International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems
Vol. 6, No. 3 (1998) 307–326

FUSINTER: A METHOD FOR DISCRETIZATION OF CONTINUOUS ATTRIBUTES

D. A. ZIGHED, S. RABASÉDA, R. RAKOTOMALALA
Equipe de Recherche en Ingénierie des Connaissances, Bât. L, Université Lumière Lyon 2
5 av. Pierre Mendès-France, 69676 Bron. (FRANCE)
Tél. (33) 04 78 77 23 76
Fax. (33) 04 78 77 23 75
E-mail: {zighed,rabaseda,rakotoma}@univ-lyon2.fr

Received October 1995
Revised April 1998

In induction graphs methods such as C4.5[1] or SIPINA[2], taking continuous attributes into account needs particular discretization procedures. In this paper, we propose on the one hand, an axiomatic leading to a set of criteria which can be used for continuous attributes discretization, and on the other hand, a method of discretization called FUSINTER. The results obtained by FUSINTER are compared to those obtained by techniques developed by Fayyad and Irani[3] and Kerber[4] and they have proved better for the majority of the examples studied.

Keywords: Discretization; Pattern Recognition; Induction Graph.

1. Introduction

In a machine learning problem, we generally wish to dispose of reliable methods, that do not modify the data structure, that do not require very high statistical hypotheses and that provide models easy to interpret. Among the techniques which correspond best to these characteristics, induction graphs take an important place. For example, there is the segmentation by binary tree[5], C4.5[1] that provides a decision tree, SIPINA[2] which provides a decision graph, ...

Initially, those methods have been conceived for categorical attributes, i.e for attributes that take their values in a discrete set of finite cardinal. In a sociological investigation for example, the "sexual" attribute takes its values in the set {male, female}. If, as it is often true, the variable is continuous, so it is discretized. This consists in cutting its field in a finite number of intervals, and each interval will be identified by a different code. The attribute being thus made categorical, it can be used in the process of induction graph process.

The choice of the discretization technique has important consequences on the induction model which will be built. In this paper we propose a new method called FUSINTER whose performances have proved better than others achieved

by Chi-Merge[4] or MDLPC[3] which are the most quoted in machine learning literature[6] [7] [8] [9]....

The presentation of this article is made in seven points. In section 2, we will try to give a precise formulation of the discretization problem of a continuous attribute. We will, of course, find ourselves in the context of the methods we have just evoked i.e those in which we deal with a pattern recognition problem by supervised learning. In section 3, we will evoke methods which are said unsupervised, that is, those which do not use the information related to classes for which we are researching a predictive model. These techniques choose a cut point, either in an arbitrary way or with a view to optimizing a criteria. We must precise that, in this case, only the information related to the attribute we have to discretize is taken into account. In section 4, we present the two supervised methods which seem the most performing ones. These are the Chi-Merge method[4], and MDLPC[3]. We establish a criticism showing the limits of these methods on concrete illustrations. Section 5 will be devoted to the presentation of the FUSINTER method. We are showing how this one properly handles the limits of MDLPC and Chi-Merge. In section 6, we propose a comparison between the three methods. To do that, we use artificial and real data sets. In section 7, we shall discuss the results of this research and its interests in the frame of the induction graph process.

2. Discretization

2.1. *Notations*

Let $X(.)$ be a statistical attribute taking its value on the straight line of the real numbers $\mathbb{R}$. For any example ω taken from a training sample noted Ω, $X(\omega)$ stands for the value taken by this example on the attribute $X(.)$, $(X(\omega) \in \mathbb{R})$. To fix our mind, let's suppose that $X(.)$ stands for the height of a person in cm. $X(\omega) = 182.3cm$ means that the person ω is $182.3cm$. Generally, in machine learning problem, we dispose of several numeric and categorical attributes. They are called, according to the various fields, exogenous variable or explicative attributes. In any problem of supervised learning, we also dispose of a particular statistical variable noted $Y(.)$. It stands for the belonging class of the examples. We try to define its values by a model. It is called endogenous variable, class or concept, according to the disciplines. Unlike exogenous attributes, $Y(.)$ is a variable supposed to be categorical. It takes its values in a finite and discrete set called "label set" and noted $E = \{y_1, ..., y_m\}$. If the example ω is labeled y_j, then, we will say that it belongs to the y_j class and we will write $Y(\omega)=y_j$. The value of $Y(.)$ is supposed to be known for all the examples of the training sample Ω. In a machine learning problem, we aim at building a model M which would enable us to calculate the value of $Y(.)$ thanks to the attributes ones :

$$Y(.) = M(X_1(.), \ldots, X_p(.))$$

In the particular case of the induction graphs, this model is expressed by a

decision graph; in rule induction from example, this model is a set of production rules.

2.2. *Formalization*

Let D_X be the definition field of $X(.)$. Discretizing the attribute $X(.)$ is to cut D_X with a set of threshold values d_j. We obtain k intervals I_j $(j = 1, \ldots, k; k \geq 2)$ which are numbered from $1, \ldots, k$.

$$\begin{aligned} I_1 &= [d_0, d_1[\\ &\vdots \\ I_j &= [d_{j-1}, d_j[\\ &\vdots \\ I_k &= [d_{k-1}, d_k[\end{aligned}$$

Once these threshold values are found, the continuous attribute $X(.)$ is replaced by a categorical attribute $\widetilde{X}(.)$ which takes its values in the set $\{1, \ldots, k\}$. Thus, $\forall \omega \in \Omega$,

if

$$d_{j-1} \leq X(\omega) < d_j$$

then

$$\widetilde{X}(\omega) = j$$

3. Unsupervised methods

Unsupervised methods do not bother about knowing whether the intervals resulting from a discretization are interesting in relation to the class. Only the information relative to the attribute $X(.)$ is taken into account regardless of any other attribute.

Among the most basic methods, we can quote the one which consists in determining k arbitrarily and in building k intervals of equal ranges, or the one which makes a subdivision of d into k intervals of equal size. The unsupervised discretization can be apprehended in a more complex way. Lafaye[10] regards it as a problem of sorting and separating intermingled probability laws where each mode characterizes a particular law. The existence of an optimum analysis was studied by Teicher[11], Yakowitz[12].

In this perspective, many discretization techniques have been proposed. Dorofeyuk[13] and Lafaye[10] list the main ones and show their limits which are generally due to very high statistical hypotheses and which are seldom checked on real data. Some of those proposed methods are recent[14].

4. Supervised methods

4.1. *General formalization*

The supervised methods search for discretization points of the attribute $X(.)$ taking the values of $Y(.)$ into account. Indeed, in the way that the final goal is to build a model which enables to predict the value of $Y(.)$, it seems natural to search for the discretization points which bring us closer to this situation, where each interval of the discretization will have to contain exclusively examples possessing the same class label $Y(.)$. Hence the new formulation of the objective:

- *we have to cut the field of definition d of $X(.)$, into the intervals I_j ($j = 1, \ldots, k; k \geq 2$).*

$$\begin{aligned} I_1 &= [d_0, d_1[\\ &\vdots \\ I_j &= [d_{j-1}, d_j[\\ &\vdots \\ I_k &= [d_{k-1}, d_k[\end{aligned}$$

such as

$$\forall I_j (j = 1, \ldots, k), \exists! y_l \in \{y_1, \ldots, y_m\} \ / \ P(y_l / I_j) \approx 1$$

Let us precise that no hypothesis on statistical distribution of $P(Y(.)/I_j)$ can be made. The probabilities $P(y_l/I_j)$ can be, for instance, estimated empirically by $f(y_l/I_j)$, the frequency of the class y_l in the interval I_j.

$$f(y_l/I_j) = \frac{Card\{\omega \in \Omega; X(\omega) \in I_j, Y(\omega) = y_l\}}{Card\{\omega \in \Omega; X(\omega) \in I_j\}}$$

Let us imagine that no examples superpose :

$$\forall (\omega, \omega') \in \Omega^2, X(\omega) \neq X(\omega')$$

A simple solution consists in cutting D_X into as many intervals as there are examples in the sample set. Since each example only belongs to one class and considering the hypothesis that no points superpose, we have thus :

$$\forall I_j (j = 1, \ldots, k), \exists! y_l \in \{y_1, \ldots, y_m\}; f(y_l/I_j) = 1$$

This discretization is not interesting of course since the probabilities have been estimated each time on one example. Refusing this procedure shows us that the solution of the discretization problem does not exist necessarily. Indeed, if the examples sorted according to their value of $X(.)$ take different and alternate values on $Y(.)$, so the only discretization which correspond to the objective defined previously consists in adopting the simple division (one example

for one interval) we have rejected. Any discretization must thus lead to intervals containing a sufficient number of examples. Before discretization methods presentation, we are going to introduce some notations which will make the writing easier.

Let us consider for $X(.)$ a division into k intervals and :

1. Let n_{ij} be the number of training examples which are in the interval I_j and which belong to the class y_i,

$$n_{ij} = Card\{\omega \in \Omega; X(\omega) \in I_j, Y(\omega) = y_i\}$$

2. Let $n_{.j}$ be the number of the examples which are in the interval I_j.

$$n_{.j} = \sum_{i=1}^{m} n_{ij}$$

3. Let $n_{i.}$ be the number of examples of the class y_i,

$$n_{i.} = \sum_{j=1}^{k} n_{ij}$$

4. n be the number of the sample.

$$n = \sum_{j=1}^{k} n_{.j}$$

$$n = \sum_{i=1}^{m} n_{i.}$$

To each discretization in k intervals we can associate a matrix T of m lines and k columns. The lines correspond to the classes and the columns to the intervals.

$$T = \begin{pmatrix} n_{11} & n_{12} & \dots & n_{1k} \\ n_{21} & n_{22} & \dots & n_{2k} \\ \vdots & \vdots & \vdots & \vdots \\ n_{m1} & n_{m2} & \dots & n_{mk} \end{pmatrix}$$

We note T_j all the column j of the matrix T;

$$T_j = \begin{pmatrix} n_{1j} \\ n_{2j} \\ \vdots \\ n_{mj} \end{pmatrix}$$

In some passages of this article, to be clearer, we adopt the following writing for T :

$$T = (T_1, \ldots, T_j, \ldots, T_k)$$

4.2. *The method based on the MDLPC Criterion*

It was proposed by Fayyad and Irani[3] and uses an information criterion called *Minimum Description Length Principal Cut.* This is a top down method: D_X is divided into two intervals which are in their turn divided each into two intervals and so on until a certain stopping rule. First, we describe the main steps of the algorithm. Second, we present the MDLPC criterion.

4.2.1. *The steps of the discretization algorithm based on the MDLPC*

1. : All examples are sorted according to the increasing values of $X(.)$ making thus runs of points identified by their class.
2. : Each run of points of a same class forms an interval.
3. : If several classes are superposed on a same value of $X(.)$, then the associated interval will be reduced to this unique value and unlike other intervals, this one will contain a mixing of classes. For example, on the figure 1, the fourth interval contains three examples among which two come under the class represented by "x" and one belonging to the one noted "o".

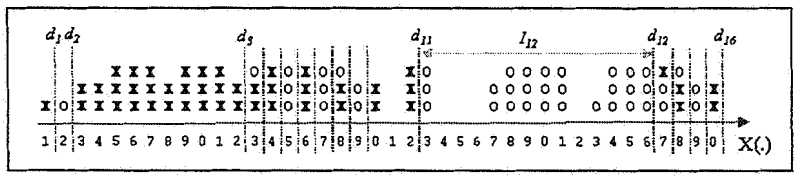

Figure 1: Definition of the runs I_j and potential discretization points d_j in a population divided up into two classes : "o" and "x".

4. : The discretization point is necessarily taken from the boundary point d_j of the intervals made up at the steps 2 and 3.
5. : Among the k points of discretization, we look for the one which leads to the "bi-partition" checking best the condition based on the $MDLPC$ criterion.
6. : The fifth steps is renewed on each of the two sub-populations.
7. : The process stops as soon as no improvement is possible. The result of the algorithm on the data of figure 1 is represented in picture 2

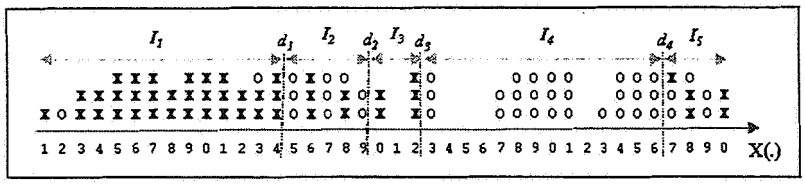

Figure 2: The MDLPC criterion leads to a five-interval discretization I_j, (j=1,...,5).

4.2.2. *The MDLPC criterion*

Let us go to the fourth stage of the algorithm with k intervals generated by the boundary points $d_j (j = 1, \ldots, k)$. We are looking for the discretization point d_t which leads to the best bi-partition on Ω. Let Ω_1 and Ω_2 be the elements of this bi-partition.

$\Omega_1 = \{\omega \in \Omega; X(\omega) \leq d_t\}$
$\Omega_2 = \{\omega \in \Omega; X(\omega) > d_t\}$
We must remind that,

- $n = Card(\Omega)$, stands for the sample number
- $n_{.j} = Card(\Omega_j), (j = 1, 2)$
- $n_{i.} = Card\{\omega \in \Omega; Y(\omega) = y_i\}$, number of the class $y_i, (i = 1, \ldots, m)$
- $n_{ij} = Card\{\omega \in \Omega_j; Y(\omega) = y_i\}$, number of the $y_i, (i = 1, \ldots, m)$ in the sub-sample $\Omega_j, (j = 1, 2)$.
- $m_j = Card(Y^{-1}(\Omega_j))$, with

$$\begin{aligned} Y^{-1} &: P[\Omega] \to E \\ &: \Omega_j \mapsto y_j = \arg\max_i Card\{\omega \in \Omega_j; Y(\omega) = y_i\} \end{aligned}$$

The MDLPC criterion is

$$\Psi(d) = Gain(d) - \frac{log_2(n-1)}{n} - \frac{\delta(d)}{n}$$

we choose the discretization point d^* that checks :

$$\begin{cases} d^* = \arg\max_d [\Psi(d)] \\ \Psi(d^*) > 0 \end{cases}$$

Notations are:

- $h(\Omega) = -\sum_{i=1}^{m} \frac{n_{i.}}{n} Log_2 \frac{n_{i.}}{n}$, the Shannon entropy;
- $h(\Omega_j) = -\sum_{i=1}^{m} \frac{n_{ij}}{n_{.j}} Log_2 \frac{n_{ij}}{n_{.j}}$, the conditional entropy;

- $Gain(d_i) = h(\Omega) - h(\Omega_j)$, the entropy gain criterion;
- $\delta(d) = log_2(3^m - 2) - mh(\Omega) + \sum_{j=1}^{2} m_j h(\Omega_j)$.

4.3. *The Chi – Merge method*

This method was proposed by Kerber[4] and lies on the use of a statistical criterion, the χ^2 test. This is a bottom-up algorithm and can be summed up as such :

1. : The examples are sorted according to increasing values or $X(.)$.
2. : Each value taken by the attribute $X(.)$ forms an interval.

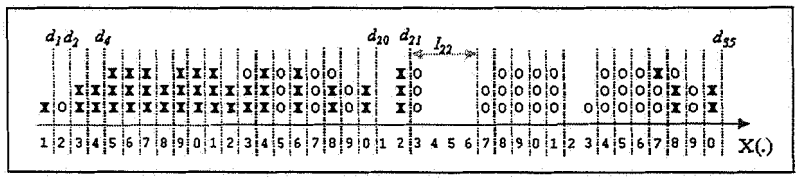

Figure 3: There are as many intervals as there are observed different values for $X(.)$.

3. : To each interval I_j is associated a distribution T_j where n_{ij} stands for the number of examples of the interval I_j belonging to the class y_i.
4. : We calculate the value of the χ^2 associated to the matrix formed by the juxtaposition of two columns T_j and T_{j+1} corresponding to two adjacent intervals:

$$\chi^2(T_q, T_{(q+1)}) = \sum_{i=1}^{m} \sum_{j=q}^{q+1} \frac{(n_{ij} - n_{.j} \sum_{k=q}^{q+1} n_{ik})^2}{n_{.j} \sum_{k=q}^{q+1} n_{ik}}$$

5. : We merge the pair of adjacent intervals I_q and I_{q+1} that gives the smallest value of χ^2 and that checks :

$$\chi^2(T_q, T_{(q+1)}) < \chi^2(\alpha, m-1)$$

where $\chi^2(\alpha, m-1)$ is read on the table of χ^2 at the threshold α (α is the type I error risk) and at $m-1$ degrees of freedom.

6. : We renew the previous step with $k-1$ intervals.
7. : The process stops as soon as no more merging is possible. The result of the discretization is shown on picture 4.

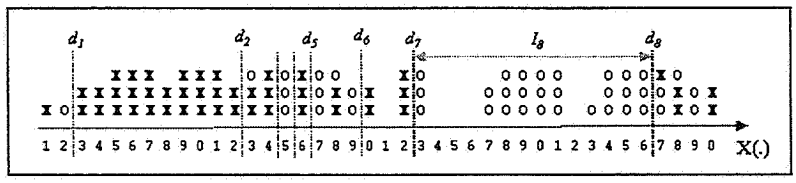

Figure 4: Result of the discretization by $Chi - Merge$ with $\alpha = 0.05$

5. The FUSINTER method

The FUSINTER method uses the same strategy than the Chi-Merge method. Its main characteristic is to be based on a measure sensitive to the sample size and which was used in the SIPINA induction graphs construction method. This measure was introduced in Zighed's book[2]. Contrary to Chi-Merge method that tries to merge adjacent intervals locally, FUSINTER method is a bottom-up algorithm to find the partition which optimizes the measure. It has also the advantage to avoid very thin partitioning due to its specific properties detailed hereafter. In the following, we propose a short description of the FUSINTER algorithm and then focus on the properties that a measure must verify.

5.1. *The FUSINTER Algorithm*

We are looking for the discretization that minimizes a criterion φ. The algorithm develops as follow :

1. : All the examples are ordered according to the increasing values of $X(.)$ thus forming runs of classes.
2. : Each run of examples of a same class will form an interval.
3. : If there is a superposition of several classes on a same value of $X(.)$, then the associated interval will be reduced to this unique value and unlike the other intervals, this one will contain a mixing of classes.

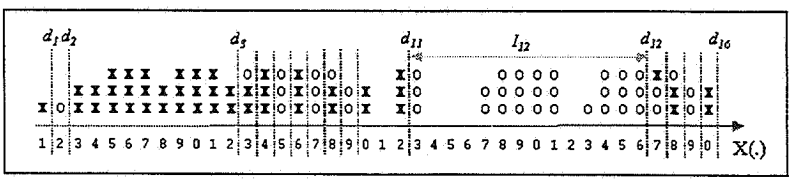

Figure 5: Constitution of a first list of possible discretization points d_j and associated intervals.

4. : The discretization points are necessarily boundary points of the intervals established at the steps 2 and 3.
5. : Let us suppose that this first discretization provides k intervals. We deduce a matrix T of m lines and k columns that allows to calculate $\varphi(T)$ $T = (T_1, \ldots, T_{(j-1)}, T_j, \ldots, T_k)$.

6. : We are looking for the two adjacent intervals whose merging would improve the value of the criterion, that is j such as :

$$\varphi(T) - \varphi(\ldots, \{T_j + T_{(j+1)}\}, \ldots) = Max_{i=1}^{k-1} \left(\varphi(T) - \varphi(\ldots, T_i + T_{(i+1)}, \ldots)\right)$$

7. : If

$$\varphi(T) - \varphi(T_1, \ldots, T_j + T_{(j+1)}, \ldots, T_k) > 0$$

then, the two intervals I_j and I_{j+1} are gathered.

8. : The process is done again from step 2 with $k-1$ intervals until no improvement is possible or k reaches the value 1. If the process stops with $k = 1$, it means that the discretization of $X(.)$ is of no interest for the determination of $Y(.)$. For the example presented before, the result of $FUSINTER$ is illustrated on picture 6.

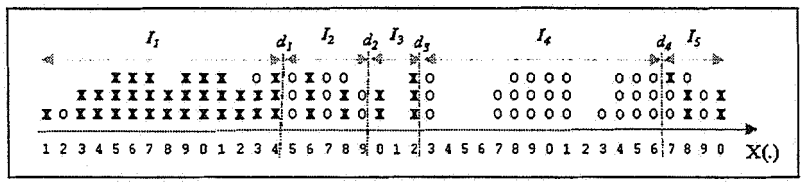

Figure 6: Result of the discretization by FUSINTER with $\alpha = 0.975$ and $\lambda = 1$

5.2. *Axiomatics for a new discretization criterion*

The success of the discretization procedure basically relies on the qualities of the measure ϕ used to quantify the relevance of the discretization. In this section, we provide a list of properties that a measure must verify to be used in the FUSINTER strategy.

Any discretization leads to a matrix T of m lines and k columns ($k \geq 1$) and ($m \geq 2$). If $k = 1$, it means that all the examples are gathered in a same interval and that consequently, there is no discretization. The criterion φ we are trying to build must be connected to the table T generated by the discretization and takes its values in $\mathbb{R}^+$, hence :

$$\varphi : \mathbb{R}^{mk} \mapsto \mathbb{R}^+$$

$$T \in \mathbb{R}^{mk} \mapsto \varphi(T) \in \mathbb{R}^+$$

φ must check the following axioms :

Axiom 1 (Minimality) *φ must be minimal if, in each interval there are only elements of a same class, that is :*

$$\forall j \in \{1, \ldots, k\}, \exists i \in \{1, \ldots, m\}; P(y_i / I_j) = 1.$$

As we are working on finite samples, it is achieved when :

$$\forall j \in \{1,\ldots,k\}, \exists i \in \{1,\ldots,m\}; n_{ij} \neq 0, n_{(t\neq i)j} = 0.$$

In this case, the matrix T has the following shape :

$$T = \begin{pmatrix} 0 & 0 & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ \alpha & 0 & \ldots & \gamma \\ \vdots & \vdots & \vdots & \vdots \\ 0 & \beta & \ldots & 0 \end{pmatrix}$$

In each column, there is only one non-null number.

Axiom 2 (Maximality) *φ must be maximal if in each interval there is the same number of elements in each of the m classes, that is :*

$$\forall j \in \{1,\ldots,k\}, P(y_i/I_j) = P(y_t/I_j), \forall (y_i, y_t) \in \{y_1,\ldots,y_m\}^2$$

As we are working on finite samples, the maximality is obtained when :

$$\forall j = 1,\ldots,k; \forall (i,t) \in \{1,\ldots,m\}, n_{ij} = n_{tj}$$

In this case, the table T has the following shape :

$$T = \begin{pmatrix} \alpha & \beta & \ldots & \gamma \\ \vdots & \vdots & \vdots & \vdots \\ \alpha & \beta & \ldots & \gamma \\ \vdots & \vdots & \vdots & \vdots \\ \alpha & \beta & \ldots & \gamma \end{pmatrix}$$

All the values in a same column are equal.

Axiom 3 (Sensitiveness to the sample size) *Let T be the matrix obtained by a discretization. If we increase the the sample size by multiplying the elements of the table T by a factor $\delta > 1$, the value of the φ criterion must decrease :*

$$\forall \delta > 1, \varphi(\delta T) < \varphi(T)$$

Let us note that the strict inequality accepted here is taken in the wide sense in the previous Zighed's works[2] and this difference is very important in the procedure of discretization whereas it is less so in the construction of the decision graph.

Axiom 4 (Symmetry) *Any permutation σ of the columns of the table T has no effect on the value of the criterion :*

$$\varphi(T_1,\ldots,T_j,\ldots,T_k) = \varphi(T_1\sigma,\ldots,T_j\sigma,\ldots,T_k\sigma)$$

Axiom 5 (Merging) *If* $\exists j, (j = 1, \ldots, k-1)$ *and* $\delta > 0$ *such as*

$$T_j = \delta T_{(j+1)}$$

then

$$\varphi(T_1, \ldots, \underbrace{T_j + T_{(j+1)}}, \ldots, T_k) < \varphi(T_1, \ldots, \underbrace{T_j}, \underbrace{T_{(j+1)}}, \ldots, T_k)$$

If, in a discretization, two consecutive intervals (D_j and $Dj+1$) have the same distribution classes, that is T_j and T_{j+1} are proportional, so, if they are merged together to make one, the value of the criterion should decrease.

In the SIPINA method [2], this axiom generally deals with any pair of columns of table T. That is what allows us to build graphs. The restriction we are introducing here and that we are limiting to the only case of adjacent intervals is not compatible with the SIPINA method.

Axiom 6 (Independence) *If we merge two intervals, the variation of the criterion must depend only on the gathered intervals.*

$$\varphi(T_1, \ldots, \underbrace{T_j + T_{(j+1)}}, \ldots, T_k) - \varphi(T_1, \ldots, \underbrace{T_j}, \underbrace{T_{(j+1)}}, \ldots, T_k) = \phi(T_j, T_{(j+1)}).$$

5.3. *Measures*

There is a family of measures that check the six previous axioms. They are derived from uncertainty measure [15]. In what follows, we present two of them, without giving the demonstrations which can be easily established from those given in previous Zighed's work[2]:

5.3.1. *Criterion based on Shannon's entropy*

$$\varphi_1(T) = \sum_{j=1}^{k} \alpha \frac{n_{.j}}{n} \left(-\sum_{i=1}^{m} \frac{n_{ij} + \lambda}{n_{.j} + m\lambda} log_2 \frac{n_{ij} + \lambda}{n_{.j} + m\lambda} \right) + (1-\alpha)\frac{m\lambda}{n.j}$$

We note that if we define $\lambda = 0$, we find the typical expression of Shannon's entropy.

5.3.2. *Criterion based on quadratic entropy*

$$\begin{aligned} \varphi_2(T) &= \sum_{j=1}^{k} \alpha \frac{n_{.j}}{n} \left(\sum_{i=1}^{m} \frac{n_{ij} + \lambda}{n_{.j} + m\lambda} \left(1 - \frac{n_{ij} + \lambda}{n_{.j} + m\lambda} \right) \right) + (1-\alpha)\frac{m\lambda}{n.j} \\ &= \sum_{j=1}^{k} \alpha H_j(h, \lambda) + (1-\alpha)\frac{m\lambda}{n_{.j}} \end{aligned}$$

5.4. *Settling λ and α parameters*

Parameters α and λ control the performances of the discretization procedure. Their values can be defined by experiments using a cross validation procedure and detect the best recognition rate ; but it is also possible to theoretically find some extreme values for those parameters in order to force the behavior of the discretization method in particular situations. We provide here some specific problems and some desired behaviour of the method, and give the specific values of α and λ for the measure ϕ_2. Of course, this approach is easily extensible to other measures ϕ.

On a given data sample, more intervals in a discretization implies less examples in each intervals. This is precisely the effect of the term $(1-\alpha)\frac{m\lambda}{n_{.j}}$ which penalizes over-splitting. The measure φ is a compromise between the purity measure $H_j(h,\lambda)$ and the splitting size measure $\frac{m\lambda}{n_{.j}}$. Following [16], an admissibility threshold*t is chosen and λ is optimized by maximizing the uncertainty $H(h,\lambda)$ computed on the following distributions:

$$\begin{array}{|c|}\hline T_t \\ \hline t \\ 0 \\ \vdots \\ 0 \end{array} \quad \text{and} \quad \begin{array}{|c|}\hline T_{(t+1)} \\ \hline t+1 \\ 0 \\ \vdots \\ 0 \end{array}$$

We thus have: $\delta = h\left(\frac{t+\lambda}{t+m\lambda}, \frac{\lambda}{t+m\lambda}, \ldots\right) - h\left(\frac{t+1+\lambda}{t+1+m\lambda}, \frac{\lambda}{t+1+m\lambda}, \ldots\right)$. Since h is the quadratic entropy, δ is given by

$$\begin{aligned}\delta \;=\; & \frac{t+\lambda}{t+m\lambda}\left(1-\frac{t+\lambda}{t+m\lambda}\right) \\ & +(m-1)\frac{\lambda}{t+m\lambda}\left(1-\frac{\lambda}{t+m\lambda}\right) \\ & -\frac{t+1+\lambda}{t+1+m\lambda}\left(1-\frac{t+1+\lambda}{t+1+m\lambda}\right) \\ & -(m-1)\frac{\lambda}{t+1+m\lambda}\left(1-\frac{\lambda}{t+1+m\lambda}\right)\end{aligned}$$

When $m=2$ and $t=2$ for instance,

$$\delta = \lambda\left(\frac{1}{2}\frac{5\lambda+6}{(1+\lambda)^2(3+2\lambda)^2}\right)$$

which is maximized for $\lambda = 0.61$.

In order to minimize the number of intervals having a too small size, we can force the merging of two intervals in the following case:

*t corresponds to the minimum size under which the size penalty increases, it depends on data set size, class distribution... and can be automatically computed[17]

$$\begin{pmatrix} \boxed{T_{t-1}} \\ t-1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} \boxed{T_1^*} \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

Thus, $\varphi(\{\ldots, T_{t-1}+T_1^*, \ldots\}) - \varphi(\{\ldots, T_{t-1}, T_1^*, \ldots\})$ must be strictly positive. This gives a value for α using the previously computed value for λ. Since the expression of φ is linear in α, this condition can be written as: $f(n,m,\lambda)\,\alpha - g(n,m,\lambda) > 0$. With $f(n,m,\lambda) > 0$, α is strictly greater than $\frac{g(n,m,\lambda)}{f(n,m,\lambda)}$.

For instance with $t = 2$, $m = 2$ and $\lambda = 0.61$, α can be set to 0.95.

Anyway, experiments showed that the discretization method is not very sensitive to the values of α and λ. So, from now on, we use the standard values $\lambda = 1$ and $\alpha = 0.975$ for the measure ϕ_2. Theses values are a good compromise between purity search and intervals sizes.

6. Comparison and discussion

6.1. *Illustrations*

In that follows, we are going to present the results of the three methods on illustrations of artificial data and real data.

1. : Let us consider an attribute $X(.)$. The training examples are ordered according to the increasing value of $X(.)$. We note by "x" or "o" the belonging class of the example. Picture 7 shows the results of each of the three methods.

   Without appealing to a statistical criterion, it is obvious that the MDLPC method has the worst discretization (figure 7a). Indeed, the series of points (o) has not been detected. On the contrary, Chi-Merge (figure 7b) detects it but has undesirable effects : intervals that contain only one point. FUSINTER (figure 7c) provides the discretization that seems for us most natural.

   The undesirable effects which appear in the MDPLC and Chi-Merge are even more intensified in picture 8.

   There, we can note that, whereas *FUSINTER* refuses any discretization (picture 8b), the two others propose an identical one (picture 8a) where the intervals situated at the extremes contain only one example. This result is not interesting in a training process whose final purpose is generalization.

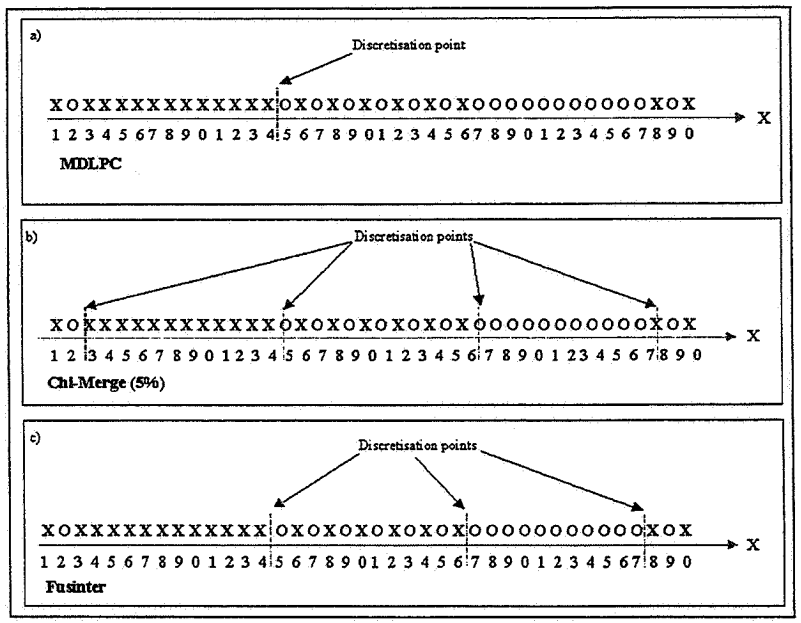

Figure 7: Discretization obtained by the three methods a) MDLPC proposes only one point of discretization. b) Chi-Merge extracts five points of discretization. c) FUSINTER spots three points of discretization.

2. : In the following data set, we have generated two normal distributions not superposing one another $N(0,1), N(2,1)$. The discretization results are given in picture 9. We note that, unlike the previous situations $MDLPC$ gives the same results as $FUSINTER$ (picture 9a). $Chi-Merge$ (picture 9c) gives a less interesting discretization.

3. : We have taken the same file as [4] concerning the discrimination of *Iris* varieties [18]. We present below the result of the discretization obtained by the three methods on the attribute $Petal-length$. For the other attributes, the results are identical for the three methods. Picture 10 shows the distribution of the three classes of *Iris Setosa*, $Versicolor$ and $Virginica$ according to the attribute $Petal-length$.

   The results of the discretization are provided in a contingency table. We can note that, again, the discretization provided by $FUSINTER$ seems more natural with a division into three classes with sufficient numbers for a good prediction of classes.

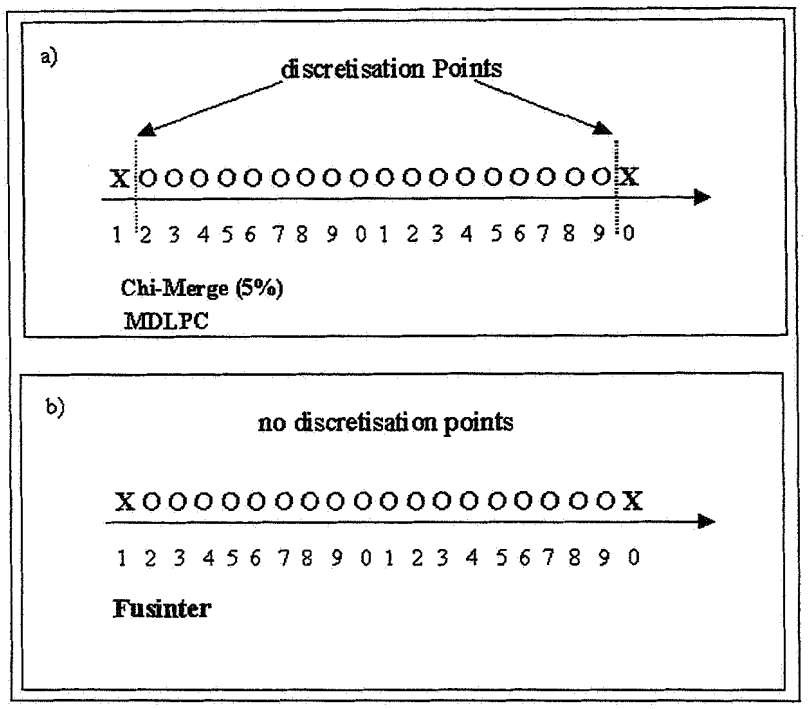

Figure 8: Discretization obtained by the three methods a) $Chi - Merge$ and $MDPLC$ provide three intervals among which two have a size of one. b) $FUSINTER$ considers this is better not to discretize in this case.

Method $MDLPC$				
Iris varieties	discretization intervals			
	]...,2.45]	]2.45,4.85]	]4.85,5.05]	]5.05,...]
Setosa	50	0	0	0
Versicolor	0	46	3	1
Virginica	0	3	6	41

Method $Chi - Merge$				
Iris varieties	discretization intervals			
	]...,3.00]	]3.00,4.80]	]4.80,5.20]	]5.20,...]
Setosa	50	0	0	0
Versicolor	0	44	6	0
Virginica	0	1	15	34

We can see that, with $FUSINTER$, the last two intervals of the attribute are gathered whereas $Chi - Merge$ does not achieve the merging.

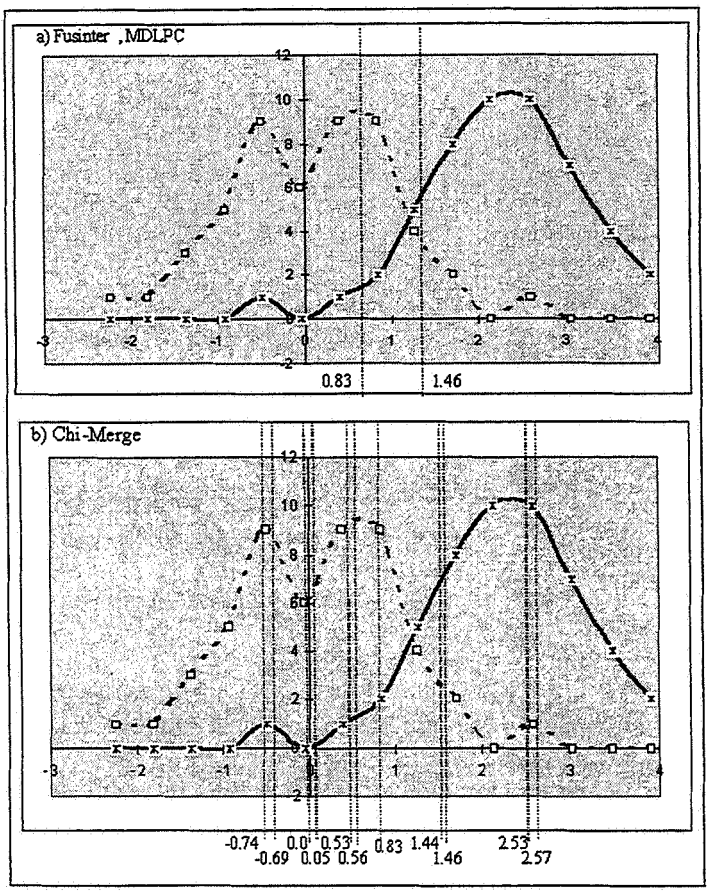

Figure 9: Discretization of a mixing of two classes with normal distributions not superposing each other. a) $FUSINTER$ and $MDLPC$ give an identical discretization in four intervals. b) $Chi-Merge$ detects twelve intervals and some of them are very close.

Method $FUSINTER$			
Iris varieties	discretization intervals		
	]...,3.00]	]3.00,4.90]	]4.90,...]
Setosa	50	0	0
Versicolor	0	46	4
Virginica	0	3	47

6.2. *Discussion*

To discretize a continuous attribute in the methods of induction by graphs, the empirical results we have obtained particularly favor $FUSINTER$. An essential reason for this is that the $MDLPC$ and $Chi-Merge$ methods do not

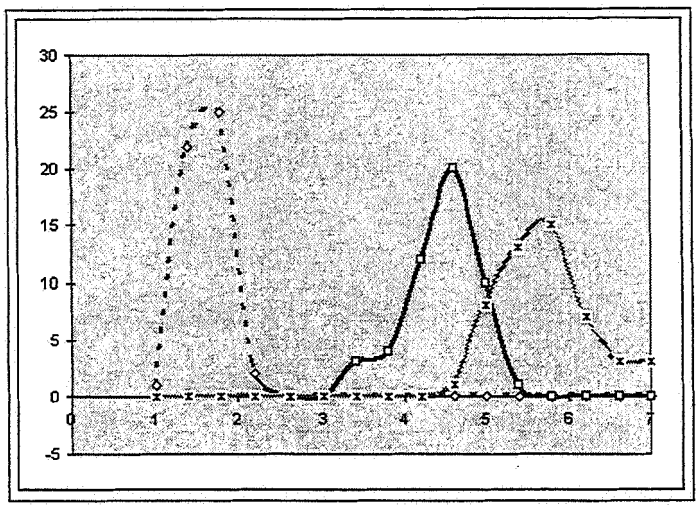

Figure 10: Distribution of the three classes of Iris : $Setosa, Versicolor$ and $Virginica$ according to the attribute $Petal - length$.

take into account finite size samples explicitly.

FUSINTER has an additional advantage since it can be used in process of induction by graphs to find the attribute that distinguishes best the classes, whatever the nature of the attribute : numeric or symbolic.

On a strictly algorithmic level, the three methods work on $O(nlogn)$. [4] considers that dynamic discretization, as [19] suggests as well, is not interesting. In fact, all the attributes have to be discretized *"a priori"* according to one of the methods presented. It should thus avoid to study again already analyzed processes since any discretization point will be one of the already stocked candidates. This argument is totally valid in the induction tree processes, but it no longer is in the SIPINA method since we admit the merging of sub-populations.

Before the discretization process, we can ask the question: is it interesting to discretize the attribute $X(.)$? The first solution is to try discretization and verify if we find a one or more discretization points. If there is no one i.e we have found no discretization points, we can conclude that the attribute $X(.)$ is not attractive for our prediction problem. The second one is the test of runs [20] that allows us to provide a statistical response to this question. It lies on the number of runs. If the classes are alternate as on picture 8, so the number of runs is important and the conclusion of the test is non discretization. On the contrary, if the number is low, then we conclude there are specific intervals belonging to classes. In this case, the intervals are deducted from the runs as in picture 1. So, in our experiments, using this test does not decrease the computation time beside classical attribute reject by discretization point research failure.

7. Conclusion

We do not think we have completely examined the problem of discretization in this study. Although it is a performing one, the FUSINTER method should be even more interesting if it could be integrated in a statistical way. Indeed, we have considered the discretization procedure as an optimization procedure. But it is also possible to consider it as an estimation problem so as to use inferential statistic. A cutting point then becomes a parameter of the population we try to generalize on a data set. Our goal could be then to reduce bias and variance of this statistical estimation.

The three discretization methods that have been presented are available on the software SIPINA-W that is accessible in free diffusion by "ftp://eric.univ-lyon2.fr" under the directory "/pub/sipina".

1. J. R. Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kaufmann, San Mateo, CA, 1993.
2. D.A. Zighed, J.P. Auray, and G. Duru. *Sipina : Methode et logiciel.* Lacassagne, 1992.
3. U.M. Fayyad and K. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *Proceedings of The 13th Int. Joint Conference on Artificial Intelligence*, pages 1022–1027. Morgan Kaufmann, 1993.
4. R. Kerber. Discretization of numeric attributes. In MIT Press, editor, *Proceedings of the Tenth National Conference on Artificial Intelligence*, pages 123–128, 1992.
5. J.P. Bouroche and M. Tenenhaus. Quelques mthodes de segmentation. *RAIRO, 42*, pages 29–42, 1970.
6. J. Dougherty, R. Kohavi, and M. Sahami. Supervised and unsupervised discretization of continuous attributes. In Morgan Kaufmann, editor, *Machine Learning : Proceedings of the Twelth International Conference (ICML-95)*, pages 194–202, 1995.
7. H. Liu and R. Setiono. Chi2 : Feature selection and discretization of numeric attributes. In *Proceedings of the 7th IEEE International Conference on Tools with Artificial Intelligence*, 1995.
8. J.R. Quinlan. Bagging, boosting and c4.5. In *Proceedings of the 13th American Association National Conference on Artificial Intelligence*, pages 725–730, Menlo Park, CA, 1996. AAAI Press.
9. T. Van de Merckt and J.R. Quinlan. Two-threshold splits of continuous attributes in decision trees. Technical report, The Basser Dept. of Computer Science - University of Sydney, Australia, 1996.
10. J. Y. Lafaye. Une mthode de discrtisation de variables continues. *Revue de statistique appliquée*, XXVII:39–53, 1979.
11. H. Teicher. Identifiability of finite mixtures. *Ann. Math. Stat.*, XXXIV:1265–1269, 1963.
12. S.S. Yakowitz and J.D. Spragins. On the identifiability of mixtures. *Ann. Math. Stat.*, XXIX(1):209–214, 1968.
13. A.A. Dorofeyuk. Automatic classification algorithms review. *Automation and Control*, 32(12):1928–1958, 1971.
14. K. Potzelberger and K. Felssenstein. On the fisher information of discretized data. *J. Statis. Comput. Simul.*, 46:125–144, 1993.

15. J. Aczel and Z. Daroczy. *On measures of information and their characterizations.* Academic Press, 1975.
16. D.A. Zighed and R. Rakotomalala. A method for non arborescent induction graphs. Technical report, Laboratory ERIC, University of Lyon 2, 1996.
17. R. Rakotomalala, D.A. Zighed, and S. Rabaseda. Validation of rules issued from induction graphs. In *Proceedings of the 6th Conference on Information Processing and Management of Uncertainty*, pages 1259–1264, 1996.
18. R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7:179–188, 1936.
19. J. Catlett. On changing continuous attributes into discrete ordered discrete attributes. In Y. Kodratoff, editor, *Proceedings of the European Working Session on Learning*, pages 164–178. Springer-Verlag, 1991.
20. A.M. Mood. The distribution theory of runs. *Ann. of Math. Stat.*, 11:367–392, 1940.