{jj'} (z_{ij} - z_{i'j}) b^{(-1)}_{jj'} (z_{ij'} - z_{i'j'})$$

Theorem 3 *For a crisp partition:*

$$r_{ii'} = 1 \qquad t_{ii'} = p_{ii'} = \frac{1}{n_j} \qquad D^R_{ii'} = D^T_{ii'} = D^P_{ii'} = 0 \qquad \text{for } i, i' \in j$$

$$r_{ii'} = t_{ii'} = p_{ii'} = 0 \qquad D^R_{ii'} = 2 \quad D^T_{ii'} = D^P_{ii'} = \frac{1}{n_j} + \frac{1}{n_{j'}} \qquad \text{for } i \in j,\ i' \in j' \text{ with } j \neq j'$$

In particular, $r^{\mathcal{N}}_{ii'} = t^{\mathcal{N}}_{ii'} = p^{\mathcal{N}}_{ii'} = \delta_{ii'}$; $r^{\mathcal{O}}_{ii'} = 1$ and $t^{\mathcal{O}}_{ii'} = p^{\mathcal{O}}_{ii'} = \frac{1}{n}$.

Proof. Straightforward. ■

Let $a_j \geq 0$ with $\sum_j a_j = 1$ be the membership profile of some object a; for instance, $g_j = \frac{1}{n}\sum_i z_{ij} = \frac{n_j}{n}$ represents the membership profile of the gravity center g. Then squared distances D^S_{ia} can be defined by the substitution $z_{i'j} \to a_j$ in (7). Define

$$I_2^S =: \frac{1}{2n^2}\sum_{i,i'} D^S_{ii'} \quad \text{(pair inertia)}, \quad I_1^S(a) =: \frac{1}{n}\sum_i D^S_{ia} \quad \text{(central inertia with center } a\text{)} \tag{8}$$

Then, for $S = R, T$ or P,

$$I_1^S(a) = I_1^S(g) + D^S_{ag} \quad \text{(strong Huygens principle)} \qquad I_2^S = I_1^S(g) \quad \text{(weak Huygens principle)} \tag{9}$$

(see e.g. Bavaud (2002)). The pair inertia $I_2^S = I_1^S(g)$ constitutes an index of classificatory diversity; for crisp partitions $\mathcal{A}$, one gets $I_2^R = \sum_{j=1}^a f_j(1-f_j)$ (Gini diversity index) and $I_2^T = I_2^P = (a-1)/n$.

As it it well known (classical MDS), coordinates $x^S_{i\alpha}$ realizing an Euclidean representation of the objects $i = 1, \ldots, n$ in dimensions $\alpha = 1, \ldots, a-1$ (that is satisfying $D^S_{ii'} = \sum_\alpha (x^S_{i\alpha} - x^S_{i'\alpha})^2$) can be obtained as $x^S_{i\alpha} := \sqrt{\lambda^S_\alpha}\, u^S_{i\alpha}$, where the λ^S_α are the eigenvalues and the $u^S_{i\alpha}$ the eigenvectors occurring in the spectral decomposition $S = U^S \Lambda^S (U^S)'$.

Example 1, continued: The (5×5) corresponding similarity matrices are

$$R = \begin{pmatrix} 1 & .2 & 0 & 0 & 0 \\ .2 & .68 & .8 & 0 & 0 \\ 0 & .8 & 1 & 0 & 0 \\ 0 & 0 & 0 & .52 & .44 \\ 0 & 0 & 0 & .44 & .68 \end{pmatrix}$$

$$T = \begin{pmatrix} .83 & .17 & 0 & 0 & 0 \\ .17 & .39 & .44 & 0 & 0 \\ 0 & .44 & .56 & 0 & 0 \\ 0 & 0 & 0 & .58 & .42 \\ 0 & 0 & 0 & .42 & .58 \end{pmatrix} \qquad P = \begin{pmatrix} .98 & .12 & -.10 & 0 & 0 \\ .12 & .40 & .48 & 0 & 0 \\ -.10 & .48 & .62 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

with associated squared distances *between objects*

$$D^R = \begin{pmatrix} 0 & 1.28 & 2 & 1.52 & 1.68 \\ 1.28 & 0 & .08 & 1.2 & 1.36 \\ 2 & .08 & 0 & 1.52 & 1.68 \\ 1.52 & 1.2 & 1.52 & 0 & .32 \\ 1.68 & 1.36 & 1.68 & .32 & 0 \end{pmatrix}$$

$$D^T = \begin{pmatrix} 0 & .89 & 1.39 & 1.42 & 1.42 \\ .89 & 0 & .06 & .97 & .97 \\ 1.39 & .06 & 0 & 1.14 & 1.14 \\ 1.42 & .97 & 1.14 & 0 & .33 \\ 1.42 & .97 & 1.14 & .33 & 0 \end{pmatrix} \qquad D^P = \begin{pmatrix} 0 & 1.14 & 1.79 & 1.98 & 1.98 \\ 1.14 & 0 & .07 & 1.40 & 1.40 \\ 1.79 & .07 & 0 & 1.62 & 1.62 \\ 1.98 & 1.40 & 1.62 & 0 & 2 \\ 1.98 & 1.40 & 1.62 & 2 & 0 \end{pmatrix}$$

Spectral decomposition of S yields the corresponding (5×4) coordinates $X^S = (x^S_{i\alpha})$:

$$X^R = \begin{pmatrix} .24 & .97 & 0 & 0 \\ .82 & 0 & 0 & 0 \\ .97 & -.24 & 0 & 0 \\ 0 & 0 & .66 & -.30 \\ 0 & 0 & .79 & .25 \end{pmatrix}$$

$$X^T = \begin{pmatrix} .58 & -.71 & 0 & 0 \\ .58 & .24 & 0 & 0 \\ .58 & .47 & 0 & 0 \\ 0 & 0 & -.71 & .29 \\ 0 & 0 & -.71 & -.29 \end{pmatrix} \quad X^P = \begin{pmatrix} -.12 & .98 & 0 & 0 \\ .61 & .19 & 0 & 0 \\ .79 & -.24 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

4 Nested partitions: coarser and finer

Definition 4 *Partition $\mathcal{B}$ (defined by the $(n \times b)$ membership matrix $Z^{\mathcal{B}}$) is* coarser *than partition $\mathcal{A}$ (defined by the $(n \times a)$ membership matrix $Z^{\mathcal{A}}$), or, equivalently, $\mathcal{A}$ is* finer *than $\mathcal{B}$, noted $\mathcal{B} \leq \mathcal{A}$, if $Z^{\mathcal{B}} = Z^{\mathcal{A}}\, W^{\mathcal{A}\mathcal{B}}$ where $W^{\mathcal{A}\mathcal{B}} = (w^{\mathcal{A}\mathcal{B}}_{jk})$ is a $(a \times b)$* class membership matrix *such that $w^{\mathcal{A}\mathcal{B}}_{jk} \geq 0$ and $\sum_{k=1}^{b} w^{\mathcal{A}\mathcal{B}}_{jk} = 1$.*

Theorem 4 *a) The relation "$\mathcal{B} \leq \mathcal{A}$" is a partial order;*
b) its minimal element is the one-group partition *$\mathcal{O}$;*
c) its maximal element is the n-groups partition *$\mathcal{N}$;*
d) if $\mathcal{B} \leq \mathcal{A}$ (with $\mathcal{A}$ and $\mathcal{B}$ both full) then $P^{\mathcal{A}}P^{\mathcal{B}} = P^{\mathcal{B}}P^{\mathcal{A}} = P^{\mathcal{B}}$.

Proof.

a) By definition, $\mathcal{A} \leq \mathcal{A}$ (with $w^{\mathcal{A}\mathcal{A}}_{jk} = \delta_{jk}$). Also, $\mathcal{B} \leq \mathcal{A}$ and $\mathcal{C} \leq \mathcal{B}$ entail $\mathcal{C} \leq \mathcal{A}$ (with $W^{\mathcal{A}\mathcal{C}} = W^{\mathcal{A}\mathcal{B}}\, W^{\mathcal{B}\mathcal{C}}$).

b) for any $\mathcal{A}$, $Z^{\mathcal{O}} = Z^{\mathcal{A}}\, W^{\mathcal{A}\mathcal{O}}$ with $w^{\mathcal{A}\mathcal{O}}_{j} = 1$ for all $j = 1, \ldots, a$.

c) for any $\mathcal{B}$, $Z^{\mathcal{B}} = Z^{\mathcal{N}}\, W^{\mathcal{N}B}$ with $w^{\mathcal{N}B}_{ij} = z^{\mathcal{B}}_{ij}$.

d) $Z^{\mathcal{A}}W^{\mathcal{A}\mathcal{B}} = Z^{\mathcal{B}}$ and $B^{\mathcal{A}} = (Z^{\mathcal{A}})'Z^{\mathcal{A}}$ yield

$$\begin{aligned} P^{\mathcal{A}}P^{\mathcal{B}} &= Z^{\mathcal{A}}(B^{\mathcal{A}})^{-1}(Z^{\mathcal{A}})'Z^{\mathcal{B}}(B^{\mathcal{B}})^{-1}(Z^{\mathcal{B}})' \\ &= Z^{\mathcal{A}} \underbrace{(B^{\mathcal{A}})^{-1}(Z^{\mathcal{A}})'Z^{\mathcal{A}}}_{I} W^{\mathcal{A}\mathcal{B}}(B^{\mathcal{B}})^{-1}(Z^{\mathcal{B}})' \\ &= Z^{\mathcal{B}}(B^{\mathcal{B}})^{-1}(Z^{\mathcal{B}})' = P^{\mathcal{B}} \end{aligned}$$

Identity $P^{\mathcal{B}}P^{\mathcal{A}} = P^{\mathcal{B}}$ is demonstrated analogously.

■

Theorem 5 *For $r \geq 1$, the sequence of partitions $\mathcal{A}^{(r)}$ associated with the iterated memberships $Z^{(r)}$ (definition 3) is decreasing (that is $\mathcal{A}^{(r+1)} \leq \mathcal{A}^{(r)}$). Its limit $\mathcal{A}^{(\infty)}$ is given by the membership matrix $Z^{(\infty)}$ defined in (4). Also, $\mathcal{A}^{(\infty)} \leq \mathcal{A}^{(0)} \leq \mathcal{A}$.*

Proof. The first two assertions follow from $Z^{(r+1)} = Z^{(r)}\, G$ (equation (3)), where G is a $(a \times a)$ non-negative matrix obeying $\sum_{j'=1}^{a} g_{jj'} = 1$ together with properties listed in Section 2.3. The last assertion is a direct consequence of (5).
■

5 Comparison and representation of partitions

5.1 The general case: Euclidean visualization

Definition 5 *Let $S^{\mathcal{A}} = (s^{\mathcal{A}}_{ii'})$, respectively $S^{\mathcal{B}} = (s^{\mathcal{B}}_{ii'})$, be the $(n \times n)$ similarity matrix associated to partition $\mathcal{A}$, respectively partition $\mathcal{B}$. The corresponding* (squared) distance $D^S_{\mathcal{A},\mathcal{B}}$ *between two partitions $\mathcal{A}$ and $\mathcal{B}$ is defined as*

$$D^S_{\mathcal{A},\mathcal{B}} := \sum_{ii'} (s^{\mathcal{A}}_{ii'} - s^{\mathcal{B}}_{ii'})^2 = \mathrm{Tr}(S^{\mathcal{A}} - S^{\mathcal{B}})^2 = \mathrm{Tr}((S^{\mathcal{A}})^2) + \mathrm{Tr}((S^{\mathcal{B}})^2) - 2\mathrm{Tr}(S^{\mathcal{A}}\, S^{\mathcal{B}}) \quad (10)$$

The distance $D^S_{\mathcal{A},\mathcal{B}}$ possesses all the properties of a squared Euclidean distance, in particular the embeddability property. Then classical MDS applied on matrix D^S yields a low-dimensional Euclidean visualization of the distances between partitions, *each partition being represented by a point* (see Figure 1).

Example 2 *Consider $n = 5$ objects. Define*

- $\mathcal{A}$ as the fuzzy partition of example 1
- $\mathcal{B}$ as the partition $\mathcal{A}^{(0)}$ in connected components (namely (123; 45))
- $\mathcal{C}$ as the crisp partition (12; 345)
- $\mathcal{D} \equiv \mathcal{N}$ as the n-groups partition (1; 2; 3; 4; 5)
- $\mathcal{E} \equiv \mathcal{O}$ as the one-group partition (12345)
- $\mathcal{F} \equiv \mathcal{A}^{(\infty)}$ as the limiting iterated partition:

$$Z^{\mathcal{A}} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0.2 & 0.8 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0.6 & 0.4 \\ 0 & 0 & 0.2 & 0.8 \end{pmatrix} \qquad Z^{\mathcal{B}} = \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix} \qquad Z^{\mathcal{C}} = \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{pmatrix}$$

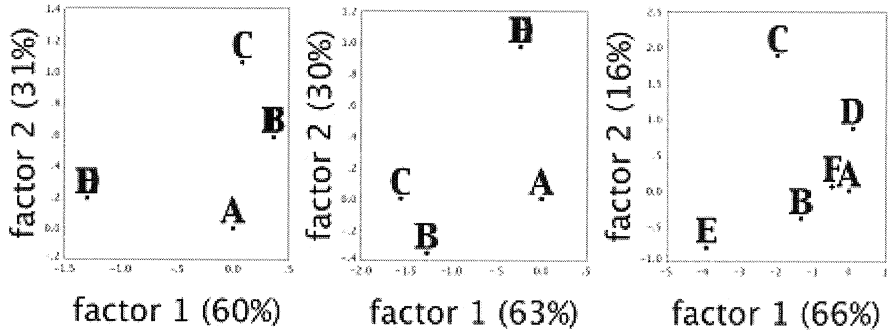


Figure 1. Euclidean visualization (classical MDS) of the distances between partitions $D^S_{\mathcal{A},\mathcal{B}}$, for $S = T$ (left), $S = P$ (middle) and $S = R$ (right). Coordinates for $\mathcal{D}$ and $\mathcal{E}$ are identical in the $T-$ and $P-$representation; also, coordinates for $\mathcal{B}$ and $\mathcal{F}$ are identical in the $T-$representation. Recall that $\mathcal{F}$ is defective and hence not $P-$representable.

$$Z^{\mathcal{D}} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \qquad Z^{\mathcal{E}} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \qquad Z^{\mathcal{F}} = \begin{pmatrix} 0.4 & 0.6 & 0 & 0 \\ 0.4 & 0.6 & 0 & 0 \\ 0.4 & 0.6 & 0 & 0 \\ 0 & 0 & 0.4 & 0.6 \\ 0 & 0 & 0.4 & 0.6 \end{pmatrix}$$

The corresponding dissimilarities (10) D^S *between partitions* (listed in alphabetical order) are[2]

$$D^R = \begin{pmatrix} 0 & 4.42 & 7.62 & 2.18 & 16.42 & 1.43 \\ 4.42 & 0 & 8 & 8 & 12 & 3.00 \\ 7.62 & 8 & 0 & 8 & 12 & 7.16 \\ 2.18 & 8 & 8 & 0 & 20 & 3.32 \\ 16.42 & 12 & 12 & 20 & 0 & 15.00 \\ 1.43 & 3.00 & 7.16 & 3.32 & 15.00 & 0 \end{pmatrix}$$

$$D^T = \begin{pmatrix} 0 & 0.63 & 1.37 & 1.74 & 1.74 & 0.63 \\ 0.63 & 0 & 1.11 & 3 & 3 & 0 \\ 1.37 & 1.11 & 0 & 3 & 3 & 1.11 \\ 1.74 & 3 & 3 & 0 & 0 & 3 \\ 1.74 & 3 & 3 & 0 & 0 & 3 \\ 0.63 & 0 & 1.11 & 3 & 3 & 0 \end{pmatrix} \qquad D^P = \begin{pmatrix} 0 & 2 & 2.63 & 1 & 1 \\ 2 & 0 & 1.11 & 3 & 3 \\ 2.63 & 1.11 & 0 & 3 & 3 \\ 1 & 3 & 3 & 0 & 0 \\ 1 & 3 & 3 & 0 & 0 \end{pmatrix}$$

5.2 The full case for $S = P$

When $\mathcal{A}$ and $\mathcal{B}$ are both full, $P^{\mathcal{A}}$ and $P^{\mathcal{B}}$ are well defined and Theorem 1 yields $D^P_{\mathcal{A},\mathcal{B}} = \mathrm{Tr}(P^{\mathcal{A}} + P^{\mathcal{B}} - 2\,P^{\mathcal{A}}\,P^{\mathcal{B}})$. If $\mathcal{B} \leq \mathcal{A}$ in addition, the distance further expresses

[2] as $\mathcal{F}$ is defective, $P^{\mathcal{F}}$ is not defined.

(Theorem 4) as $D^P_{\mathcal{A},\mathcal{B}} = \mathrm{Tr}(P^{\mathcal{A}}) - \mathrm{Tr}(P^{\mathcal{B}}) = a - b$: the distance between two nested, full partitions is measured by the difference of their number of groups. In particular

- $D^P_{\mathcal{A},\mathcal{C}} = D^P_{\mathcal{A},\mathcal{B}} + D^P_{\mathcal{B},\mathcal{C}}$ if $\mathcal{C} \leq \mathcal{B} \leq \mathcal{A}$ or $\mathcal{A} \leq \mathcal{B} \leq \mathcal{C}$
- for $\mathcal{A}$ full, $D^P_{\mathcal{O},\mathcal{A}} = a - 1$ and $D^P_{\mathcal{N},\mathcal{A}} = n - a$
- for $\mathcal{A}$ full, $D^P_{\mathcal{A}^{(0)},\mathcal{A}} = a - c(\mathcal{A})$.

5.3 The crisp case: chi-square and Mirkin-Cherny-Rand indices

Let $\mathcal{A}$ and $\mathcal{B}$ be two crisp partitions possessing respectively a and b non-empty classes. Let $n^{\mathcal{A}}_j := \sum_i z^{\mathcal{A}}_{ij}$ be the number of objects in class j of $\mathcal{A}$, $n^{\mathcal{B}}_k := \sum_i z^{\mathcal{B}}_{ik}$ be the number of objects in class k of $\mathcal{B}$, and define $n^{\mathcal{A},\mathcal{B}}_{jk} := \sum_i z^{\mathcal{A}}_{ij} z^{\mathcal{B}}_{ik}$ as the number of objects both in class j of $\mathcal{A}$ and k of $\mathcal{B}$.

Definition 6 $N_{\mathcal{A},\mathcal{B}} = \sum_{ii'}\sum_{jj'} z^{\mathcal{A}}_{ij}\, z^{\mathcal{A}}_{i'j}\, z^{\mathcal{B}}_{ij'}\, z^{\mathcal{B}}_{i'j'} = \sum_{ii'} r^{\mathcal{A}}_{ii'}\, r^{\mathcal{B}}_{ii'} = \sum_{jk}(n^{\mathcal{A},\mathcal{B}}_{jk})^2$ *denotes the* number of pairs *(distinct or not) which are simultaneously classified in the same group j of $\mathcal{A}$ and k of $\mathcal{B}$.*

$n^{\mathcal{A},\mathcal{B}}_{jk} \leq n^{\mathcal{A},\mathcal{A}}_{jj} = (n^{\mathcal{A}}_j)^2$, and thus $N_{\mathcal{A},\mathcal{B}} \leq N_{\mathcal{A},\mathcal{A}} = \sum_j (n^{\mathcal{A}}_j)^2$ (and $N_{\mathcal{A},\mathcal{B}} \leq N_{\mathcal{B},\mathcal{B}} = \sum_k (n^{\mathcal{B}}_k)^2$), with equality if and only if the two (crisp) partitions $\mathcal{A}$ and $\mathcal{B}$ are identical.

Theorem 6

$$D^R_{\mathcal{A},\mathcal{B}} = N_{\mathcal{A},\mathcal{A}} + N_{\mathcal{B},\mathcal{B}} - 2N_{\mathcal{A},\mathcal{B}} \qquad D^T_{\mathcal{A},\mathcal{B}} = D^P_{\mathcal{A},\mathcal{B}} = (a-1) + (b-1) - \frac{2}{n}\chi^2_{\mathcal{A},\mathcal{B}} \quad (11)$$

where $\chi^2_{\mathcal{A},\mathcal{B}} := \sum_{j=1}^{a}\sum_{k=1}^{b} \frac{(n_{jk} - \frac{n_{j\bullet} n_{\bullet k}}{n})^2}{\frac{n_{j\bullet} n_{\bullet k}}{n}}$ *is the chi-square associated to the contingency table* $n_{jk} = n^{\mathcal{A},\mathcal{B}}_{jk}$.

The quantity $\frac{1}{n^2} D^R_{\mathcal{A},\mathcal{B}}$ is called "relative symmetric-difference distance" by Mirkin and Cherny (1970). Its complement to unity[3] is known as the "Rand similarity index" (Rand 1971).

[3] at least in the variant restricted to the contribution of *distinct* pairs only.

Proof. The first identity follows from

$$
\begin{aligned}
D^R_{\mathcal{A},\mathcal{B}} &= \sum_{ii'}(r^{\mathcal{A}}_{ii'} - r^{\mathcal{B}}_{ii'})^2 \\
&= \sum_{ii'}\left(\sum_j z^{\mathcal{A}}_{ij} z^{\mathcal{A}}_{i'j} - \sum_k z^{\mathcal{B}}_{ik} z^{\mathcal{B}}_{i'k}\right)^2 \\
&= \sum_{ii'}\left[\sum_{jj'} z^{\mathcal{A}}_{ij} z^{\mathcal{A}}_{i'j} z^{\mathcal{A}}_{ij'} z^{\mathcal{A}}_{i'j'} + \sum_{j'k'} z^{\mathcal{B}}_{ik} z^{\mathcal{B}}_{i'k} z^{\mathcal{B}}_{ik'} z^{\mathcal{B}}_{i'k'} - 2\sum_{jk} z^{\mathcal{A}}_{ij} z^{\mathcal{A}}_{i'j} z^{\mathcal{B}}_{ik} z^{\mathcal{B}}_{i'k}\right] \\
&= \sum_{jj'} \delta_{jj'}\, n^{\mathcal{A}}_j \delta_{jj'}\, n^{\mathcal{A}}_j + \sum_{kk'} \delta_{kk'}\, n^{\mathcal{B}}_k \delta_{kk'}\, n^{\mathcal{B}}_k - 2\sum_{jk}(n^{\mathcal{A}\mathcal{B}}_{jk})^2 \\
&= \sum_j (n^{\mathcal{A}}_j)^2 + \sum_k (n^{\mathcal{B}}_k)^2 - 2\sum_{jk}\left(n^{\mathcal{A}\mathcal{B}}_{jk}\right)^2
\end{aligned}
$$

and the second from $D^P_{\mathcal{A},\mathcal{B}} = \mathrm{Tr}(P^{\mathcal{A}}) + \mathrm{Tr}(P^{\mathcal{B}}) - 2\mathrm{Tr}(P^{\mathcal{A}}P^{\mathcal{B}}) = a + b - 2\mathrm{Tr}(P^{\mathcal{A}}S^{\mathcal{B}})$ and

$$
\mathrm{Tr}(P^{\mathcal{A}}P^{\mathcal{B}}) = \sum_{ii'}\sum_{jk} \frac{z^{\mathcal{A}}_{ij} z^{\mathcal{A}}_{i'j}}{n^{\mathcal{A}}_j} \frac{z^{\mathcal{A}}_{i'k} z^{\mathcal{A}}_{ik}}{n^{\mathcal{B}}_k} = \sum_{jk} \frac{(n^{\mathcal{A},\mathcal{B}}_{jk})^2}{n^{\mathcal{A}}_j n^{\mathcal{B}}_k} = 1 + \frac{1}{n}\chi^2_{\mathcal{A},\mathcal{B}}
$$

■

5.4 The crisp case: instability of a group relatively to another partition

Let $\mathcal{A}$ and $\mathcal{B}$ be two crisp partitions whose non-empty groups are respectively indexed by $j = 1, \ldots, a$ and $k = 1, \ldots, b$; let $n_j := n^{\mathcal{A}}_j > 0$ denote the number of objects $i \in j$.

Theorem 7

$$
D^R_{\mathcal{A},\mathcal{B}} = \sum_{j=1}^{a} \rho^{\mathcal{B}}_j \qquad \rho^{\mathcal{B}}_j := n_j^2 - 2\alpha^{\mathcal{B}}_j + \beta^{\mathcal{B}}_j \qquad \alpha^{\mathcal{B}}_j := \sum_{i,i' \in j} r^{\mathcal{B}}_{ii'} \qquad \beta^{\mathcal{B}}_j := \sum_{i \in j; i'} r^{\mathcal{B}}_{ii'} \quad (12)
$$

$$
D^T_{\mathcal{A},\mathcal{B}} = D^P_{\mathcal{A},\mathcal{B}} = \sum_{j=1}^{a} \tau^{\mathcal{B}}_j \quad \tau^{\mathcal{B}}_j := 1 - 2\gamma^{\mathcal{B}}_j + \delta^{\mathcal{B}}_j \quad \gamma^{\mathcal{B}}_j := \frac{1}{n_j}\sum_{i,i' \in j} p^{\mathcal{B}}_{ii'} \quad \delta^{\mathcal{B}}_j := \sum_{i \in j} p^{\mathcal{B}}_{ii} \quad (13)
$$

$\rho^{\mathcal{B}}_j$ and $\tau^{\mathcal{B}}_j$ constitute measures of the *instability* of group j (of partition $\mathcal{A}$) relatively to partition $\mathcal{B}$; by construction, their sum over the groups $j = 1, \ldots, a$ yields the (squared) distance between partitions $\mathcal{A}$ and $\mathcal{B}$. Note that:

- n_j^2 is the number of (distinct or not) pairs of objects in j
- $\alpha^{\mathcal{B}}_j$ is the number of pairs in j which are also classified in the same group k of $\mathcal{B}$
- $\beta^{\mathcal{B}}_j$ is the number of pairs classified in the same group k of $\mathcal{B}$, such that the first

object of the pair belongs to j.

As $n_j^2 \geq \alpha_j^{\mathcal{B}}$ and $\beta_j^{\mathcal{B}} \geq \alpha_j^{\mathcal{B}}$, one has $\rho_j^{\mathcal{B}} \geq 0$ with equality if and only if $n_j^2 = \alpha_j^{\mathcal{B}}$ (all pairs in j are pairs for $\mathcal{B}$) and $\beta_j^{\mathcal{B}} = \alpha_j^{\mathcal{B}}$ (all pairs (i, i') for $\mathcal{B}$ such that $i \in j$ satisfy $i' \in j$). Also:

- $\gamma_j^{\mathcal{B}}$ is a measure of the pair cohesion in $\mathcal{B}$ "as seen from j"
- $\delta_j^{\mathcal{B}}$ is a measure of the fineness of groups of $\mathcal{B}$ "as seen from j".

Properties $p_{ii'}^{\mathcal{B}} \leq p_{ii}^{\mathcal{B}}$ and $\sum_{i'} p_{ii'}^{\mathcal{A}} = 1$ entail $\gamma_j^{\mathcal{B}} \leq \delta_j^{\mathcal{B}}$ and $\gamma_j^{\mathcal{B}} \leq 1$, and thus $\tau_j^{\mathcal{B}} \geq 0$.

Proof.

$$\begin{aligned} D_{\mathcal{A},\mathcal{B}}^R &= \sum_{ii'} (r_{ii'}^{\mathcal{A}} - r_{ii'}^{\mathcal{B}})^2 \\ &= \sum_j \sum_{i \in j} \sum_{i'} \left[r_{ii'}^{\mathcal{A}} - 2 r_{ii'}^{\mathcal{A}} r_{ii'}^{\mathcal{B}} + r_{ii'}^{\mathcal{B}} \right] \\ &= \sum_j \left[n_j^2 - 2 \sum_{i,i' \in j} r_{ii'}^{\mathcal{B}} + \sum_{i \in j; i'} r_{ii'}^{\mathcal{B}} \right] \end{aligned}$$

$$\begin{aligned} D_{\mathcal{A},\mathcal{B}}^T = \sum_{ii'} (p_{ii'}^{\mathcal{A}} - p_{ii'}^{\mathcal{B}})^2 &= \sum_j \left[\sum_{i \in j} p_{ii}^{\mathcal{A}} - 2 \sum_{i \in j} p_{ii'}^{\mathcal{A}} p_{ii'}^{\mathcal{B}} + \sum_{i \in j} p_{ii}^{\mathcal{B}} \right] \\ &= \sum_j \left[1 - \frac{2}{n_j} \sum_{i,i' \in j} p_{ii'}^{\mathcal{B}} + \sum_{i \in j} p_{ii}^{\mathcal{B}} \right] \end{aligned}$$

where $(r_{ii'}^{\mathcal{A}})^2 = r_{ii'}^{\mathcal{A}}$ and $\sum_{i'} (p_{ii'}^{\mathcal{A}})^2 = p_{ii}^{\mathcal{A}} = 1/n_{j(i)}$ have been used. ■

Example 3 *The instability of class j with regard to the n-groups partition $\mathcal{B} = \mathcal{N}$ is:*

- $\rho_j^{\mathcal{N}} = n_j \ (n_j - 1)$ (*large* groups are unstable)
- $\tau_j^{\mathcal{N}} = n_j - 1$ (*large* groups are unstable)

Its instability with regard to the one-group partition $\mathcal{B} = \mathcal{O}$ is:

- $\rho_j^{\mathcal{O}} = (n - n_j) \ n_j$ (*medium* groups are unstable)
- $\tau_j^{\mathcal{O}} = \frac{n - n_j}{n} = 1 - f_j$ (*small* groups are unstable).

Aknowledgements: The work has benefited from stimulating discussions with M. Rajman in the framework of the joint UNIL-EPFL "Clavis" project (2001).

References

[1] Bavaud, F. (2002). Quotient Dissimilarities, Euclidean Embeddability, and Huygens' Weak Principle. In K. Jajuga and al. (Eds.). *Classification, Clustering, and Data Analysis.* Springer, 195-202.

[2] Bezdek, J. (1981). *Pattern recognition with Fuzzy Objective Function Algorithms*, Plenum Press.

[3] Gower, J.C. (1982). Euclidean distance geometry. *The Mathematical Scientist*, 7, 1-14.

[4] Marcotorchino, J.F. (2000). Dualité Burt-Condorcet: relation entre analyse factorielle des correspondances et analyse relationnelle. In J. Moreau and al. (Eds.). *L'analyse des correspondances et les techniques connexes.* Springer, 211-227.

[5] Mirkin, B. (1996). *Mathematical Classification and Clustering*, Kluwer.

[6] Mirkin, B. and Cherny, L.B. (1970). On a distance measure between partitions of a finite set, *Automation and Remote Control*, 31, 786-792.

[7] Rand, W.M. (1971). Objective criteria for the evaluation of clustering methods, *Journal of the American Statistical Association*, 66, 846-850.

[8] Saporta, G. (1990). *Probabilités, analyse de données et statistique*, Editions Technip, Paris.

[9] Schoenberg, I.J. (1935). "Remarks to Maurice Fréchet's article "Sur la définition axiomatique d'une classe d'espaces vectoriels distancés applicables vectoriellement sur l'espace de Hilbert"". *Annals of Mathematics*, 36, 724-732.

Student (2004), Vol. 5 No. 1, 43-58

Clustering under Connectivity Hypothesis

Catherine Aaron

SAMOS-MATISSE
Université Paris I
90 Rue de Tolbiac
F-75013 Paris

Abstract: The aim of this paper is to show the theoretical interest of the connectivity concept in clustering and modelling. We first show that the well known hierarchical algorithm with minimum distance is coherent with connectivity hypothesis. We build a within-cluster distance which is linked with connectivity notion and show that it is the one that is minimized by our clustering algorithm. Then we compare the power of our new distance for the choice of the number of clusters with the quality of classical within-cluster distance based on Euclidian metric.

Key words: Un-supervised clustering, connectivity, number of clusters, gap-statistic.

1 Introduction: Connectivity, clustering and modelling

Let (E, d) be a metric space and X a subset of E. We assume that the connected components of X are, topologically, the best partition of it. As connectivity cannot be transposed to finite sets we will use path-connectivity concept. In a clustering point of vue the path-connectivity concept is the non-linear matching of convexity: "two points can be linked with a continuous path instead of "the linear path linking two points is include in the set".

In a modelling purpose where the problem is to find a continuous function f from E to $F = f(E)$, connectivity concept also has its interest: If E is not connected then we may have a connecting problem as shown in Figure 1:

Then if such a function exists the sets $F = f(E)$ and $E \times f(E)$ are connected. The converse is false but the connectivity of $E \times f(E)$ and E implies that the set of

Figure 1. Examples of union of connected but not convex sets

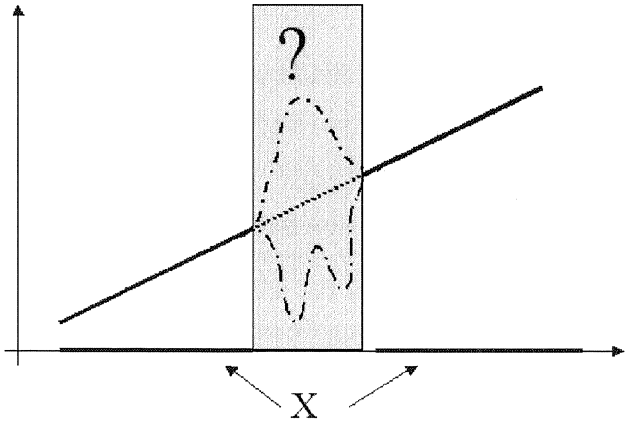


Figure 2. connecting problem if X is not connected

uncontinuity points of f doesn't cut E in several connected clusters. Then if $E \in \mathbb{R}$ the connectivity of $E \times f(E)$ is equivalent to the continuity of f.

The aim of this article is then to give a method to separate a set into its connected component. The modelling application, for instance, dictates us an un-supervised method that gives as well clusters and number of clusters.

With only the connectivity hypothesis we cannot use all the clustering algorithms that are based on linear separation, variance minimisation or grouping around barycenters. Same examples as that in Figure 1 easily illustrate this point. So methods as K-means, hierarchical clustering with Ward distance or ants algorithm may fail for finding connected clusters. Non linear methods as neural ones imply an idea on the number of clusters. They may be used whenever we get an indication on it but are useless before. So we have to build a method that leads us to know the number of clusters and the associated clusters.

2 Preliminary

2.1 Connectivity and finite sets

Definition 1 (Connectivity) *The Connectivity notion is defined by one the four following properties:* (X, d) *is connected if and only if:*

i) if $X = E \cup F$ where E and F are open sets and $E \cap F = \emptyset$ then $E = X$ and $F = \emptyset$ (or $F = X$ and $E = \emptyset$);

ii) if $X = E \cup F$ where E and F are closed sets and $E \cap F = \emptyset$ then $E = X$ and $F = \emptyset$ (or $F = X$ and $E = \emptyset$);

iii) for all continuous function f from X to $0, 1$ f is constant;

iv) the only open and closed subsets of X are X and $\emptyset$.

We cannot adapt this definition to finite sets because finite sets never are open. But the path-connectivity notion, which implies connectivity and which is defined by the following assertion will be easier to adapt.

Definition 2 (Path-connectivity) (X, d) *is path-connected if and only if for all* $(x, y) \in E^2$ *exists* $u \in C^0([0, 1], E)$ *such that* $u(0) = x$ *and* $u(1) = y$, *that means that a continuous path which links* x *to* y *exists.*

We can adapt to finite sets by:

Definition 3 ($\delta-$path connectivity) *E is $\delta-$path connected if and only all couples of points (x, y) can be linked by a $\delta-$path of points of E.*

A $\delta-$path linking x to y is a finite sequence of points of E, $x_1, \ldots, x_k$ with $x_1 = x$, $x_k = y$ and $\forall p$, $d(x_p, x_{p+1}) < \delta$.

For ease of writing we are going to speak of $\delta-$connectivity and not any more of $\delta-$path connectivity.

2.2 Properties

Let E be a finite set of cardinal n. For a fixed δ, there exists an unique maximal clustering $\delta-$connected (maximal means here that the number of clusters is minimal).

Unicity: Let us assume that there exist two maximal clustering with p clusters: $\{F_1, \ldots, F_p\}$ and $\{G_1, \ldots, G_p\}$. Then there exists x in two different classes (let assume that $x \in F_1$, $x \in G_1$ and $G_1 \subsetneq F_1$). Then there exists $y \in G_1$, $y \in F_i$ with $i \neq 1$: let assume for exemple that $i = 2$. Then $(F_1 \cup F_2, F_3, \ldots, F_p)$ is a $\delta-$connected clustering with $p - 1$ clusters. So the initial clustering were not maximal.

Existence: The function f which, for a clustering, associates the number of clusters, gets its values in $\{1, \ldots, n\}$ thus a minimum is reached.

We can define $p(\delta)$ the function which associates the number of clusters of the maximal $\delta-$connected clustering. This function decreases from n to 0 and is a stair function.
We note:
$\delta_{\max}(p) = \sup\{\delta, p(\delta) = p\}$
$\delta_{\min}(p) = \inf\{\delta, p(\delta) = p\}$
with: $\delta_{\max}(p) = \delta_{\min}(p-1)$

3 Clustering

3.1 Hierarchical algorithm for clustering

The hierarchical clustering algorithm with the minimum distance

$$d(A, B) = \min\{d(a, b)\ , a \in A\ , b \in B\}$$

computes all the $\delta-$connected clustering.
In fact, in order to realize the maximal $\delta-$connected clustering with $p - 1$ clusters $((C_1^{p-1}, \ldots, C_{p-1}^{p-1}))$ when we get the maximal $\delta-$connected clustering with p clusters $(C_1^p, \ldots, C_p^p)$ we have to aggregate the two closest classes for the minimum distance. If $m = \min(d(C_i^p, C_j^p))$ the new clustering is now $m-$connected and, $\delta_{\min}(p-1) = \delta_{\max}(p) = m$.

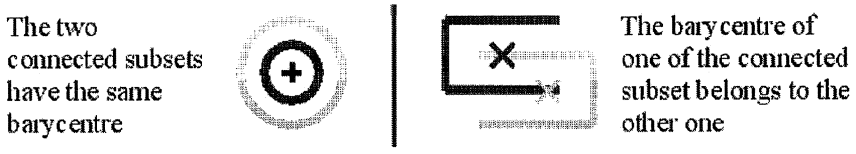


Figure 3. Weakness of barycentre concept with connectivity notion

3.2 Within-cluster distance

Here we have seen that the well known algorithm of hierarchical clustering with minimum distance gives connected classes. The main part of our work consists to build an intra-class distance linked to the connectivity concepts that helps us to determine a "good" number of clusters.
First let notice that the concepts of inertia or barycentres don't make any sense if we have connected but not convex classes as illustrated in the Figure 3.

3.2.1 Distance between two points

An Euclidian distance between two points A and B doesn't give any clue on the "level" of connectivity that links A and B. In the same way every distance that doesn't use the other points of the set to be computed is useless. The following picture illustrates this purpose: the two points have the same Euclidian distance but the connected situations are obviously different.
To solve this problem, we will define, as distance between two points of E the minimum of the greatest gap observed when we link the point with a path of points of E.
A path w of points of E linking x_i to x_j is an application from $\{1, \ldots, n\}$ to itself that only verifies $w(1) = i$ and $w(n) = j$.
Let note $W(x_i, x_j)$ the set of ways of points of E linking x_i to x_j.
On a fixed way w, the greatest gap is: $\text{gap}_{\max}(x_i, x_j, w) = \max(d(x_{w(i)}, x_{w(i+1)}))$
Then, the new distance, associated with the initial metric d is:

$$d_c(x, y) = \min(\text{gap}_{\max}(x_i; x_j, w))$$

We can verify that we have defined an ultra-metric in a finite set where all points are different:
$d_c(x, y) = 0 \Rightarrow x = y$ in a finite space if all points are different (elsewhere that would be true).
$d_c(y, x) = d_c(x, y)$ obviously if we reverse the path.
$d_c(x, z) \leq \max(d_c(x, y), d_c(y, z))$ because on the right side of the inequality the paths are forced to pass by the y point.

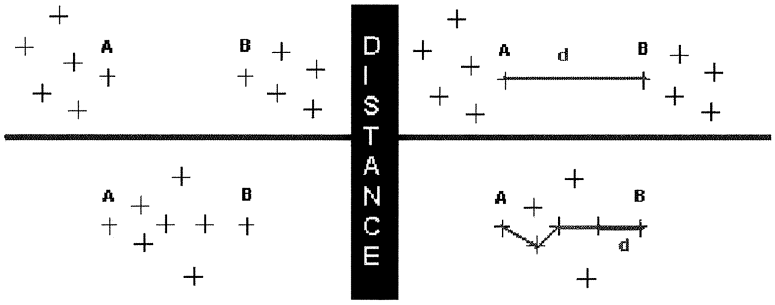


Figure 4. Distance and connected spaces

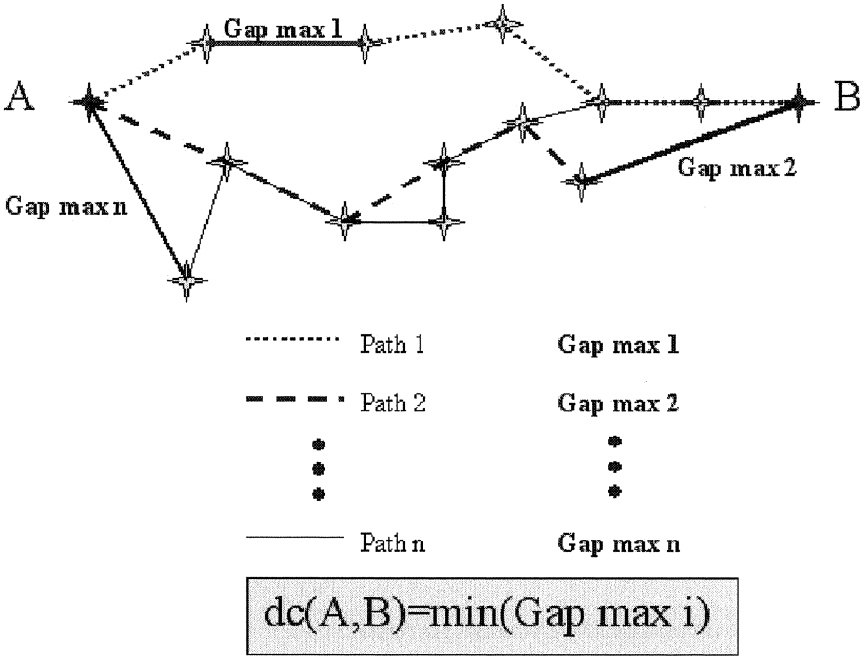


Figure 5. Distance between two points

3.2.2 Characterization

For all pairs of points the number of path linking them is: $\sum_{k=1}^{n-2} A_n^k$ with $A_n^k = n!/(n-p)!$
Thus obviously we cannot calculate the distance between two points computing all the possible ways. The computation time would be too long. But, we can notice that, at the end of the hierarchical clustering, we get a quick way to compute all the distances with the following algorithm.
Let us note $C_k(x)$ the cluster of x if the clustering has k classes:

$$d_c(x,y) = \delta_{\min}(\max(k/C_k(x) = C_k(y)))$$

or

$$d_c(x,y) = \delta_{\max}(\min(k/C_k(x) \not= C_k(y))).$$

3.2.3 Proof

$\delta_{\min}(\max(k/C_k(x) = C_k(y))) = \delta_{\max}(\min(k/C_k(x) \neq C_k(y)))$ because $\max(k/C_k(x) = C_k(y)) = \min(k/C_k(x) \neq C_k(y)) + 1$ and $\delta_{\min}(p+1) = \delta_{\max}(p)$ and:
$d_c(x,y) \leq \delta_{\min}(\max(k/C_k(x) = C_k(y)))$ because:
$\forall \delta \geq \delta_{\min}(\max(k/C_k(x) = C_k(y)))$ x and y are in the same $\delta-$connexe cluster so they can be linked by a $\delta-$way so $d_c \leq \delta$.
In the same way $d_c(x,y) \geq \delta_{\max}(\min(k/C_k(x) = C_k(y)))$ because:
$\forall \delta \leq \delta_{\max}(\min(k/C_k(x) \neq C_k(y)))$ x and y are not in the same $\delta-$connexe cluster so they cannot be linked by a $\delta-$way so $d_c \geq \delta$.

Then we can see that the set of distances between the pairs of points has only $n-1$ values instead of $n(n-1)/2$ values.

3.2.4 Within-cluster distance

The within-cluster distance is the mean value of d_c calculated for the pairs of points which are in the same cluster. Let note $D_c(p)$ the within-cluster distance for a clustering in p clusters:

$$D_c(p) = (1/N_p) \sum_{i,j} \mathbf{1}_{\{C_p(x_i)=C_p(x_j)\}} d_c(x_i,x_j) \quad \text{with} \quad N_p = \sum_{i,j} \mathbf{1}_{\{C_p(x_i)=C_p(x_j)\}}$$

The previous characterization of d_c with the issue of the hierarchical clustering shows that the hierarchical clustering algorithm with minimum distance is the algorithm which minimizes the intra-class distance D_c for a fixed number of clusters.

3.2.5 Link with graph theory

In fact at the issue of the hierarchical classification with the minimum distance we can build the minimum connected graph linking all the points of the set. Then for all pairs of points there is only one path linking them (on the graph) and the associated

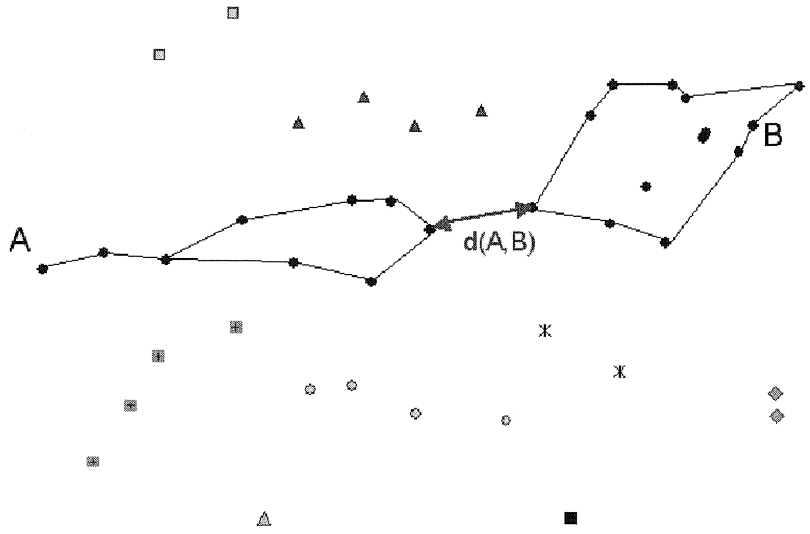


Figure 6. Characterization to compute distance. A and B are in the same cluster for a 9 classes clustering; A and B are separated in a 10 cluster clustering.

connected distance between point is the maximum gap observed when walking on the graph.

3.3 Choosing the number of clusters

3.3.1 Principle

In a classical way we are going to choose the clustering that gives the greatest variation of within-cluster distance which is interpreted as the greatest change of structure. We observe the within-cluster distance's diagram in order to identify the greatest value of $G(k) = (D_c(k) - D_c(k-1))$ which is known as the "elbow method". Other indicators which only used within-cluster distance exist as, for instance:

- $KL(k) = W_k/W_{k+1}$, with $W_k = (k-1)^{2/p}W_{k-1} - k^{2/p}W_k$ (Krzanowsky and Lai, 1985).
- $H(k) = (D_c(k) - D_c(k-1))/(D_c(k-1)(n-k-1))$ (Hartigan, 1975).

The Gap statistic method with usual within-cluster distance, based on comparison of $G(k)$ for the tested set and $G_{ref}(k)$ for uniform sets is a way to "normalize" within-class distance that help to choose the number of clusters and that seems to outperform other indicators when cluster are convex [5]. We are going to construct a method to select the number of cluster based on the same idea of comparison with uniform data.

3.3.2 Testing the connectivity with comparison with uniform data

Ideally we should randomize N points inside the convex envelope of X. As it is too difficult and long to compute, especially for dimension higher than 2 we proceed as follows: We compute the principle axes of the data-set X.
We compute the bounding box of X.
For a number N_{mc} of time we randomize a set Y as N points uniformly inside the bounding box.

For each Y we compute $G_Y(.)$ the within-cluster variation.
We compute $G_{\text{unif}}(.)$ as $G_{\text{unif}}(k) = \max(G_Y(k))$.
Finally we calculate $G_{\text{ref}}(.)$ as $G_{\text{ref}}(k) = \max\{G_{\text{unif}}(k'), k' \geq k\}$.

We also calculate $G_X(.)$ the within-cluster variation of the initial set X and decide to retain a number k of cluster when $G_X(k) \geq G_{\text{ref}}(k)$.

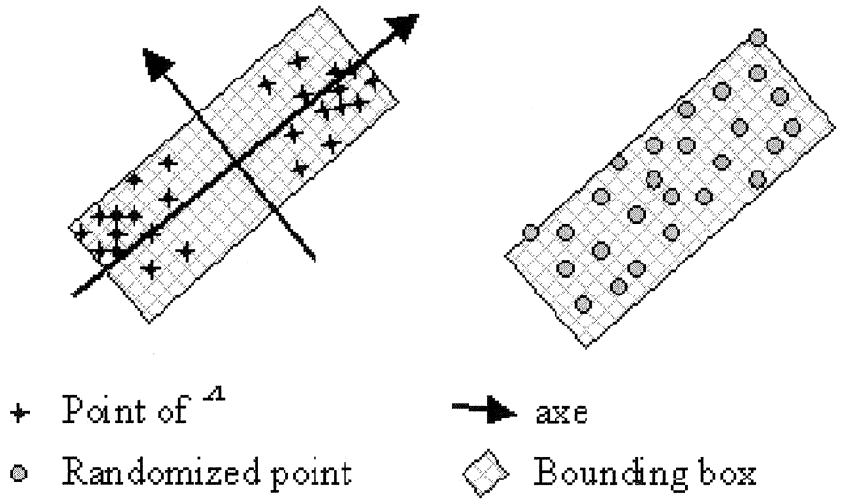


Figure 7. Randomize point

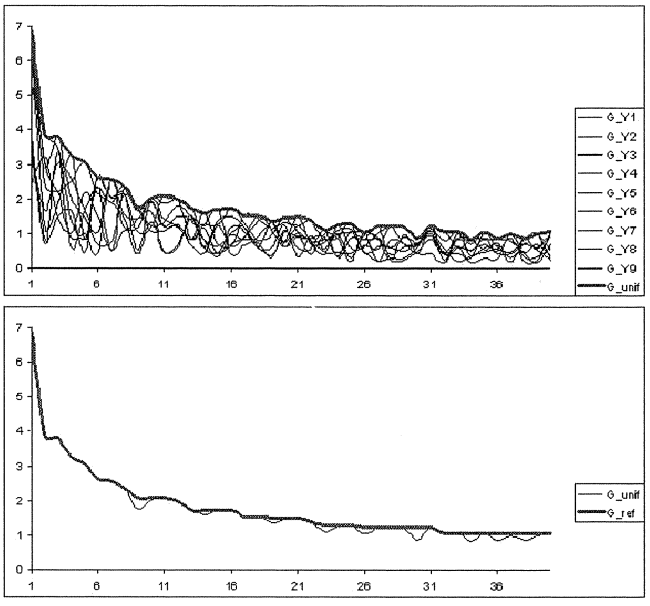


Figure 8. Construction of $G_{\text{unif}}(.)$.

4 Some clustering results

Our principal work here is the construction of the within-cluster distance linked to the connectivity notion. So we will, in this section compare results obtained using our within-cluster distance and the usual one (Euclidian). So we both calculate within-cluster distance's variation for the connected distance and for the Euclidian distance. Then we chose the number of cluster with the comparison with uniform sets described previously. In all the following examples $N_{mc} = 100$.

4.1 Partition of Gaussian sets

We are going to test the method to separate two classes issued of Gaussian random drawing. The variances of Gaussian density are: 50 points are centered on $(0, 0)$ and 50 other points are centered on $(0, m)$. Let first remark that our method, based only on connectivity should not outperform other existing method for un-supervised clustering because these ones use convexity hypothesis which is a stronger hypothesis and which is verified with Gaussian sets.
Let remark that for $m \leq 2$ underlying density has an unique modality so clustering has non sense. We can observe here than Euclidian within cluster distance is better than connected one, each time it recognizes an absence of clustering.

On the other hand, when $m \geq 2$ we have bi-modality and clustering has sense. Here we can observe that connected within cluster distance is better than Euclidian distance. We recognize interest in clustering since $m = 3$ and we can see that, even if we find too many clusters, there is two principal clusters and little satellite ones. In the same time Euclidian within cluster distance needs $m = 5$ to begin to be significant.
Generally we observe that connected within-cluster distance use to cluster data more than Euclidian distance.

4.2 Connected but not convex sets

We work here on a data base grouping 2 subsets formed by concentric cercles. One is issued from a uniform distribution between radius 1 and radius $1, 2$ (100 points) and the other one is issued from a centered Gaussian distribution and with variable standard deviation $(0.05, 0.1, 0.15, 0.20)$ (100 points).
In all the cases clusters are visually very well separated. The Euclidian within-cluster distance doesn't find any clustering. In the other hand connected within-cluster distance finds the "good" number of clusters.

4.3 Acceleration with preliminary K-means

Time for computation is quite long for our algorithm. Hierarchical clustering has a computational time of N^3 and we have to do it one time for the tested data-set and

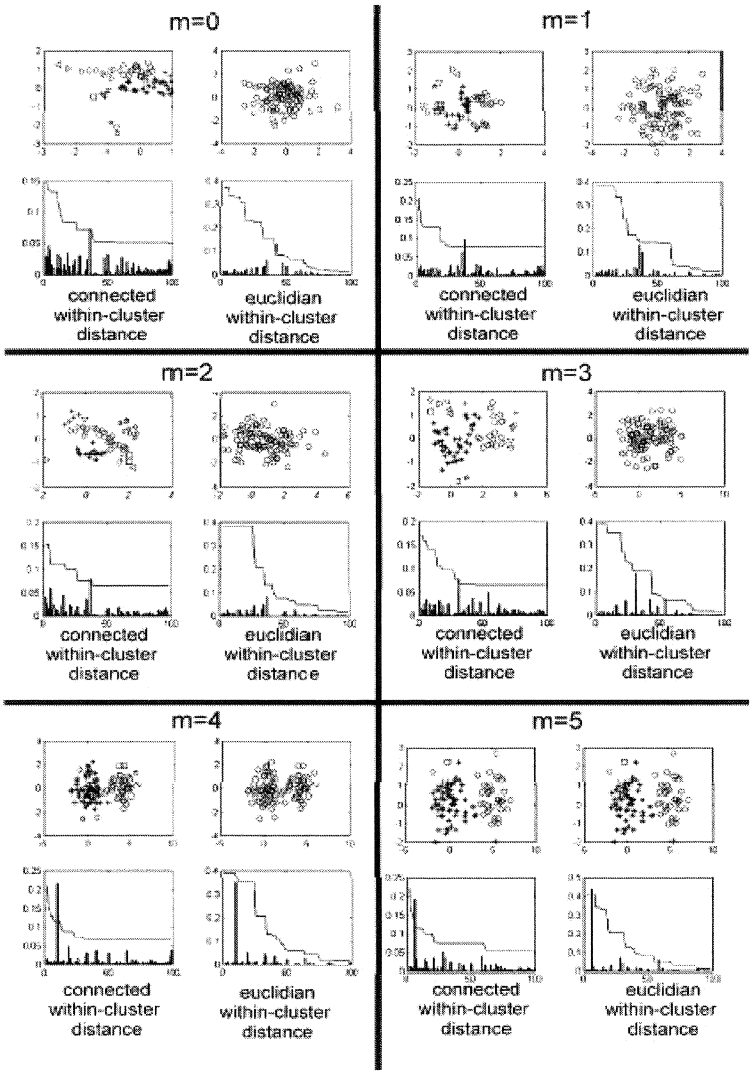


Figure 9. Clustering of Gaussian sets: In the first part of each figure only the 12 first clusters are represented because of a lack of symbols. In the second part the plain line represents G_{unif} and bars represents within cluster distance.

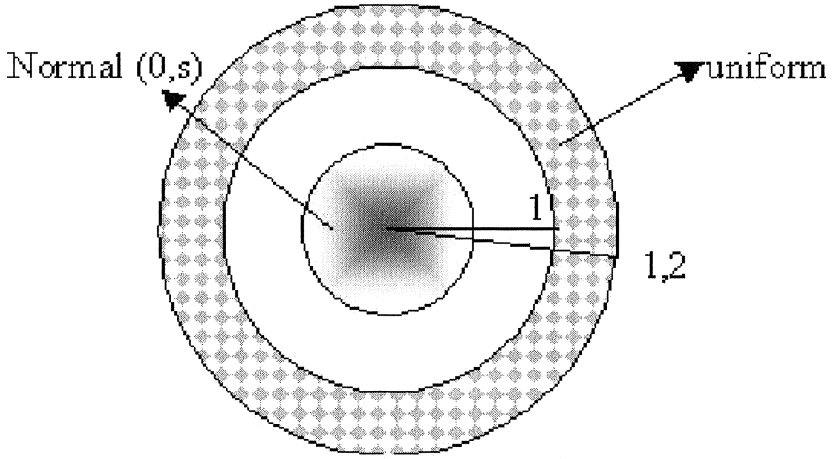


Figure 10. Simulated data.

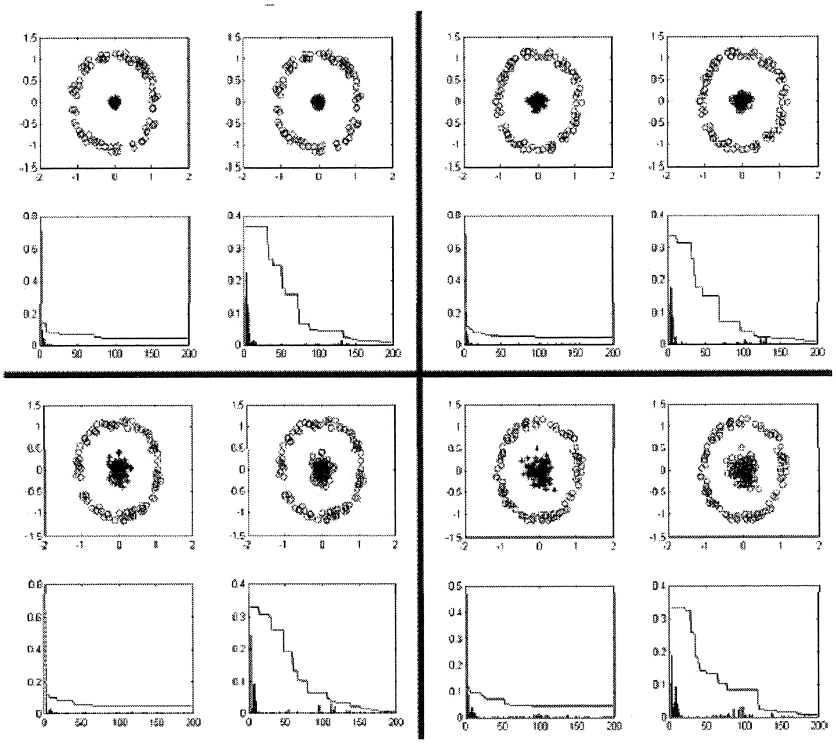


Figure 11. Simulated data.

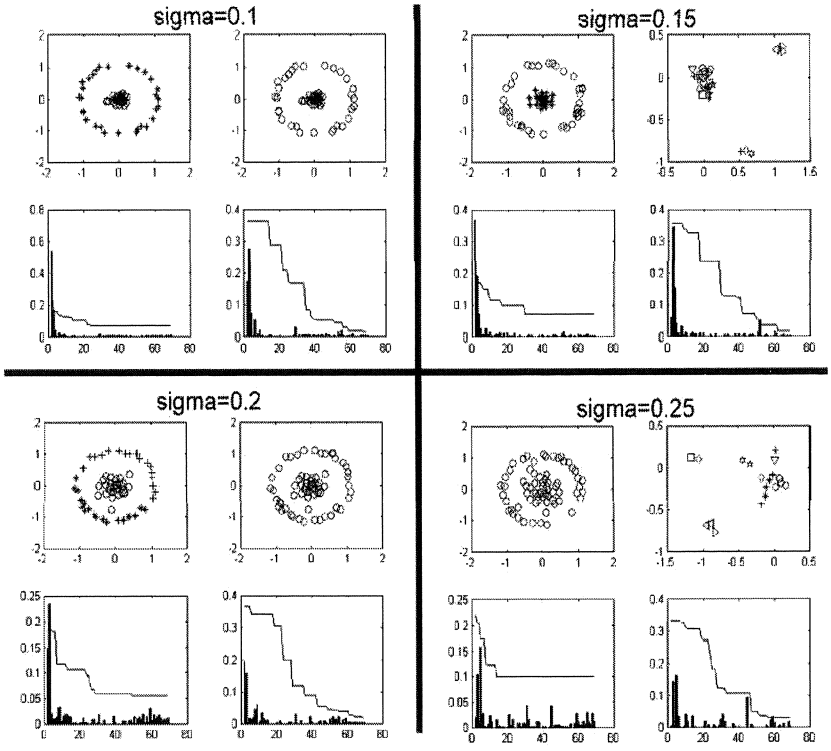


Figure 12. Simulated data and preliminary K-means.

N_{mc} times for the construction of $G_{unif}(.)$ (in previous examples $N_{mc} = 100$). So a reduction of the number of initial points may be really useful to accelerate the process. We tested on the previous data-set made of concentric cercles a 70-means algorithm to reduce the number of points from 200 to 70 then we applied our method on the 70 points.

In the two first cases the connected within-cluster distance find exactly the good number of clusters when Euclidian within-cluster distance do not find clusters. When standard deviation increases some difficulties appear in the third cases the bigger cercle is cut and in the fourth ones no clusters are found.

5 Conclusion

If the "true" classes are homogeneous (in term of "connected" dispersion inside the classes) and well separated the previous algorithm gives quite good results as well for the number of classes (if we except isolated points) and for the found classes.

Problems exists when classes are heterogeneous. In extreme cases it can happen that we have to isolate all points of a class. If we are in this case but with convex clusters usual methods like K-means are better than ours. So it would be interesting to construct a test of convexity for the clusters. if verified we should prefer classical linear methods.
But, when classes are "exotic" and only characterized by the connectivity, our method open perspectives for un-supervised clustering.
A great problem lies in cluster analysis. If we only get connectivity the usual representation of a cluster with its barycentre is not justified so we have now to build efficient centroids.

References

[1] Coeurjolly D. (2002). Algorithmique et géométrie discrète pour la visualisation des courbes et des surfaces, *Thesis.*
[2] Gordon A., (1996). Hiearchical classification, *Clustering and Classification World Scientific Publ, River Edge, NJ,* 65–121.
[3] De Soete G., Carrol D. (1996). Tree and other network model for representing proximity data, *Clustering and Classification World Scientific Publ, River Edge,NJ,* 65–121.
[4] Tchoumatchenko K., Rousset P. (1997). Modélisation de réseaux de communication par la géométrie stochastique, *Thesis.*
[5] Tibshirani R., Guenther W., Hastie T. (2001). Estimating the number of data clusters via the gap statistic, *J. R. Statistic Society B(2001),* 63, 411–423.
[6] Ripley B. (1981), Spatial statistics, *Wiley series in probability and mathematical statistics.*

Student (2004), Vol. 5 No. 1, 59-75

Mining Association Rules with Multiple Min-supports - Application to Symbolic Data

Tao Wan and Karine Zeitouni

PRISM Laboratory
University of Versailles
45 avenue des Etats-Unis,
F-78035 Versailles Cedex, France

Abstract: Conventional data mining techniques apply well to simple data types. However, as more and more complex data are produced such as spatial or multimedia datasets, traditional techniques can no longer fulfil their analysis. Symbolic data is a new data description paradigm where the data are more complex than the tabular form. It holds more structured data having internal variations such as value distributions, and rules such as taxonomies. Since its introduction [1], the problem of mining association rules from large databases has been the subject of numerous studies. Popular methods of mining association rules involve finding all frequent itemsets satisfying a uniform support threshold. The uniform min-support approach, however, has some limitations if some important items are unlikely to occur as frequently as the others. This paper proposes a new method that aims to mine inter-dimensional association rules from symbolic data described by categorical multi-valued variables, while allowing specifying different min-supports for certain dimensions depending on their interest within actual application.

Key words: Data mining, symbolic data, categorical multi-valued variables, mining frequent itemsets.

1 Introduction

As a new data description paradigm, symbolic data gives more complex data structures than classical or mono-valued data. It is better adapted to the nowadays explosively propagating data sources. In this context, we address the problem of association rules mining that is different than that in conventional data analysis. Conventional association rules mining requires finding all frequent itemsets that satisfy a uniform *min-support*. This limitation frequently results in specifying a high *min-support* that

idAccident	Gravity
Acc1	Light
Acc2	Serious

idAccident	AccidentNeighbourDistance	Neighbour
Acc1	5	Metro
Acc1	6	School
Acc2	15	School
Acc2	17	School
Acc2	22	Market

idAccident	Gravity	AccidentNeighbour Distance	Neighbour
Acc1	Light	5	Metro
Acc1	Light	6	School
Acc2	Serious	15	School
Acc2	Serious	17	School
Acc2	Serious	22	Market

Table 1: ***Relational table obtained by joining the two above tables***

idAccident	Gravity	AccidentNeighbour Distance	Neighbour
Acc1	Light	[5, 6]	Metro (50%), School (50%)
Acc2	Serious	[15, 22]	School (66.7%), Market (33.3%)

Table 2: ***Symbolic data table obtained from the two above tables***

Figure 1. A relational and symbolic data table arisen from two relational tables.

will miss interesting but less frequent patterns, as well as reducing *min-support* risks to produce a large result difficult to filter by the user. Mining association rules with *multiple min-supports* from symbolic data is an efficient algorithm that aims to:

(1) Deal with complex data such as symbolic data where no equivalent method has been previously developed.

(2) Fulfill, more precisely, the users' requirements by allowing them to specify *multiple min-supports* within the application.

(3) Achieve a good performance due to existing efficient algorithm FP-growth.

This section introduces the context of this method and emphasizes its motivation.

1.1 Symbolic data

According to [12], a symbolic data table is an extension of a relational table where attribute values may contain distributions, intervals or several values linked by taxonomy and logical rules. As input, symbolic data can arise from many source types and aggregate large datasets according to their underlying concepts. Hence, the symbolic data concept allows us to summarize large and complex databases more naturally, with greater richness and a more manageable size. Fig. 1 shows a simple example of a symbolic data table generation procedure from some relational tables describing road accidents in their neighboring environments.

In Table 2, we can see that each individual (e.g. Acc1, Acc2) is described by only one summary tuple and attributes can be of interval type or multiple values linked by

their distribution, while in Table 1, each individual requires many tuples. Therefore, the size of the symbolic table is much smaller. Furthermore, this summary table captures more detailed information than what is obtained by relational aggregates, as it aggregates categorical attributes (such as neighbor) and keeps internal variation and numerical range. This is why the symbolic data paradigm allows us to summarize large and complex databases more naturally and effectively.

"Symbolic Data Analysis" (SDA) [12] is an extension of standard data analysis. It extends many data analysis and data mining techniques towards symbolic data.

1.2 Mining Association rules

Since its introduction [1], the problem of mining association rules from large databases has been the subject of numerous studies. Those studies can be broadly divided into three categories, as summarized in [26]:

Scalability: the central question is how to compute the conventional association rules as efficiently as possible. Studies in this category can be further classified into three subgroups: (i) fast algorithms based on the level wise Apriori framework [3, 11]; (ii) partitioning [17, 19] and sampling; and (iii) incremental updating and parallel algorithms [2, 6, 8, 16].

Extensibility: the central question is how to extend association rules mining. Studies in this category can be further classified into two generations. Studies in the first generation attempt to extend this method to different data types and applications, e.g. mining association rules from time series data [22], web data [23], complex data [27, 28], while studies in the second generation apply larger logic formalism, such as datalog queries in [24].

Functionality: the central question is what kind of rules to compute. Studies in this category can be further classified into two types. Studies in the first type basically considered the data mining exercise in isolation. Examples include multi-level association rules [9], quantitative and multi-dimensional rules [7, 14, 21], correlations and causal structures [5, 20], mining long patterns [4], and ratio rules [12]. Studies in the second type have explored how data mining can best interact with other key components in the broader framework of knowledge discovery. One key component is the Database Management System (DBMS), and some studies explored how association rule mining can handshake with the DBMS most effectively. For instance, the integration of association rule mining with relational DBMS [13], or query flocks [18]. Another component, which is arguably even more important when it comes to knowledge discovery, is the *human user*. Studies of this kind try to provide much better support for (i) user interaction: e.g. dynamical change of parameters midstream; and (ii) user guidance and focus: e.g. limit of the computation to what interests the user.

This study contributes to the field by exploring association rules from complex symbolic data. It is a novel method that has never been treated before. In particular, it belongs to the category of functionality and extensibility, as it is centered

idAccident	Gravity	Distance	Neighbour
Acc1	Light	5	Metro
Acc1	Light	6	School
Acc2	Serious	15	School
Acc2	Serious	17	School
Acc2	Serious	22	Market

Table 1: ***Relational table of simple data***

Aggregate	*Most-frequent*	*Max*	*Most-frequent*
idAccident	Gravity	Distance	Neighbour
Acc1	Light	5	Metro
Acc2	Serious	22	School

Table 3: ***A classic aggregated table from table 1***

idAccident	Gravity	Distance	Neighbour
Acc1	Light	[5, 6]	Metro (50%), School (50%)
Acc2	Serious	[15, 22]	School (66.7%), Market (33.3%)

Table 2: ***Symbolic data table***

Figure 2. Comparison of a classic aggregated table and a symbolic data table.

upon the human user's application objectives by allowing the user to specify different *min-supports* from categorical multi-valued data in order to obtain more interesting information. Additionally, it avoids combinatorial explosion calculations.

1.3 Why mining association rules from symbolic data?

In data mining, we are sometimes interested in finding information from the aggregated data. For example, in analysing the shopping basket problem, to obtain a description of all customers we can analyse all the purchases of each customer, but not each customer's individual purchases. This is achieved by firstly preprocessing the data in order to aggregate them according to the users' requirements.

The advantage of symbolic data in relation to the classic aggregation methods is that they summarize large and complex databases and provide richer information. Continuing with the example provided in Table 1, Fig. 1, if we compile each accident by using classic aggregation, we lose more information than by using the symbolic data method, whose results are presented in Table 2.

In Fig. 2, Table 3 is a generated table from Table 1 by using classic aggregating operators while Table 2 is a symbolic table generated by using symbolic aggregation method. Very clearly, we can see that information in the table on the right is richer and more informative than that of Table 3.

Thus, if we extract association rules from symbolic data we obtain certain information that we are unable to find from using classic aggregated data. For example, in Table 2 of Fig. 2, it is possible to obtain a rule with "light", "metro" and "school" at the same time.

1.4 Why mining association rules with multiple min-supports?

A limitation of the uniform *min-support* approach is that some important items are unlikely to occur as frequently as others. Consequently, if the *min-support* is set too high, those rules that involve rare items will not be found. In order to find rules involving rare items and frequent items, the *min-support* has to be set very low, potentially causing a combinatorial explosion, as the frequent items will be associated with one another in all possible ways. Thus, this is why mining association rules with *multiple min-supports* is more informative. In most studies concerning association rules mining, the *min-support* is fixed. The two following references are exceptions:

(1) Mining multi-level association rules [9] is a method involving items at different levels of abstraction. It is an effective method that can produce interesting information from low or primitive levels of abstraction to high concept levels with potentially different *min-supports* between levels, while remaining the same for each level. However, as some combinations of items within a given level could be either very frequent or very rare, this method fails to find interesting rules. Indeed, specifying a high *min-support* will miss interesting but less rules, while reducing *min-support* risks to produce a large result difficult to filter by the user.

(2) In [15], the authors argued that using a single *min-support* for the whole database is inadequate because it cannot capture the inherent natures and/or frequency differences of the items in the database. For this reason, they proposed an association rules mining method with *multiple min-supports*. This method, based on "Apriori", extends the association rule model by allowing the user to specify *multiple min-supports* to better reflect the different natures and/or frequencies of items. By using these specified *multiple min-support*, we can prune certain itemsets in the phase of generating frequent patterns, while removing many rules in the phase of rules generation. In this extended model, the definition of association rules remains the same but the definition of *min-support* is changed, each item in the database can have a *min-support* called *MIS* (for Minimum Item Support) specified by the user. The authors' experiments proved that this model enables us to find rare item rules without producing a large number of meaningless rules with frequent items. To keep rare items, the *min-support* of an itemset is relaxed to the smallest MIS of all its items. However, by doing so, many non-meaningful frequent patterns are generated because they are likely to satisfy the *min-support* definition in [15] when they contain a rare item.

In our work, we borrow the concept of [15] "mining association rules with multiple min-supports" and its proposed model MIS, but we introduce a new definition for *frequent itemset*. This alternative definition is first explained by way of example 1.

Example 1 *Consider an itemset* $L = \{1, 2, 3\}$ *and* $MIS(1) = 10\%$, $MIS(2) = 20\%$, $MIS(3) = 30\%$. *L is frequent if and only if* $support(1) \geq 10\%$, $support(2) \geq 20\%$, $support(3) \geq 30\%$, $support(1, 2, 3) \geq 10\%$, *and* $support(2, 3) \geq 20\%$.

Definition 1 *(frequent itemset) An itemset L is frequent if and only if every subset of L satisfies the smallest MIS of its items.*

To summarize the difference between our approach and the one of [15], each MIS is directly taken into account in our definition whereas only the smallest MIS is considered in candidate generation in [15]. Hence, this definition allows us to reduce more unnecessary itemsets without losing much interesting information. Moreover, our definition satisfies the closure property of frequent patterns, i.e. each subset of a frequent pattern is itself frequent. This is not the case of the method proposed in [15]. Besides, the symbolic data form is more complex than classic or mono-valued data.

In order to implement the proposed concept of *frequent itemset*, while maintaining a good performance, we have adapted and extended the classic method. One of the mining frequent itemsets methods is the algorithm FP-growth; it consists first in building a compressed database called the Frequent Pattern Tree (FP-tree) [10] that retains the complete frequent items association information. This structure allows us to verify separately the support of each frequent item to generate the frequent itemsets. Since the performance of this algorithm has been proven, the issue of performance is less crucial. The remainder of the paper will introduce our extended method of FP-growth adapted to our presented problems. The paper is organized as follows. Section 2 introduces the method of mining frequent patterns from symbolic data. Section 3 presents our experimental study which demonstrates the interesting results of this new attempt. Section 4 summarizes our study and points out its contribution.

2 Mining association rules from symbolic data

The paper's method extends mining association rules from simple data to symbolic data. This method is restricted only to symbolic objects that are described by categorical multi-valued variables. This section details this method and describes its different phases. These include the pre-processing phase, the underlying data structure building phase and the algorithm of mining frequent itemsets phase.

2.1 Conversion from symbolic data to transactional data

Mining frequent patterns in transaction databases has been widely studied in data mining research, and many good methods have been approved, such as *Apriori* [1], *FP-growth* [10], and *TreeProjection* [2]. If the form of symbolic data could be converted to transactional data, we can benefit from the advantages of these traditional methods.

To attain this objective, three *issues* need resolving:

(1) *Performance*: the data of every individual are separated by the columns, looking for frequent itemsets one column after another, and then for all their possible combinations. This exponentional calculation will inevitably decrease the performance.

#i	A	B	C	D
1	1 (50%), 2 (50%)	1, 2	1, 2, 3	1
2	1(20%), 2(30%), 3 (50%)	1, 2, 3	1	1, 2
3	1(66%), 4 (34%)	2	2	1, 3
4	1(30%), 3 (70%)	2, 3	1, 3	3
5	1(20%), 2(50%), 4(30%)	2	1	1, 2
6	2(50%), 3(50%)	2	3	1, 5
7	2(30%), 3(70%)	1, 3	1	2, 3
8	1(55%), 3(45%)	2	1	2,4
9	2(60%), 4(40%)	3	1	1, 2
10	1 (60%), 3 (40%)	1, 3	1, 2	1, 2, 3

Table 4: ***A symbolic data table***

#i	Items
T1	A1, A2, B1, B2, C1, C2, C3, D1
T2	A1, A2, A3, B1, B2, B3, C1, D1, D2
T3	A1, A4, B2, C2, D1, D3
T4	A1, A3, B2, B3, C1, C3, D3
T5	A1, A2, A4, B2, C1, D1, D2
T6	A2, A3, B2, C3, D1, D5
T7	A2, A3, B1, B3, C1, D2, D3
T8	A1, A3, B2, C1, D2, D4
T9	A2, A4, B3, C1, D1, D2
T10	A1, A3, B1, B3, C1, C2, D1, D2, D3

Table 5: ***A table of transformed symbolic data (a set of transactions)***

Figure 3. Conversion of symbolic data to transactional type data.

Solution: a strategy is used to deal with this problem very easily. It consists in assigning a different code to the values of every attribute and removing their column bars, so that a value of each tuple appears as an item of a transaction and the classic methods become applicable.

(2) *Data type*: transaction data is only of Boolean data type, but symbolic data has several types: multi-valued, multi-valued with weights (as distributions), and interval. How to mine frequent itemsets from interval type data?

Solution: interval data are aggregated from quantitative data at the beginning of symbolic data generation. They can be converted to qualitative data by discretizing them. This process could be based on predefined concept hierarchies where numeric values are replaced by ranges.

(3) *The weights of values*: if an attribute is of modal type, i.e. multi-valued variable with weights, do we need to consider their weights in mining?

Solution: according to association mining definition, it is the occurrence frequency which is counted for every item support as argued in [1]. Accordingly, we do not need to care about the weight of values.

With the solutions, we can easily derive transactional form symbolic data. Example 2 shows this conversion.

Example 2 *In Fig. 3, Table 4 displays a symbolic data table with 10 individuals and 4 attributes. Furthermore the data type of attribute A is multi-valued with weights. Table 5 includes the corresponding transactional data type. Each attribute value in Table 4 becomes a transaction item in Table 5, having the column code as a prefix.*

2.2 Construction of eXtended Frequent Pattern (XFP-tree)

The phase of conversion reduces the complexity of data types. Now the only difficulty for us is how to verify if an itemset is frequent. As previously indicated, FP-growth has been selected as the base of our method due to the special data structure FP-tree. Hence, we first build an extension of FP-tree called *XFP-tree* (eXtended Frequent Pattern Tree), established by a special relative order. This relative order can afterwards help us to better explore this structure. Further advantages of this special order are outlined in section 2.3.

Let T={T_1, T_2,..., T_n} be a *symbolic data table* where $T_i(i \in [1,\ldots,n])$ is an individual, Attr={A, B, C,..., N} be its set of attributes and I={ A_1, A_2, ..., A_{m1}, B_1, B_2, ..., B_{m2}, ..., N_1, N_2, ..., N_{mn} } be the *set of all the values (or items)* of all attributes. To simplify the assignment of MIS for each item, we give a *MIS* to each attribute, i.e. MIS(A) = V_a, MIS(B) = V_b,..., MIS(N) = V_c, so each value of an attribute must have the same MIS. The *support* (or occurrence frequency) of a pattern (a set of items) P is the number of individuals containing P in T. P is a *frequent itemset* if it can satisfy the condition of *definition 1.*

Notice that in FP-tree, the frequent items are sorted in descending order of support. In XFP-tree, only the order is different, it is tailored to the optimization of the mining phase.

Definition 2 *(Relative order) A relative order in XFP-tree is the arrangement of its nodes.*

It is built by placing, in ascending order, the frequent single items' MIS. But if two frequent single items have the same MIS, they are ordered by their support descending order.

Before designing the XFP-tree, let's first examine an example.

Example 3 *builds an XFP-tree with a set of MIS of each attribute MIS(A) = 40%, MIS(B) = 20%, MIS(C) = 45%, MIS(D) = 30%, using a converted transactional table like in example 2 (Table 5 in Fig. 3). In this example, for the item set L = [A1:8, C1:6, B2:5, A3:5, D1:5], in which every element satisfies its MIS, the relative order will be computed as: B2 ≥ D1 ≥A1 ≥A3 ≥C1.*

After determining this relative order, XFP-tree construction can start following exactly the same procedure as FP-tree construction in [10].

Continuing example 3, to avoid repeatedly scanning the converted symbolic database, like the construction of FP-tree, the values of every individual are listed according to the relative order shown in Table 6.

We may create the root of the tree, labeled with "null". *Scan the converted symbolic database for a second time.* The scan of the values of the first individual T1 leads to the construction of the first branch of the tree: <(B2:1), (A1:1), (C1:1)>. Notice that the frequent items in the transaction are ordered according to the relative order

obtained by *definition 2*. For the second individual T2, since its (ordered) frequent item list <B2, D1> shares a common node <B2> with the existing path <B2, A1, C1>, the count of the node B2 is *incremented by* 1, and one new node <D1> is created and linked as the child of <B2>. The scan of the third individual T3 leads to the construction of the second branch of the tree, <(A1:1), (C1:1)>, and so on.

To facilitate tree traversal, an item header table is built in which each item points to its occurrence in the tree via a head of node-link. Nodes with the same item-name are linked in sequence via such node-link. After scanning the values of all individuals, the tree with the associated node-links is shown in Fig. 4.

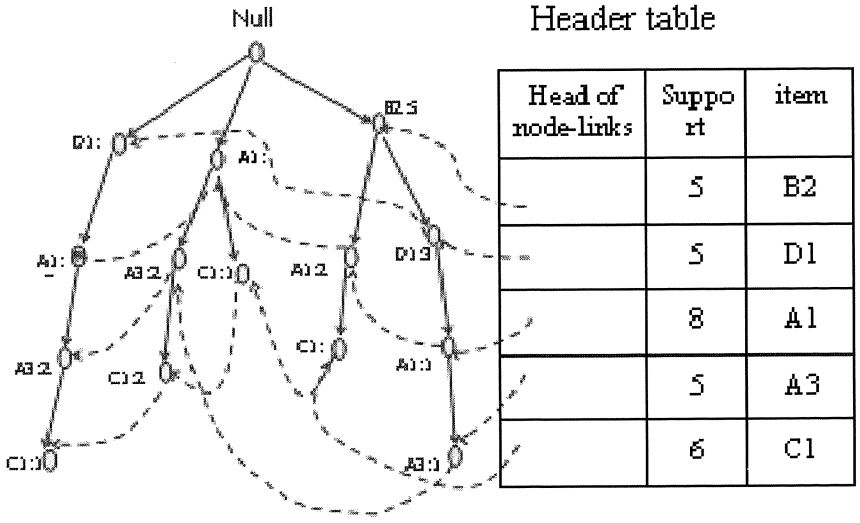


Figure 4. The XFP-tree of example 2.

TID	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
Item list	A1, B2, C1, D2	A2, B1 B2, D1	A1, C1, C2	A1, B2, C1	A1, B2, A3, C1 D1, D2	B2, D1,	A1, A2, A3, C1	A1, A2, A3, C1 D1	A1, A3, C1, D1,	A1, B2, A3, C2
(ordered) frequent items	B2, A1,	B2, D1,	A1, C1,	B2, A1,	B2, D1, A1, A3,	B2, D1	A1, A3,	D1, A1 A3, C1	D1, A1, A3	A1, A3,

Table 6. A transformed symbolic data table as a running example according to relative frequent items order.

2.3 Mining Frequent Patterns using XFP-tree

XFP-tree construction with a relative order of frequent items makes this structure less compact than the original FP-tree. However, it is highly efficient since one avoids the *complicate combinatorial problem of candidate generation* when relative order is employed. In this section, we will study the ways to explore the information stored in this XFP-tree structure when the *min-supports* are not uniform, and extend the efficient mining method FP-growth for mining our complete set of frequent patterns.

Example 4 *This example examines the mining process based on the constructed XFP-tree shown in Fig. 4. By using the header table, all the patterns where a node a_i participates are collected, by starting from head a_i and following a_i node-links. Thus, starting from the bottom of the header table we can easily perform the mining process.*

For node C1, it derives a frequent pattern (C1:6) and four paths in the XFP-tree: <B2:5, A1:2, C1:2>, <A1:3, C1:1>, <A1:3, A3:2, C1:2> and <D1:2, A1:2, A3:2, C1:1>. The first path indicates that string "<B2:5, A1:2, C1:2>" appears twice in the database. Notice, however, that even if <B2 > itself appears five times, it only appears twice together with <A1, C1>. Thus to study, only C1 prefix path <B2:2, A1:2, C1:2> counts. Similarly, the second path indicates string "(A1, C1)" appears once in the set of Symbolic database, the third path indicates string "(A1, A3, C1)" appears twice and the fourth path indicates string "(D1, A1, A3, C1)" appears also once. These four prefix paths of C1, "{(B2:2, A1:2), (A1:1), (A1:1, A3:2), (D1:1, A1:1, A3:1)}", form C1 sub-pattern base, which is called C1 *extended conditional pattern base* (i.e. the sub-pattern base under the condition of C1 existence). Construction of an FP-tree on this conditional pattern base (called C1 *extended conditional XFP-tree*) leads to a recursive mining of the XFP-tree base <{(B2:2, A1:2), (A1:1), (A1:2, A3:2), (D1:1, A1:1, A3:1)}|C1>.

This conditional XFP-tree construction can be designed based on the following Lemma:

Lemma 1 *(eXtended fragment growth) Let α be an itemset in an XFP-tree, B be α conditional pattern base, and β be an item in B. If α is frequent and the support of β satisfies β MIS, the itemset < α, β > is also frequent.*

Proof. According to the definition of conditional pattern base and XFP-tree, each subset in B occurs under the condition of the occurrence of α in the original converted symbolic database. If an item β appears in B n times, it appears with α in the Symbolic database n times as well. As the definition of *XFP-tree*, β has a *MIS* smaller than that of α. If β support satisfies his *MIS*, $\alpha \cup \beta$ also satisfies this *MIS*. As α has already been frequent, the itemset < α, β > is therefore also frequent.

From this Lemma, we can easily derive an important corollary.

Corollary 1 *(eXtended pattern growth) Let α be an item in Symbolic database, B be α conditional pattern base, and β be an item in B. Then $\alpha \cup \beta$ is frequent in Symbolic database if and only if β is frequent in B.*

Proof. This lemma is the case when α is a frequent item but not an itemset.

We verify nodes frequency from the nodes children to the nodes father. Since we have established the XFP-tree structure by the ascending order of the node *MIS*, a node father appears frequently together with its nodes children if it is frequent in the extended conditional pattern base of its frequent children.

Fig. 5 shows mining C1 conditional pattern base "mine<{(B2:2, A1:2), (A1:1), (A1:2, A3:2), (D1:1, A1:1, A3:1)}|C1>" involving the first mining 2-itemsets. The two items of the first 2-itemset <A3, C1> appear together three times, C1 appears six times, satisfying his threshold, but this itemset is not frequent because A3 *MIS* is 4. The second 2-itemset <A1, C1> appears six times, so it is frequent. Like the reason of the first 2-itemset, <D1, C1> is not either frequent. The two items of the fourth 2-itemset <B2, C1> appear together two times, satisfying the B2's *MIS*. Since C1 has satisfied its *MIS*,. this 2-itemset is frequent. Notice C1 has a threshold bigger than that of B2, according to Corollary 1, <B2, C1> is frequent because B2 has satisfied its *MIS* and C1's support been previously verified. Notice also a set of items is not frequent if one of its subsets is not frequent. So we only need to verify the support of 3-itemset <B2, A1, C1> as <A3, C1> and <D1, C1> are not frequent. Since <B2, A1, C1> appears twice, satisfying B1's *MIS*, according to Lemma 1, this 3-itemset is frequent. Since <A1, C1> and <B2, C1> are included in <B2, A1, C1>, C1's conditional XFP-tree is <B2, A1, C1>.

The mining conditional XFP-tree <B2, A1 > |C1 is exactly the same as the conditional FP-tree mining in [10], so we do not introduce it here. The result of this mining is {{B2, C1}, {A1, C1} {B2, A1, C1}}.

Similarly to the generation of itemsets of C1, we obtain a table of mining all the frequent patterns of Example 3 in Table 7.

Based on Corollary 1 and Lemma 1, mining can be easily performed like FP-growth in [10]. But this advantage is based on the relative order. Without this relative order at the beginning of XFP-tree construction, the tree structure would be inevitably more difficult. Let's see an example.

Example 5 *We construct a FP-tree according to the method of construction in [10] where the frequent items are sorted in their descending order support as shown in Fig. 6 and the information pertaining to the supports and MISs of each node is located in the right header table.*

In the conditional pattern base {<A:2, B:2, C:2>, <A:1, C:1>, <B:1>}, we analyze first the support of all the 2-itemsets. <C, D> appears three times, satisfying D's *MIS*. As C has a *MIS* bigger than that of D, we must also verify if C's support is greater than its *MIS*. Finally we find that <C, D>, <B, D>, <A, D> are frequent 2-itemsets. The next step involves verifying the support of the 3-itemsets. For 3-itemset < B, C,

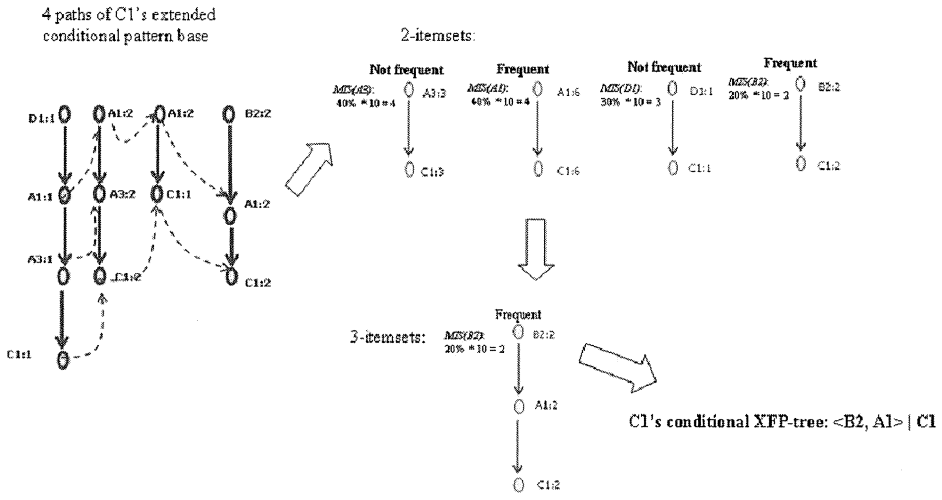
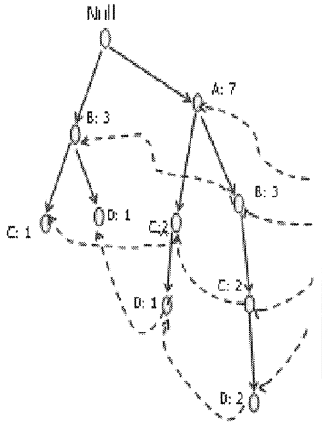


Figure 5. A conditional XFP-tree built from XFP-base for a node C1.



Node-link	Item	Support	MIS
	A	7	50%
	B	6	30%
	C	5	40%
	D	4	20%

Figure 6. The FO-tree of example 4.

D >, <C, D> appears three times, <B, C, D> twice, satisfying D's *MIS*. However, the support of <B, C> must be found in another conditional pattern base. Due to the lenghty verification procedure, this renders the process slow and complicated. Finally <B, C, D> is not frequent, as the <B, C> support is 3 that does not satisfy C's *MIS*. The three items of the second 3-itmeset <A, C, D> appear together 3 times, satisfying D's *MIS*, and the support of <A, C> is four, equally satisfying the *MIS* of C, so <A, C, D> is frequent. And so on.

From example 5 we can easily see that the process of mining conditional pattern bases is more complicated when the frequent items are not sorted by the relative order. Since every frequent item no longer has the same *MIS*, if a node child has a *MIS* smaller than its ancestor, we must verify not only its support with every ancestor, but also the support of every combination of all ancestors. Sorting these frequent items by their *relative order* first becomes very useful in mining the XFP-tree, as proved in Lemma 1.

3 Experiments study

The section evaluates the extended XFP-tree model and the XFP-growth algorithm. The performance and advantages of this model will be demonstrated by the comparison of XFP-growth with the classical frequent pattern mining algorithm and the proposed approach in [15] "*MsApriori*".

All the experiments are performed on a 2.4GHz Pentium PC machine with 256 DDR megabytes of main memory, running on Microsoft Windows/XP. All the programs are written in Microsoft/Visual C++6.0. We implement other algorithms to the best of our knowledge based on the published reports on the same machine and compare in the same running environment.

To compare these methods, we need to assign MIS values to each item in the datasets. Here, we use the method of MIS value distribution as in [15] for each item, with the following formula:

$$MIS(i) = \begin{cases} M(i) & M(i) > LS \\ LS & \text{Otherwise} \end{cases}$$

$$M(i) = BF(i)$$

F(i) is the support of the actual items. *LS* is the lowest minimum item support that is allowed by the user. B ($0 \leq B \leq 1$) is a controlling parameter of MIS which is related to its support. If B = 0, we only have one *min-support*, *LS*, which is the same as the conventional association rules mining. Here, for illustration purposes, we show the results from one dataset. The other datasets are similar and are thus omitted. This dataset contains about 30,000 transactions; the transaction size is around 10 to 15. Like in [15], we use also three very low LS values, being 0.1%, 0.2%, and 0.3% respectively. For the value B, we let B = 1/a where "a" varies from 1 to 15.

From section 1.4, we know that our approach and the *MsApriori* method together allow the user to specify *multiple min-supports* in order to find rare item rules without

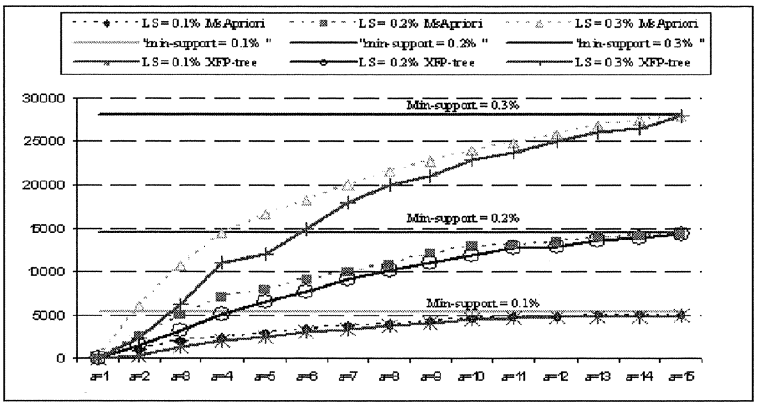


Figure 7. Number of large itemsets found.

producing a large number of meaningless rules with frequent items. However, our method provides another definition for frequent itemsets which directly takes into account each MIS in every candidate itemset. Fig. 7 shows the number of large itemsets found. The thick lines indicate the number of large itemsets found by the existing approach with a uniform *min-support* at 0.1%, 0.2%, and 0.3%.

From Fig. 7 we can see that the number of large itemsets is significantly reduced by the *MsApriori* method and is further reduced by our approach when "a" is not too large. When "a" becomes larger, the results of the three methods get closer as "a" reaches *LS*.

Since our method checks the MIS of all the subsets of each itemset to verify if the itemset is frequent, this procedure is more complicated than the MsApriori method. However, as our method is based on FP-growth where there is no need to generate candidate itemsets, the performance issue is less crucial. Fig. 8 shows the execution time comparison of the XFP-tree and *MsApriori* methods.

In Fig. 8, B is set to 0.1% and 0.2%, while LS varies from 0 to 3. We can see that the execution of our method is significantly faster than that of the *MsApriori* method when LS is small and B is fixed.

4 Conclusion

The paper contributes to the existing literature by extending functionally association rules mining. Specifically, our paper extends and contributes to the research field in three main ways: i) by dealing with complex data, such as symbolic data, where no equivalent method has been previously developed; ii) by more accurately fulfilling the user's requirements, by allowing the specification of different *MISs* depending on the support/nature of data; and finally iii), to gain a good performance by being based

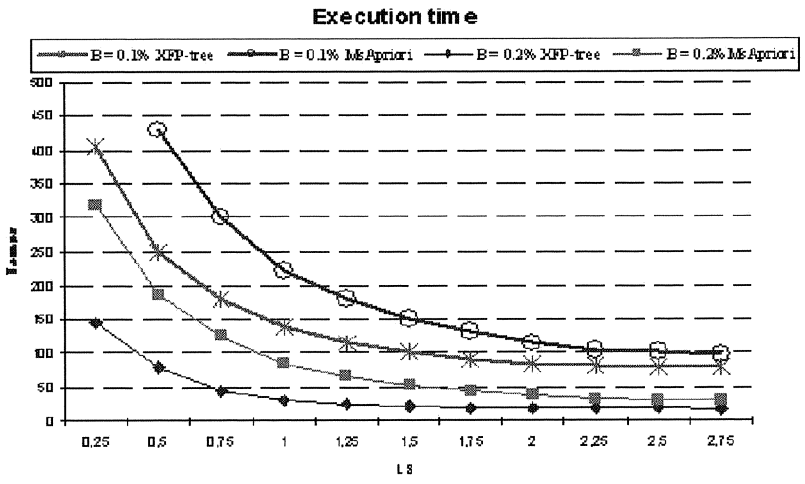


Figure 8. Scalability with LS

on the existing efficient algorithm FP-growth. Concerning symbolic data analysis, a conversion phase to transaction data has also been proposed. This method is not only tailored to symbolic tables but also applies to tabular data when some important items are unlikely to occur as frequently as others. The proposed method allows specifying different *min-supports* for certain dimensions (i.e. attributes) depending on their interest within the actual application. With different *min-supports* according to the users' requirements, we may obtain interesting information that we could not find by using the classical methods. The novel definition of *frequent itemsets* reduces the frequent pattern amounts without losing important information. By re-using the tree structure, FP-tree, we can separately verify every subset support of an itemset, resulting in the attainment of a good performance when the data is sufficiently dense.

The principle of our method is similar to that of the FP-growth algorithm. It first extends the FP-tree data structure to XFP-tree by changing the tree nodes order. Thus, the mining process could reuse the FP-growth algorithm with minimal changes.

This algorithm has been implemented, and the experimental results have been obtained, in the context of road traffic safety analysis, which is a spatial data mining application. This study demonstrates the interest of our method and a comparison of our method to that employed in [15].

References

[1] Agrawal, R., Imielinski, T. and Swami, A. (1993). Mining association rules between sets of items in massive databases. In *Proc. 1993 SIGMOD*, 207-216.

[2] Agarwal, R.C. Aggarwal, C. and Prasad, V.V.V. (2001). A tree projection algorithm for generation of frequent itemset. *J. of Parallel and Distributed Computing*, 61, 350-371.

[3] Agrawal, R. and Srikant, R. (1994). Fast algorithms for mining association rules. In *Proc. 1994 VLDB*, 487-499.

[4] Bayardo, R.J. (1998). Efficiently mining long patterns from databases. In *Proc. 1998 SIGMOD*, 85-93.

[5] Brin, S., Motwani, R. and Silverstein, C. (1997). Beyond market basket: Generalizing association rules to correlations. In *Proc. 1997 SIGMOD*, 265-276.

[6] Cheung, D.W., Han, J., Ng, V.T. and Wong, C.Y. (1996) Maintenance of discovered association rules in large databases: An incremental updating technique. In Proc. *1996 ICDE*, 106-114.

[7] Fukuda, T., Morimoto, Y., Morishita, S. and Tokuyarna, T. (1996) Data mining using two-dimensional optimized association rules: Scheme, algorithms, and visualization. In *Proc. 1996 SIGMOD*, 13-23.

[8] Han, E.-H., Karypis, G. and Kumar, V. (1997). Scalable parallel data mining for association rules. In *Proc. 1997 SIGMOD*, 277-288.

[9] Han, J. and Fu, Y. (1995). Discovery of multiple-level association rules from large databases. In *Proc. 1995 VLDB*, 420-431.

[10] Han, J., Pei, J. and Yin, Y. (2000). Mining Frequent Patterns without Candidate Generation. In *Proc, SIGMOD, 2000.*

[11] Klemettinen, M., Mannila, H., Ronkainen, P., Toivonen, H. and Verkamo, A.I. (1994). Finding interesting rules from large sets of discovered association rules. In *Proc. 1994 CIKM*, 401-408.

[12] Bock, H.H. and Diday, E. (2000). *Analysis of Symbolic Data - Exploratory methods for extracting statistical information from complex data.* Springer Verlag, Heidelberg.

[13] Sarawagi, S., Thomas, S. and Agrawal, R. (1998). Integrating association rule mining with relational data base systems; Alternatives and implications. In *Proc.1998 SIGMOD*, 343-354.

[14] Miller, R.J. and Yang, Y. (1997). Association rules over interval data. In *Proc. 1997 SIGMOD*, 13-24.

[15] Liu, B., Hsu, W. and Ma, Y. (1999). Mining association rules with multiple minimum supports. In *Proc ACM Intl. Conference KDD.*

[16] Park, J.S., Lakshmanan, M.S., Han, J. and Pang, A. (1995). Exploratory mining and pruning optimizations of constrained associations rules. In *Proc. 1995 CIKM*, 31-36.

[17] Park, J.S., Chen, M.S. and Yu, P.S. (1997). Using a hash-based method with transaction trimming for mining association rules. *IEEE TKDE*, 9, 813-825.

[18] Tsur, D., Ullman, J.D., Abiteboul, S., Clifton, C., Motwani, T. Nestorov, S. and Tosenthal, A. (1998). Query flocks: A generalization of association-rule mining. In *Proc. 1998 SIGMOD*, 1-12.

[19] Savasere, A., Omiecinski, E. and Navathe, S. (1995). An efficient algorithm for mining association rules in large databases. In *Proc. 1995 VLDB*, 432-443.

[20] Silverstein, C., Brin, S., Motwani, P. and Ullman, J. (1998). Scalable techniques for mining causal structures. In *Proc. 1998 VLDB*, 594-602.

[21] Toivonen, H. (1996). Sampling large databases for association rules. In *Proc. 1996 VLDB*, 134-145.

[22] Yang, J., Wang, W. and Yu, P.S. (2000). Mining asynchronous periodic patterns in time series data. In *Proc. 2000 KDD*, 275-279.

[23] Borges, J. and Levene, M. (1998). Mining association rules in hypertext databases. In *Proc, 1998, KDD*, 149-153.

[24] Dehaspe, L., Toivonen, H. and King, R.D. (1998). Finding frequent substructures in chemical compounds, In *Proc. 1998 KDD*, 30-36.

[25] Mannila, H. (1998). Database methods for data mining. *KDD-98*, tutorial.

[26] Lakshmanan, L.V.S., Leung. C.K.-S. and Ng, R.T. (2000). The Segment Support Map: Scalable Mining of Frequent Itemsets. In *Proc. 2000 SIGKDD*, 21-27.

[27] Dehaspe, L. and De Raedt, L. (1997). Mining association rules in multiple relations. In Proc *of the* 7^{th} *International workshop 1997 (ILP-97).* LNAI 1297. 125-132.

[28] Inokuchi, A., Washio, T. and Motoda, H. (2000). An apriori-based algorithm for mining frequent substructures from graph data. In *Proc PKDD 2000*13-23, LNAI, Springer.