MODL: a Bayes Optimal Discretization Method for Continuous Attributes

MARC BOULLE
France Telecom R&D
2, Avenue Pierre Marzin
22300 Lannion – France
marc.boulle@francetelecom.com

Abstract. While real data often comes in mixed format, discrete and continuous, many supervised induction algorithms require discrete data. Efficient discretization of continuous attributes is an important problem that has effects on speed, accuracy and understandability of the induction models. In this paper, we propose a new discretization method MODL*, founded on a Bayesian approach. We introduce a space of discretization models and a prior distribution defined on this model space. This results in the definition of a Bayes optimal evaluation criterion of discretizations. We then propose a new super-linear optimization algorithm that manages to find near-optimal discretizations. Extensive comparative experiments both on real and synthetic data demonstrate the high inductive performances obtained by the new discretization method.

Key words: Data Mining, Machine Learning, Discretization, Bayesianism, Data Analysis

1 Introduction

Discretization of continuous attributes is a problem that has been studied extensively in the past (Catlett, 1991; Holte, 1993; Dougherty et al., 1995; Zighed and Rakotomalala, 2000; Liu et al., 2002). Many classification algorithms rely on discrete data and need to discretize continuous attributes, i.e. to slice their domain into a finite number of intervals. Decision tree algorithms first discretize the continuous attributes before they proceed with the attribute selection process. Rule-set learning algorithms exploit discretization methods to produce short and understandable rules. Bayesian network methods need discrete values to compute conditional probability tables.

In the discretization problem, a compromise must be found between information quality (homogeneous intervals in regard to the attribute to predict) and statistical quality (sufficient sample size in every interval to ensure generalization). The chi-square-based criteria (Kass, 1980; Bertier and Bouroche, 1981; Kerber, 1991) focus on the statistical point of view whereas the entropy-based criteria (Catlett, 1991;

* French patent N° 04 00179

Quinlan, 1993) focus on the information theoretical point of view. Other criteria such as Gini (Breiman et al., 1984) or Fusinter criterion (Zighed et al., 1998) try to find a trade off between information and statistical properties. The Minimum Description Length (MDL) criterion (Fayyad and Irani, 1992) is an original approach that attempts to minimize the total quantity of information both contained in the model and in the exceptions to the model. While most discretization methods are univariate and consider only a single attribute at a time, some multivariate discretization methods have also been proposed (Bay, 2001).

In this paper, we focus on univariate supervised discretization methods and propose a new method called MODL based on a Bayesian approach. First, we define a space of discretization models. The parameters of a specific discretization are the number of intervals, the bounds of the intervals and the class frequencies in each interval. Then, we define a prior distribution on this model space. Finally, we derive an evaluation criterion of discretizations, which is a direct application of the Bayesian approach for the discretization model space and its prior distribution. This criterion is minimal for the Bayes optimal discretization. Another important characteristic of the MODL discretization method is the search algorithm used to find optimal discretizations, i.e. discretizations which minimize the evaluation criterion. We describe an optimal search algorithm time complexity $O(n^3)$ where n is the sample size. We also propose a greedy search heuristic with super-linear time complexity and a new post-optimization algorithm that allows obtaining optimal discretizations in most cases. We demonstrate through numerous experiments that the theoretical potential of the MODL method leads to high quality discretizations.

The remainder of the paper is organized as follows. Section 2 presents the MODL method and its optimal evaluation criterion. Section 3 focuses on the MODL discretization algorithm. Section 4 proceeds with an extensive experimental evaluation both on real and synthetic data. Section 5 studies the relative contribution of the optimization criterion versus the search strategy.

2 The MODL evaluation criterion

The discretization methods have to solve a problem of model selection, where the data to fit is a string of class values and the model is a discretization model. The Bayesian approach and the MDL approach (Rissanen, 1978) are two techniques to solve this problem. In this section, we first recall the principles of these model selection techniques, and second present the MODL method, based on a Bayesian approach of the discretization problem.

2.1 Bayesian versus MDL model selection techniques

In the Bayesian approach, the best model is found by maximizing the probability $P(Model/Data)$ of the model given the data. Using Bayes rule and since the probability $P(Data)$ is constant while varying the model, this is equivalent to maximizing:

$$P(Model)P(Data/Model). \quad \textbf{(1)}$$

Once the prior distribution of the models is fixed, the Bayesian approach finds the optimal model of the data, provided that the calculation of the probabilities $P(Model)$ and $P(Data/Model)$ is feasible.

To introduce the MDL approach, we can reuse the Bayes rule, replacing the probabilities by their negative logarithms. These negative logarithms of probabilities can be interpreted as Shannon code lengths, so that the problem of model selection becomes a coding problem. In the MDL approach, the problem of model selection is to find the model that minimizes:

$$DescriptionLength(Model) + DescriptionLength(Data/Model). \quad (2)$$

The relationship between the Bayesian approach and the MDL approach has been examined by (Vitanyi and Li, 2000). The Kolmogorov complexity of an object is the length of the shortest program encoding an effective description of this object. It is asymptotically equal to the negative log of a probability distribution called the *universal distribution*. Using these notions, the MDL approach turns into *ideal MDL*: it selects the model that minimizes the sum of the Kolmogorov complexity of the model and of the data given the model. It is asymptotically equivalent to the Bayesian approach with a universal prior for the model. The theoretical foundations of MDL allow focusing on the coding problem: it is not necessary to exhibit the prior distribution of the models. Unfortunately, the Kolmogorov complexity is not computable and can only be approximated.

To summarize, the Bayesian approach allows selecting the optimal model relative to the data, once a prior distribution of the models is fixed. The MDL approach does not need to define an explicit prior to find the optimal model, but the optimal description length can only be approximated and the approach is valid asymptotically.

2.2 The MODL optimal evaluation criterion

The objective of the discretization process is to induce a list of intervals that split the numerical domain of a continuous explanatory attribute. The data sample consists of a set of instances described by pairs of values: the continuous explanatory value and the class value. If we sort the instances of the data sample according to the continuous values, we obtain a string S of class values. In Definition 1, we introduce a space of discretization models.

Definition 1: A *standard* discretization model is defined by the following properties:

1. the discretization model relies only on the order of the class values in the string S, without using the values of the explanatory attribute,
2. the discretization model splits the string S into a list of substrings (the intervals),
3. in each interval, the distribution of the class values is defined by the frequencies of the class values in this interval.

Such a discretization model is called a SDM model.

Notation:

n: number of instances
J: number of classes
I: number of intervals
n_i: number of instances in the interval i
n_{ij}: number of instances of class j in the interval i

A SDM model is defined by the parameter set $\{ I, \{n_i\}_{1 \leq i \leq I}, \{n_{ij}\}_{1 \leq i \leq I, 1 \leq j \leq J} \}$.

This definition is very general and most discretization methods rely on SDM models. They first sort the samples according to the attribute to discretize (property 1) and try to define a list of intervals by partitioning the string of class values (property 2). The evaluation criterion is always based on the frequencies of the class values (property 3).

Once a model space is defined, we need to fix a prior distribution on this model space in order to apply the Bayesian approach. The prior Definition 2 uses a uniform distribution at each stage of the parameters hierarchy of the SDM models. We also introduce a strong hypothesis of independence of the distributions of the class values. This hypothesis is often assumed (at least implicitly) by many discretization methods that try to merge similar intervals and separate intervals with significantly different distributions of class values. This is the case for example with the ChiMerge discretization method (Kerber, 1991), which merges two adjacent intervals if their distributions of class values are statistically similar (using the chi-square test of independence).

Definition 2: The following distribution prior on SDM models is called the *three-stage prior*:

1. the number of intervals I is uniformly distributed between 1 and n,
2. for a given number of intervals I, every division of the string to discretize into I intervals is equiprobable,
3. for a given interval, every distribution of class values in the interval is equiprobable,
4. the distributions of the class values in each interval are independent from each other.

Owing to the definition of the model space and its prior distribution, the Bayes formula is applicable to exactly calculate the prior probabilities of the models and the probability of the data given a model. Theorem 1 introduces the MODL evaluation criterion.

Theorem 1: A SDM model distributed according to the three-stage prior is Bayes optimal for a given set of instances to discretize if the value of the following criterion is minimal:

$$\log(n) + \log\binom{n+I-1}{I-1} + \sum_{i=1}^{I} \log\binom{n_i+J-1}{J-1} + \sum_{i=1}^{I} \log\left(n_i ! / n_{i,1} ! n_{i,2} ! \ldots n_{i,J} !\right). \qquad (3)$$

Proof:

The prior probability of a discretization model M can be defined by the prior

probability of the parameters of the model $\{ I, \{n_i\}_{1\le i\le I}, \{n_{ij}\}_{1\le i\le I,\ 1\le j\le J} \}$.

Let us introduce some notations:

- $p(I)$: prior probability of the number of intervals I,
- $p(\{n_i\})$: prior probability of the parameters $\{n_1,...,n_I\}$,
- $p(n_i)$: prior probability of the parameter n_i,
- $p(\{n_{ij}\})$: prior probability of the parameters $\{n_{11},...,n_{ij},...,n_{IJ}\}$,
- $p(\{n_{ij}\}_i)$: prior probability of the parameters $\{n_{i1},...,n_{iJ}\}$.

The objective is to find the discretization model M that maximizes the probability $p(M/S)$ for a given string S of class values. Using Bayes formula and since the probability $p(S)$ is constant under varying the model, this is equivalent to maximizing $p(M)\,p(S/M)$.

Let us first focus on the prior probability $p(M)$ of the model. We have

$$\begin{aligned} p(M) &= p(I, \{n_i\}, \{n_{ij}\}) \\ &= p(I)\, p(\{n_i\}/I)\, p(\{n_{ij}\}/I, \{n_i\}). \end{aligned}$$

The first hypothesis of the three-stage prior is that the number of intervals is uniformly distributed between 1 and n. Thus we get

$$p(I) = \frac{1}{n}.$$

The second hypothesis is that all the divisions of S into I intervals are equiprobable for a given I. Computing the probability of one set of intervals turns into the combinatorial evaluation of the number of possible interval sets. Dividing the string S into I intervals is equivalent to decomposing the natural number n as the sum of the frequencies n_i of the intervals. Using combinatorics, we can prove that the number of choices of such any $\{n_i\}_{1\le i\le I}$ is equal to $\binom{n+I-1}{I-1}$. Thus we obtain

$$p(\{n_i\}/I) = \frac{1}{\binom{n+I-1}{I-1}}.$$

The last term to evaluate can be rewritten as a product using the hypothesis of independence of the distributions of the class values between the intervals. We have

$$\begin{aligned} p(\{n_{ij}\}/I, \{n_i\}) &= p(\{n_{ij}\}_1, \{n_{ij}\}_2, ..., \{n_{ij}\}_I / I, \{n_i\}) \\ &= \prod_{i=1}^{I} p(\{n_{ij}\}_i / I, \{n_i\}) \\ &= \prod_{i=1}^{I} p(\{n_{ij}\}_i / n_i). \end{aligned}$$

For a given interval i with size n_i, all the distributions of the class values are equiprobable. Computing the probability of one distribution is a combinatorial problem, which solution is:

$$p\left(\left\{n_{ij}\right\}/n_i\right) = \frac{1}{\binom{n_i + J - 1}{J - 1}}.$$

Thus,

$$p\left(\left\{n_{ij}\right\}/I, \left\{n_i\right\}\right) = \prod_{i=1}^{I} \frac{1}{\binom{n_i + J - 1}{J - 1}}.$$

The prior probability of the model is then

$$p(M) = \frac{1}{n} \frac{1}{\binom{n + I - 1}{I - 1}} \prod_{i=1}^{I} \frac{1}{\binom{n_i + J - 1}{J - 1}}.$$

Let us now evaluate the probability of getting the string S for a given model M. We first split the string S into I sub-strings S_i of size n_i and use again the independence assumption between the intervals. We obtain

$$\begin{aligned} p(S/M) &= p\left(S/I, \left\{n_i\right\}, \left\{n_{ij}\right\}\right) \\ &= p\left(S_1, S_2, ..., S_I/I, \left\{n_i\right\}, \left\{n_{ij}\right\}\right) \\ &= \prod_{i=1}^{I} p\left(S_i/I, \left\{n_i\right\}, \left\{n_{ij}\right\}\right) \\ &= \prod_{i=1}^{I} \frac{1}{\left(n_i!/n_{i,1}!n_{i,2}!...n_{i,J}!\right)}, \end{aligned}$$

as evaluating the probability of a sub-string S_i under uniform prior turns out to be a multinomial problem.

Taking the negative log of the probabilities, the maximization problem turns into the minimization of the claimed criterion

$$\log(n) + \log\binom{n + I - 1}{I - 1} + \sum_{i=1}^{I} \log\binom{n_i + J - 1}{J - 1} + \sum_{i=1}^{I} \log\left(n_i!/n_{i,1}!n_{i,2}!...n_{i,J}!\right). \blacksquare$$

The first term of the criterion corresponds to the choice of the number of intervals and the second term to the choice of the bounds of the intervals. The third term represents the choice of the class distribution in each interval and the last term encodes the probability of the data given the model.

When the MODL criterion is used, we prove in Theorem 2 that optimal splits always fall on boundary points, where boundary points are located between two instances in the string S having different class values. The same property have already be presented for the MDLPC criterion in (Fayyad and Irani, 1992) and for a larger class of impurity measures in (Elomaa and Rousu, 1996).

Theorem 2: In a Bayes optimal SDM model distributed according to the three-stage prior, there is no split between two instances related to the same class.

Proof:

Assume that, contrary to the claim, such a split exists between two instances related to the same class (indexed as class 1 for convenience reasons). Let A1 and B1 be the intervals from each side of the split.

We construct two new intervals A0 and B2 by moving the last instance from A1 to B1. The cost variation $\Delta Cost1$ of the discretization is

$$\begin{aligned}\Delta Cost1 = &\log\left((n_{A0}+J-1)!/n_{A0}!(J-1)!\right)+\log\left(n_{A0}!/n_{A0,1}!n_{A0,2}!...n_{A0,J}!\right)\\ &+\log\left((n_{B2}+J-1)!/n_{B2}!(J-1)!\right)+\log\left(n_{B2}!/n_{B2,1}!n_{B2,2}!...n_{B2,J}!\right)\\ &-\log\left((n_{A1}+J-1)!/n_{A1}!(J-1)!\right)-\log\left(n_{A1}!/n_{A1,1}!n_{A1,2}!...n_{A1,J}!\right)\\ &-\log\left((n_{B1}+J-1)!/n_{B1}!(J-1)!\right)-\log\left(n_{B1}!/n_{B1,1}!n_{B1,2}!...n_{B1,J}!\right)\end{aligned}$$

The frequencies are the same for each class except for class 1, thus

$$\begin{aligned}\Delta Cost1 &= \log\left((n_{A1}+J-2)!/(n_{A1}+J-1)!\right)+\log\left((n_{B1}+J)!/(n_{B1}+J-1)!\right)+\\ &\quad \log\left(n_{A1,1}!/(n_{A1,1}-1)!\right)+\log\left(n_{B1,1}!/(n_{B1,1}+1)!\right)\\ &= \log\left((n_{B1}+J)/(n_{A1}+J-1)\right)+\log\left(n_{A1,1}/(n_{B1,1}+1)\right)\\ &= \log\left((n_{B1}+1)/(n_{B1,1}+1)\right)-\log\left(n_{A1}/n_{A1,1}\right)+\\ &\quad \log\left((n_{B1}+J)/(n_{B1}+1)\right)-\log\left((n_{A1}+J-1)/n_{A1}\right).\end{aligned}$$

Using the property $1 \le x < y \Rightarrow (y+1)/(x+1) < y/x$, we get

$$\begin{aligned}\Delta Cost1 < &\log\left(n_{B1}/n_{B1,1}\right)-\log\left(n_{A1}/n_{A1,1}\right)+\\ &\log\left((n_{B1}+J)/(n_{B1}+1)\right)-\log\left((n_{A1}+J)/(n_{A1}+1)\right).\end{aligned}$$

Similarly, we construct two intervals A2 and B0 by moving the first instance from B1 to A1. This time, the cost variation $\Delta Cost2$ of the discretization subject to

$$\begin{aligned}\Delta Cost2 < &\log\left(n_{A1}/n_{A1,1}\right)-\log\left(n_{B1}/n_{B1,1}\right)+\\ &\log\left((n_{A1}+J)/(n_{A1}+1)\right)-\log\left((n_{B1}+J)/(n_{B1}+1)\right).\end{aligned}$$

We notice that the two cost variations $\Delta Cost1$ and $\Delta Cost2$ have exactly opposite values. Therefore, the cost variation of the discretization is strictly negative and the initial discretization could not be optimal. As this is contradictory with the initial assumption, the claim follows. ■

Based on Theorem 2, important reductions in time consumption can be obtained in the optimization algorithms, since only the boundary points need to be evaluated to find optimal or near-optimal discretizations.

Theorem 3: In a Bayes optimal SDM model distributed according to the three-stage prior, there is no pair of adjacent intervals each containing one single instance.

Proof:

Let A and B be two adjacent intervals each containing one single instance related to different classes (otherwise, the intervals should be merged according to Theorem 2). We compute the cost variation $\Delta Cost$ of the discretization after the

merge of the two intervals into a new interval $A \cup B$, bringing the number of intervals from I down to I-1.

$$\Delta Cost = \log\binom{n+I-2}{I-2} - \log\binom{n+I-1}{I-1}$$
$$+\left(\log\binom{n_{A\cup B}+J-1}{J-1} + \log\left(n_{A\cup B}!/n_{A\cup B,1}!n_{A\cup B,2}!...n_{A\cup B,J}!\right)\right)$$
$$-\left(\log\binom{n_A+J-1}{J-1} + \log\left(n_A!/n_{A,1}!n_{A,2}!...n_{A,J}!\right)\right)$$
$$-\left(\log\binom{n_B+J-1}{J-1} + \log\left(n_B!/n_{B,1}!n_{B,2}!...n_{B,J}!\right)\right)$$

$$\Delta Cost = \log\left(I-1/n+I-1\right) + \log\left(\left(n_{A\cup B}+J-1\right)!\left(J-1\right)!/\left(n_A+J-1\right)!\left(n_B+J-1\right)!\right)$$
$$-\sum_{j=1}^{J}\log\binom{n_{A\cup B,j}}{n_{A,j}}$$

Since $n_A = n_B = 1$ and $n_{A\cup B} = 2$, we obtain:

$$\Delta Cost = \log\left(I-1/n+I-1\right) + \log\left(J+1/J\right).$$
$$\Delta Cost \le 0 \Leftrightarrow I \le nJ+1.$$

The cost variation is always strictly negative after the merge of two adjacent singleton intervals. The claim follows. ■

The main interest of Theorem 3 is to provide an intuitive evaluation of the asymptotic behaviour of the three-stage prior: a discretization model based on two adjacent singleton interval is not optimal for a good generalization. Theorem 4 is another interesting property resulting from the three-stage prior.

Theorem 4: In a SDM model distributed according to the three-stage prior and in the case of two classes, the discretization composed of one single interval is more probable than the discretization composed of one interval per instance.

Proof:

Let $Cost_1$ and $Cost_n$ be the value of the MODL criterion in the case of one interval and of n intervals.

$$Cost_1 = \log\left(n\right) + \log\left(n+1\right) + \log\left(n!/n_1!n_2!\right).$$
$$Cost_n = \log\left(n\right) + \log\left(\left(2n-1\right)!/\left(n-1\right)!n!\right) + n\log\left(2\right).$$

Since $n+1 = \left(n+1\right)!/1!n! \le \left(2n-1\right)!/\left(n-1\right)!n!$ and $n!/n_1!n_2! < 2^n$, the claim follows. ■

A close inspection of formula 3 reveals that the second term, which encodes the number of choices when dividing the string S into I intervals, includes the possibility of empty intervals. This is a deliberate choice in the three-stage prior that favours discretizations with small numbers of intervals. If we exclude this

possibility, the theorems 2, 3 and 4 are no longer true. These empty intervals do not need to be explored in optimization algorithms since adding empty intervals is always penalized by an increase of the evaluation criterion.

To summarize, the MODL discretization method is directly based on the Bayesian approach. The definitions of the SDM models and the three-stage prior are both general enough to capture the complexity of real data and simple enough to allow an exact calculation of the probabilities involved in the Bayes rule. This provides the guarantee of optimality in the choice of the discretization model, in the context of the three-stage prior. The bias resulting from the choice of this prior leads to interesting demonstrable properties.

3 The MODL algorithm

Once the optimality of this evaluation criterion is established, the problem is to design a search algorithm in order to find a discretization model that minimizes the criterion. In this section, we present three algorithms that represent different trade-off between the time complexity of the search algorithm and the quality of the discretizations.

3.1 Optimal algorithm

The MODL evaluation criterion consists of an evaluation of the partition of the string S into I intervals (first and second term in formula 3) and of the sum of the evaluation of the intervals S_i (third and last term in formula 3). The first part of the MODL criterion (value of the partition) depends only upon the sample size n and the number of intervals I and the second part (value of the intervals) is cumulative on the intervals. Hence, if a partition of S into I intervals S_1, S_2, … S_I is a MODL optimal discretization of S, then a partition of S-S_1 into I-1 intervals S_2, … S_I is a MODL optimal discretization of S-S_1. This interesting property is sufficient to adapt the dynamic programming algorithm presented in (Fischer, 1958; Lechevallier, 1990; Fulton et al., 1995; Elomaa and Rousu, 1996). We summarize in Table 1 this dynamic programming algorithm applied to the MODL discretization method.

The main loop of the algorithm finds the optimal discretizations of S into exactly k intervals ($1 \leq k \leq n$), and thus to obtain the optimal MODL discretization of the string S. The algorithm runs in $O(n^3)$ time. Although it is not applicable in the case of large databases, this optimal algorithm is helpful to evaluate the quality of search heuristics such as these presented in the next sections.

Table **1**: Dynamic programming algorithm

Let $S^{i,j}$ be the substring of S consisting of the instances from i to j. $S^{1,n} = S$.
Let Disc($S^{i,j}$, k) be the optimal discretization of $S^{i,j}$ into exactly k intervals.

- For each k, $1 \leq k \leq n$,
  - For each j, $1 \leq j \leq n$,
    - If k=1, Disc($S^{1,j}$, 1) = {$S^{1,j}$}
    - If k>1, Disc($S^{1,j}$, k) is obtained by minimizing the MODL value of all discretizations Disc($S^{1,i}$, k-1) U {$S^{i+1,j}$} for $1 \leq i \leq j$.

3.2 Greedy heuristic

In this section, we present a standard greedy bottom-up heuristic. The method starts with initial single value intervals and then searches for the best merge between adjacent intervals. This merge is performed if the MODL value of the discretization decreases after the merge and the process is reiterated until not further merge can decrease the criterion.

Table **2**: Optimized greedy bottom-up merge algorithm

- Initialization
  - Sort the explanatory attribute values: O(n log(n))
  - Create an elementary interval for each value: O(n)
  - Compute the value of this initial discretization: O(n)
  - Compute the Δvalues related to all the possible merges: O(n)
  - Sort the possible merges: O(n log(n))
- Optimization of the discretization
  Repeat the following steps: at most n steps
  - Search for the best possible merge: O(1)
  - Merge and continue if the best merge decreases the discretization value
    - Compute the Δvalues of the two intervals adjacent to the merge: O(1)
    - Update the sorted list of merges: O(log(n))

With a straightforward implementation of the algorithm, the method runs in O(n^3) time. However, the method can be optimized in O(n log(n)) time owing to an algorithm similar to that presented in (Boullé, 2004) and summarized in Table 2. The algorithm is mainly based on the additivity of the evaluation criterion. Once a discretization is evaluated, the value of a new discretization resulting from the merge between two adjacent intervals can be evaluated in a single step, without scanning all the other intervals. Minimizing the value of the discretizations after the merges is the same as maximizing the related variation of value Δvalue. These Δvalues can be kept in memory and sorted in a maintained sorted list (such as an AVL binary search tree for example), or more simply in a priority-queue. After a merge is completed, the Δvalues need to be updated only for the new interval and its adjacent intervals to prepare the next merge step.

3.3 Post-optimization algorithm

Compared to the optimal algorithm, the greedy heuristic is time efficient, but it may fall into a local optimum. First, the greedy heuristic may stop too soon and produce too many intervals. Second, the boundaries of the intervals may be sub-optimal since the merge decisions of the greedy heuristic are never rejected. Given that the MODL criterion is optimal, a time efficient post-optimization of the discretization is meaningful.

We propose a new post-optimization algorithm based on hill-climbing search in the neighborhood of a discretization. The neighbors of a discretization are defined with combinations of interval splits and interval merges, as pictured in Figure 1.

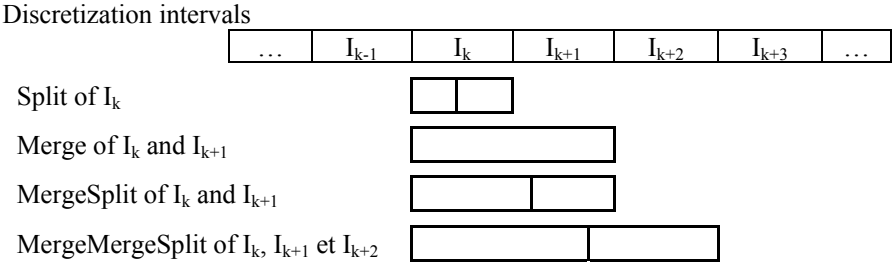

Figure **1**: Combinations of interval splits and interval merges used to explore the neighborhood of a discretization

In a first stage called *exhaustive merge*, the greedy heuristic merge steps are performed unconditionally until the discretization consists of a single interval. The best encountered discretization is then memorized. This stage allows escaping local minima with several successive merges and needs $O(n \log(n))$ time.

In a second stage called *greedy post-optimization*, all the neighbours of the best discretization consisting of Splits, MergeSplits and MergeMergeSplits are evaluated. The best improvement of the discretization is performed if the evaluation criterion decreases, and this steps is reiterated until no neighbour can decrease the value of the discretization. The calculation of all these discretization neighbours can be done in $O(n)$ time and inserted in sorted lists in $O(n \log(n))$ time, like in the greedy bottom-up merge algorithm. Each improvement of the discretization requires $O(\log(n))$ time to maintain the data structures kept in memory. This second stage converges very quickly and requires only a few steps, so that its overall time complexity is still $O(n \log(n))$.

The post-optimization holds two straightforward notable properties. The first one is that post-optimizing can only improve the results of the standard greedy heuristic since its comes after. The second one is that in case of attributes whose optimal discretization consists of one single interval, the optimum is necessarily found owing to the exhaustive merge stage of the post-optimization.

4 Experiments

In our experimental study, we compare the MODL discretization method with other discretization algorithms. In this section, we introduce the evaluation protocol, the alternative evaluated discretization methods and expose the evaluation results on real and artificial datasets.

4.1 The evaluation protocol

Discretization is a general purpose preprocessing method that can be used for data exploration or data preparation in data mining. While they are critical in the case of decision tree methods, discretization methods can also be used for bayesian networks, rule-set algorithms or logistic regression. However, discretization methods have mainly been evaluated using decision trees (Fayyad and Irani, 1992; Kohavi and Sahami, 1996; Elomaa and Rousu, 1999; Liu and al. 2002) and less frequently using naïve Bayes methods (Dougherty et al., 1995). In their experiments, (Kohavi and Sahami, 1996) report that entropy-based discretization methods perform better that error-based methods. Maximizing the accuracy on each attribute can hide the variations of the class conditional density and hinder improvements of the classifier accuracy when the attributes are combined. In the case of naïve Bayes classifiers, (Dougherty et al., 1995) demonstrate that using any discretization algorithm outperforms the naïve Bayes algorithm with the normality assumption for continuous attributes. (Elomaa and Rousu, 1999) show that using optimal multi-splitting discretization algorithms does not bring a clear accuracy advantage over binary splitting in the case of decision trees.

Although these evaluations bring many insights on the impact of discretization methods on classifiers, the contribution of the discretization is not always clear. For example, decision tree are composed of several modules including a preprocessing algorithm, a selection criterion, a stopping rule and a pruning algorithm. The discretization preprocessing step can be done once at the root of the tree or repeated at each node of the tree. It can use a binary-splitting strategy or a multi-splitting strategy (better partition, but more fragmented subsequent data). The performances of such classifiers result from complex interactions between these modules and strategies.

In order to evaluate the intrinsic performance of the discretization methods and eliminate the bias of the choice of a specific induction algorithm, (Zighed et al. 1999) consider each discretization method as an elementary inductive method that predicts the local majority class in each learned interval. They apply this approach on the waveform dataset (Breiman, 1984) to compare several discretization methods on the accuracy criterion.

We extend this protocol as in (Boullé, 2003) and evaluate the elementary discretization inductive methods using all the continuous attributes contained in several datasets. This allows us to perform hundreds of experiments instead of at most tens of experiments. The discretizations are evaluated for three criteria: accuracy, robustness (test accuracy/train accuracy) and number of intervals. To facilitate the interpretation of the very large set of experimental results, we first proceed with a multi-criteria analysis based on gross global summaries of the

results. We then come to a more detailed analysis by the means of curves representing all the experiments sorted by increasing difference with the MODL method. Finally, we zoom into some specific experiments in order to provide both an illustration on some sample MODL discretizations and an explanation of typical differences of behavior between the MODL method and the alternative methods. In addition, we perform extensive experiments on artificial datasets designed to help understand the performance, bias and limits of each discretization method.

4.2 The evaluated methods

The discretization methods studied in the comparison are:

- MODL
- MDLPC (Fayyad and Irani, 1992)
- BalancedGain (Kononenko et al., 1984)
- Fusinter (Zighed et al, 1998)
- Khiops (Boullé, 2004; Boullé, 2003)
- ChiMerge (Kerber, 1991)
- ChiSplit (Bertier and Bouroche, 1981)
- Equal Frequency
- Equal Width

The MDLPC method is a greedy top-down split method, whose evaluation criterion is based on the Minimum Description Length Principle (Rissanen, 1978). At each step of the algorithm, the MDLPC evaluates two hypotheses (to cut or not to cut the interval) and chooses the hypothesis whose total encoding cost (model plus exceptions) is the lowest. The BalancedGain method exploits a criterion similar to the GainRatio criterion (Quinlan, 1993): it divides the entropy-based InformationGain criterion by the log of the arity of the partition in order to penalize excessive multisplits. This method can be embedded into a dynamic-programming based algorithm, as studied in (Elomaa and Rousu, 1999). The Fusinter method is a greedy bottom-up method that exploits an uncertainty measure sensitive to the sample size. Its criterion employs a quadratic entropy term to evaluate the information in the intervals and is regularized by a second term in inverse proportion of the interval frequencies. The Khiops algorithm uses the chi-square criterion in a global manner to evaluate all the intervals of a discretization, in a greedy bottom-up merge algorithm. Its stopping criterion has been enhanced in (Boullé, 2003) in order to provide statistical guarantees against overfitting: discretizations of independent attributes consist of a single interval with a user defined probability. The ChiMerge method is a greedy bottom-up merge method that locally exploits the chi-square criterion to decide whether two adjacent intervals are similar enough to be merged. The ChiSplit method, comparable to the ChiMerge method, uses a greedy top-down split algorithm.

The MODL, MDLPC and BalancedGain methods have an automatic stopping rule and do not require any parameter setting. For the Fusinter criterion, we use the regularization parameters recommended in (Zighed et al., 1998). The Khiops probability parameter is set to 0.95. For the ChiMerge and ChiSplit methods, the significance level is set to 0.95 for the chi-square test threshold. The Equal Width and Equal Frequency unsupervised discretization methods use a number of intervals

set to 10. Since several instances can share the same descriptive value, the actual number of intervals can be less than expected in the case of the two unsupervised methods.

The MDLPC, ChiMerge and ChiSplit criteria are local to two adjacent intervals: these methods cannot be optimized globally on the whole set of intervals. The Khiops criterion is global, but its stopping criterion is based on the statistic of the variation of the criterion during the merge process used in the algorithm. Thus, the Khiops search algorithm cannot be changed. The BalancedGain and the Fusinter method are the only other compared methods with a global criterion. In order to allow a fair comparison, we use exactly the same search algorithm (greedy bottom-up heuristic followed by the post-optimization) both for the MODL, BalancedGain and Fusinter methods.

4.3 Real data experiments

In this section, we test the evaluated methods on real data. After introducing the datasets and the details of the evaluation protocol, we comment the aggregated results using a multi-criteria analysis. Then, we exhibit the relative differences between the methods on the complete set of experiments. Finally, we illustrate some sample discretization using histograms.

4.3.1 The datasets

We gathered 15 datasets from U.C. Irvine repository (Blake, 1998), each dataset has at least one continuous attribute and at least a few tens of instances for each class value in order to perform reliable tenfold cross-validations. Table 3 describes the datasets; the last column corresponds to the relative frequency of the majority class.

Table **3**: Datasets

Dataset	Continuous Attributes	Nominal Attributes	Size	Class Values	Majority Class
Adult	7	8	48842	2	76.07
Australian	6	8	690	2	55.51
Breast	10	0	699	2	65.52
Crx	6	9	690	2	55.51
German	24	0	1000	2	70.00
Heart	10	3	270	2	55.56
Hepatitis	6	13	155	2	79.35
Hypothyroid	7	18	3163	2	95.23
Ionosphere	34	0	351	2	64.10
Iris	4	0	150	3	33.33
Pima	8	0	768	2	65.10
SickEuthyroid	7	18	3163	2	90.74
Vehicle	18	0	846	4	25.77
Waveform	21	0	5000	3	33.92
Wine	13	0	178	3	39.89

The discretizations are performed on the 181 continuous attributes of the datasets, using a ten times stratified tenfold cross-validation. We have re-implemented the alternative discretization methods in order to eliminate any variance resulting from different cross-validation splits. In order to determine whether the performances are significantly different between the MODL method and the alternative methods, the t-statistics of the difference of the results is computed. Under the null hypothesis, this value has a Student's distribution with 99 degrees of freedom. The confidence level is set to 1% and a two-tailed test is performed to reject the null hypothesis.

4.3.2 Multi-criteria analysis of the results

The whole result tables related to the 181 attribute discretizations are too large to be printed in this paper. The results are summarized by datasets in the Appendix. Table 10 reports the geometric mean of the accuracy for all the attributes in each dataset, and the global summary by attribute and by dataset. Table 11 details the number of MODL significant wins and losses for each dataset and for all the attributes. Although these three global indicators look consistent, the summary by dataset seems preferable since it gives the same weight to each dataset whatever their number of attributes is. The robustness results are presented in Tables 12 and 13, and the number of intervals results in Tables 14 and 15.

In multi-criteria analysis, a solution *dominates* (or is *non-inferior* to) another one if it is better for all criteria. A solution that cannot be dominated is *Pareto optimal*: any improvement of one of the criteria causes a deterioration on another criterion. The *Pareto surface* is the set of all the Pareto optimal solutions.

In order to analyze both the accuracy and robustness results, we report the dataset geometric means on a two-criteria plan in Figure 2, with the accuracy on the

x-coordinate and the robustness on the y-coordinate. Similarly, we report the accuracy and the number of intervals in Figure 3. Each point in these figures represents the summary of 18,100 experiments. An inspection of the results tables presented in the Appendix shows that a relative difference of about 0.5% between two summary points reflects significant differences between the two corresponding methods. The multi-criteria figures are thus reliable and informative: they allow us to clearly differentiate the behavior of almost all the methods.

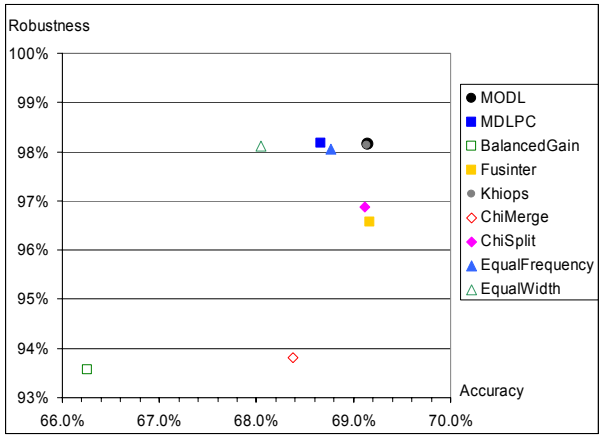

Figure **2**: Bi-criteria evaluation of the discretization methods for the accuracy and the robustness, using datasets geometric means

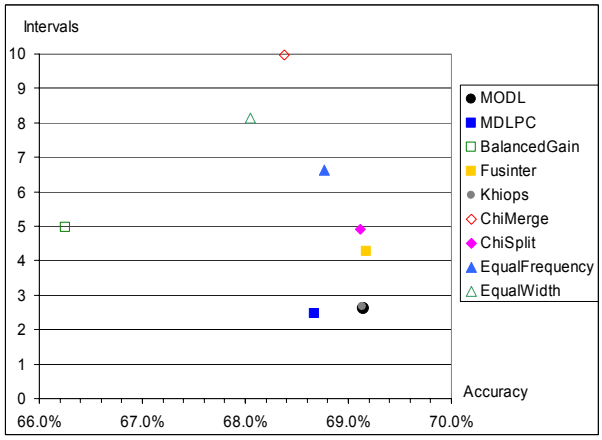

Figure **3**: Bi-criteria evaluation of the discretization methods for the accuracy and the number of intervals, using datasets geometric means

The accuracy is the most common evaluated criterion. Looking at this criterion shows that the four methods MODL, Fusinter, Khiops and ChiSplit perform equally well. They are followed by the MDLPC and EqualFrequency methods in a second group. Last come the ChiMerge method, the EqualWidth method and finally the

BalancedGain method.

The robustness is an interesting criterion that allows to estimate whether the performance on the train data is a good prediction of the performance on the test data. The higher is the robustness, the most accurate will be a ranking of attributes based on their train accuracy. This can be critical in the case of classifiers such as decision trees that incorporate an attribute selection method to build the next node of the tree. The unsupervised methods exhibit a very good robustness in spite of their large number of intervals. They are not subject to over-fitting since they explore one single hypothesis to discretize the continuous attributes. The MODL, MDLPC and Khiops methods are as robust as the unsupervised methods. They are followed by the Fusinter and ChiSplit methods in a second group. At last, the BalancedGain and ChiMerge methods come with a far weaker robustness.

The number of intervals criterion is potentially related to the simplicity of the discretizations. In the case of decision trees, discretizations with the same accuracy but with fewer intervals are preferable since they cause less fragmentation of the data in the sub-nodes of the tree. The MODL, MDLPC and Khiops methods clearly dominate the other methods on this criterion. Then come the BalancedGain, Fusinter and ChiSplit methods with about twice the number of intervals than that of the leading methods, followed by the unsupervised EqualFrequency and EqualWith methods and finally the ChiMerge method.

If we take into account the three criterion together, the BalancedGain and ChiMerge methods are clearly dominated by all the other methods, followed by the EqualWith method. The EqualFrequency and MDLPC methods obtain similar results on accuracy and robustness, but the MDLPC methods produces far less intervals. Among the four most accurate methods, the Fusinter and ChiSplit methods are clearly dominated both on robustness and number of intervals. The MODL and Khiops methods are Pareto optimal: they dominate all the other methods on the three evaluated criteria. However, they cannot be distinguished from each other.

4.3.3 Detailed differences between the methods

In order to analyze the relative differences of accuracy for the 181 attributes in more details, we collect all the geometric mean ratios per attribute in ascending order. Figure 4 shows the repartition function of the relative differences of accuracy between the MODL method and the other discretization methods. Each point in this repartition function is the summary of 100 discretization experiments performed on the same attribute. The robustness results are presented in Figure 5 and the number of intervals results in Figure 6.

Such repartition functions represent a convenient tool for the fine grain analysis of the differences between methods, in complement with the multi-criteria analysis carried out on the coarse dataset geometric means. A flat curve reflects two methods that do not differentiate on any of the experiments. A symmetric curve correspond to methods that globally perform equally well, but with differences among the experiences. An unbalanced curve reveals a situation of dominance of one method over the other, with insights on the intensity of the domination and on the size of the region of dominance. On the left of Figures 4 and 5, the MODL method is

dominated by the other methods and, on the right, it outperforms the other algorithms. It is the reverse situation for Figure 6.

Concerning the accuracy criterion presented in Figure 4, the Khiops curve is almost flat, with about 80% of the attributes having exactly the same performances than the MODL method. The two other most accurate Fusinter and ChiSplit methods exhibit balanced but slightly dissymmetric curves: they dominate the MODL method in about 10% of the attributes and are dominated – less significantly - in about 20% of the attributes. Compared to the MDLPC method, the MODL method is between 0 and 3% less accurate in about 10% of the attributes, but is between 3 and 10% more accurate in about 10% of the attributes. The average relative difference of 0.7% between the MODL and MDLPC methods is thus significant and reflects potential large differences of accuracy on individual attributes. The other methods are far less accurate than the MODL method on a large proportion of the attributes.

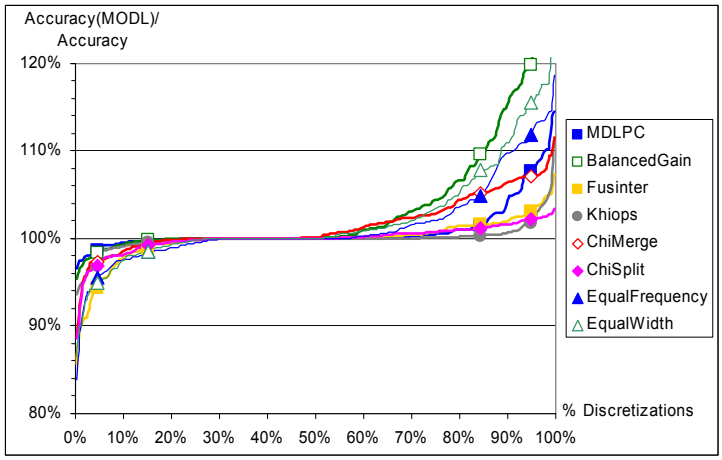

Figure **4**: Repartition function of the relative differences of accuracy between the MODL method and the other discretization methods

Dealing with the robustness pictured in Figure 5, the MODL, MDLPC and Khiops methods demonstrate the same level of performance on almost all the attributes. The two unsupervised EqualFrequency and EqualWith methods have the same average level of robustness that the three preceding methods. However, they faintly dominate on 40% of the attributes and are more strongly dominated on 20% of the attributes. The two accurate Fusinter and ChiSplit methods are significantly dominated on a large proportion of the attributes. Finally, the remaining BalancedGain and ChiMerge methods are strongly dominated. The BalancedGain exhibits a very sharp transition for the right-most 20 % of the discretizations. In the last 10% (not shown in the figure for scalability reasons), the robustness of the MODL method goes between 25% and 60% higher than that of the BalancedGain method.

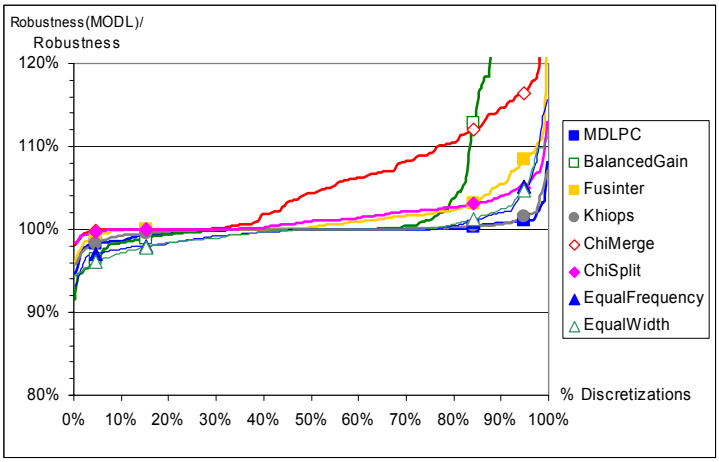

Figure **5**: Repartition function of the relative differences of robustness between the MODL method and the other discretization methods

Regarding the number of intervals displayed in Figure 6, the three robust MODL, MDLPC and Khiops methods express similar behaviour on a large majority of the attributes. All the other curves are heavily asymmetrical. The methods produce from 2 to 4 times more intervals on average, and up to one hundred times more intervals in the extreme cases. Once more, the BalancedGain curve distinguishes from the other curves with significantly smaller number of intervals that the MODL method in half of the discretization and the largest numbers of intervals for 15% of the attributes.

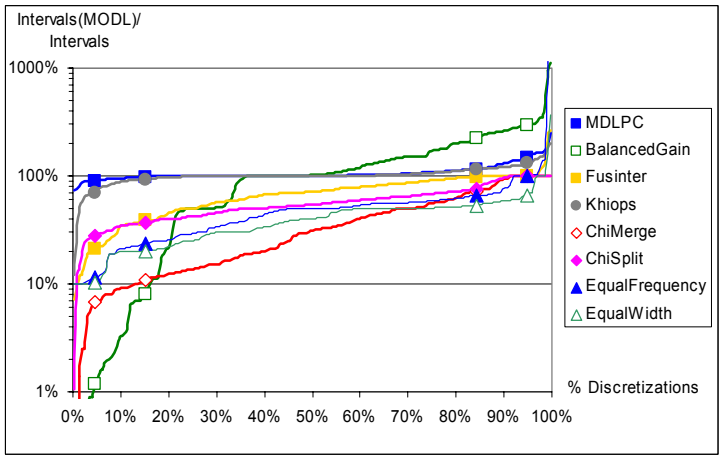

Figure **6**: Repartition function of the relative differences of number of intervals between the MODL method and the other discretization methods

The atypical behavior of the BalancedGain method requires some explanation. The BalancedGain criterion is the ratio of the InformationGain by the log of the

number of intervals. These two functions are increasing functions with respect to the number of intervals. According to their mutual position, the shape of the ratio function exhibit radically different behaviors. In the case of random or very noisy attributes, the InformationGain is almost null for small numbers of intervals and steadily increases toward its maximum value. The resulting BalancedGain ratio is an increasing function of the number of intervals, with subsequent discretizations containing very large numbers of intervals and having very poor robustness. On the opposite case of very informative attributes, the first split brings a lot of information. This translates the InformationGain first values upward and flatten the increase rate of the function. The ensuing BalancedGain ratio turns into a decreasing function of the number of intervals, with resulting discretizations having only two intervals and a very good robustness. This is equivalent to performing a binary-split using the InformationGain function. Between these two extreme situations, any behavior can theoretically be observed: this happens to be infrequent in the UCI datasets.

The BalancedGain criterion, very similar to the GainRatio criterion used in C4.5 (Quinlan, 1993), has been experimented in the context of decision trees in (Elomaa and Rousu, 1999). The reported prediction accuracy results are reasonably good, even though the multi-split discretizations produced by the BalancedGain method are not convincing. Indeed, the selection module of the decision tree always prefers the most informative attributes, which benefit from a very good binary-split (though a weak multi-split) and ignores the noisy attributes which suffer from severe over-splitting. Thus, the complex machinery of a classifier can sometimes hide the effect of poor quality multi-split discretizations.

4.3.4 Some sample discretizations

In this section, we select four attributes among the 181 benchmark attributes, both to illustrate sample discretizations through histograms and to focus on qualitative differences between the evaluated methods.

4.3.4.1 V1 attribute of the Waveform dataset

This attribute corresponds to the least predictive attribute among the Waveform attributes. Figure 7 presents the waveform class frequency histogram using 20 bins, computed on the whole dataset.

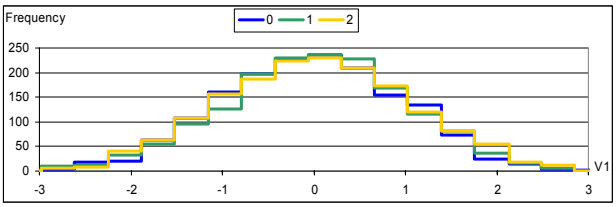

Figure **7**: Class frequency histogram for the V1 attribute of the Waveform dataset

The three classes look equidistributed, meaning that the V1 attribute holds little conditional information about the class density. The results obtained by the

discretization methods during the ten times tenfold cross-validation are reported in Table 4. The three robust MODL, MDLPC and Khiops methods build exactly one interval.

Table **4**: Mean and standard deviation discretization results for the V1 attribute of the Waveform dataset

Method	Accuracy		Robustness		Intervals	
	Mean	Std Dev	Mean	Std Dev	Mean	Std Dev
MODL	33.9	± 0.1	100.0	± 0.3	1.0	± 0.0
MDLPC	33.9	± 0.1	100.0	± 0.3	1.0	± 0.0
BalancedGain	34.6	± 2.0	70.1	± 4.4	431.0	± 6.7
Fusinter	33.8	± 1.8	91.9	± 5.3	8.3	± 1.1
Khiops	33.9	± 0.1	100.0	± 0.3	1.0	± 0.0
ChiMerge	33.7	± 2.0	86.0	± 5.6	40.2	± 5.4
ChiSplit	33.4	± 1.0	95.5	± 4.1	5.0	± 2.8
EqualFrequency	33.1	± 1.9	93.0	± 5.7	10.0	± 0.0
EqualWidth	33.1	± 1.9	94.4	± 5.8	10.0	± 0.0

4.3.4.2 V7 Attribute of the Waveform dataset

This attribute corresponds to the most predictive attribute among the Waveform attributes. Figure 8 presents the class frequency histogram and Table 5 the results of the discretizations.

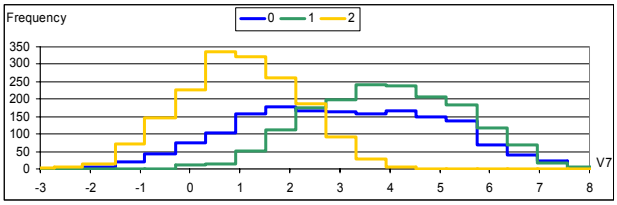

Figure **8**: Class frequency histogram for the V7 attribute of the Waveform dataset

Table **5**: Mean and standard deviation discretization results for the V7 attribute of the Waveform dataset

Method	Accuracy		Robustness		Intervals	
	Mean	Std Dev	Mean	Std Dev	Mean	Std Dev
MODL	57.5	± 1.2	99.9	± 2.2	6.0	± 0.0
MDLPC	57.4	± 1.2	99.9	± 2.2	6.0	± 0.2
BalancedGain	57.4	± 1.2	99.9	± 2.2	2.0	± 0.0
Fusinter	57.4	± 1.2	99.8	± 2.3	6.4	± 0.6
Khiops	57.5	± 1.1	99.7	± 2.2	6.5	± 0.9
ChiMerge	56.6	± 1.4	95.3	± 2.7	55.9	± 6.4
ChiSplit	57.3	± 1.3	99.1	± 2.5	14.8	± 2.5
EqualFrequency	57.5	± 1.1	100.0	± 2.2	10.0	± 0.0
EqualWidth	57.0	± 1.2	100.0	± 2.4	10.0	± 0.0

The MODL methods produces exactly 6 intervals (with a null variance). Figure

9 reports the class conditional density histogram corresponding to the MODL discretization build on the whole Dataset. The discretization behaves like a density estimator, with a focus on the variations of the class densities. The MDLPC, Fusinter and Khiops methods build approximatively the same intervals and obtain the same level of performances on accuracy and robustness. The BalancedGain method produces a binary-split. The other methods builds too many intervals.

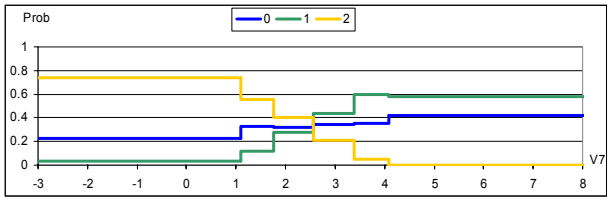

Figure **9**: Class conditional density histogram for the V7 attribute of the Waveform dataset, based on the MODL discretization

4.3.4.3 A30 attribute of the Ionosphere dataset

A close look at the result tables presented in the Appendix indicates a special behavior of the Ionosphere dataset, where the MDLPC method is largely dominated by the other methods. We chose the A30 attribute as one of the most illustrative attributes to explain this behavior. Figure 10 presents the class frequency histogram and Table 6 the results of the discretizations.

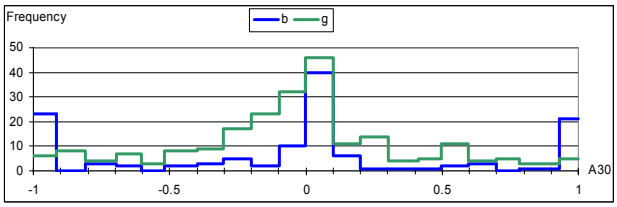

Figure **10**: Class frequency histogram for the A30 attribute of the Ionosphere dataset

Table **6**: Mean and standard deviation discretization results for the A30 attribute of the Ionosphere dataset

Method	Accuracy		Robustness		Intervals	
	Mean	Std Dev	Mean	Std Dev	Mean	Std Dev
MODL	80.6	± 5.9	97.9	± 7.8	5.0	± 0.0
MDLPC	73.2	± 5.9	99.4	± 8.8	3.0	± 0.3
BalancedGain	72.4	± 6.8	76.0	± 7.5	78.3	± 20.4
Fusinter	80.5	± 5.9	97.8	± 7.8	5.0	± 0.1
Khiops	80.6	± 5.9	97.9	± 7.8	5.0	± 0.0
ChiMerge	75.5	± 7.1	86.3	± 8.9	33.2	± 5.6
ChiSplit	80.9	± 5.6	98.2	± 7.5	6.0	± 1.0
EqualFrequency	73.3	± 7.0	99.8	± 10.1	9.0	± 0.0
EqualWidth	70.7	± 6.5	100.2	± 10.2	10.0	± 0.0

The four more accurate methods build between 5 and 6 intervals, whereas the MDLPC methods produces only 3 intervals. Figure 11 shows the class conditional density histogram obtained with the MODL discretization. Two density peaks are identified in the borders of the value domain. These peaks are also identified by the MDLPC method, with exactly the same bounds. The third density peak, in the center, is so thin that it cannot be discovered on the regular class frequency histogram pictured in Figure 10. Nevertheless, it contains 37 instances and thus corresponds to a reliable class density estimation. This kind of pattern is easily detected by the bottom-up methods, and by the ChiSplit method that tends to produce too many intervals. It is harder to discover for the MDLPC top-down algorithm since it requires two successive splits with the first one not very significant.

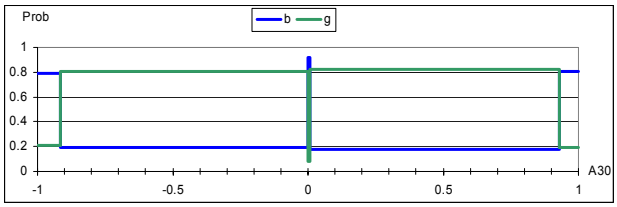

Figure **11**: Class conditional density histogram for the A30 attribute of the Ionosphere dataset, based on the MODL discretization

4.3.4.4 V11 attribute of the Wine dataset

This attribute corresponds to one the largest loss in accuracy for the MODL method when compared to the other accurate Fusinter and ChiSplit methods. Figure 12 presents the class frequency histogram and Table 7 the results of the discretizations.

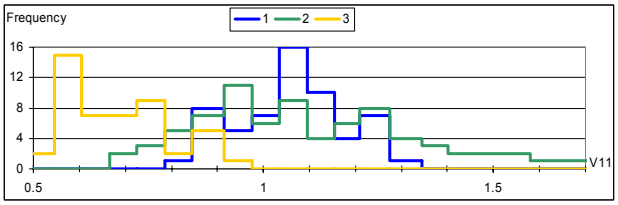

Figure **12**: Class frequency histogram for the V11 attribute of the Wine dataset

Table **7**: Mean and standard deviation discretization results for the V11 attribute of the Wine dataset

Method	Accuracy		Robustness		Intervals	
	Mean	Std Dev	Mean	Std Dev	Mean	Std Dev
MODL	56.8	± 7.1	93.4	± 14.2	2.9	± 0.6
MDLPC	58.8	± 8.1	94.9	± 14.2	2.8	± 0.9
BalancedGain	58.6	± 6.7	98.8	± 12.1	2.0	± 0.0
Fusinter	66.3	± 9.9	94.7	± 15.7	4.7	± 0.7
Khiops	58.7	± 6.7	96.2	± 13.9	2.3	± 0.8
ChiMerge	62.0	± 10.4	89.4	± 16.7	6.2	± 1.1
ChiSplit	62.5	± 10.5	91.0	± 16.5	6.5	± 0.7
EqualFrequency	62.4	± 9.8	94.8	± 16.1	9.5	± 0.5
EqualWidth	63.0	± 11.1	97.2	± 18.7	9.1	± 0.3

The Wine Dataset contains 178 instances. In a tenfold cross-validation, there are less than 20 instances in the test set: one more correctly classified instance brings a 5% increase in accuracy. This reflects in Table 7 with important differences in accuracy, but large standard deviations. Figure 13 shows that the MODL method builds only three intervals: there are not enough instances to build new reliable intervals.

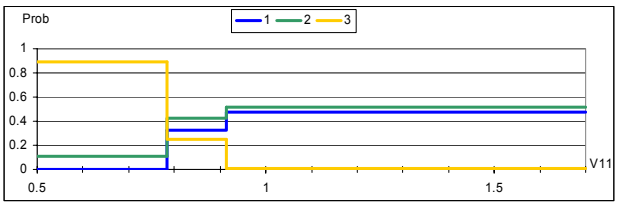

Figure **13**: Class conditional density histogram for the attribute V11 of the Wine dataset, based on the MODL discretization

Figure 14 shows the five intervals of the Fusinter discretization. The last interval is a mixture of classes, since the Fusinter criterion tries to compromise class discrimination and minimum frequency per interval.

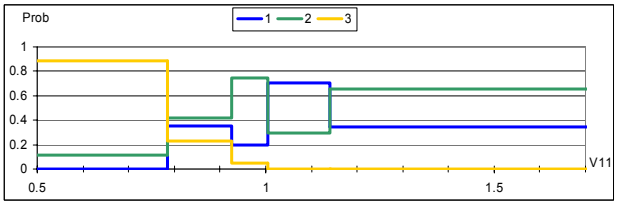

Figure **14**: Class conditional density histogram for the V11 attribute of the Wine dataset, based on the Fusinter discretization

Figure 15 shows the seven intervals of the ChiSplit discretization. The class densities are more contrasted than with the Fusinter method, but the methods suffers from over-fitting.

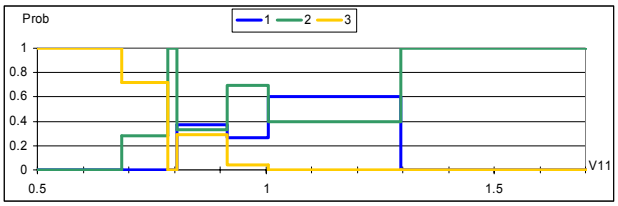

Figure **15**: Class conditional density histogram for the V1 attribute of the Wine dataset, based on the ChiSplit discretization

4.4 Synthetic data experiments

In this section, we focus on the tested supervised discretization methods and try to characterize their bias and performances. We use synthetic datasets, which allow comparing the tested discretization methods with an oracle that knows the true class distribution.

4.4.1 Noise pattern

The purpose of the noise pattern experiment is to evaluate the noise resistance of the discretization methods, under variation of the sample size. In the case of pure noise data, the oracle discretization builds a single interval, meaning that there is no class information in the explanatory attribute. This experiment is strongly biased in favor of the top-down discretization heuristics, since they just have to reject one single split to be as good as the oracle. On the contrary, the bottom-up methods have to accept many merge decisions before they can produce a single interval discretization.

The *noise pattern* dataset consists of an explanatory continuous attribute independent from the class attribute. The explanatory attribute is uniformly distributed on the [0, 1] numerical domain and the class attribute consists of two equidistributed class values. The evaluated criterion is the number of unnecessary intervals, since the test accuracy is always 0.5 whatever the number of intervals is in the discretizations. The experiment is done on a large number of sample sizes ranging from 100 instances to 100,000 instances. In order to obtain reliable results, it is performed 10,000 times. Figure 16 presents the average unnecessary interval number obtained by the tested discretization methods, for different sample sizes.

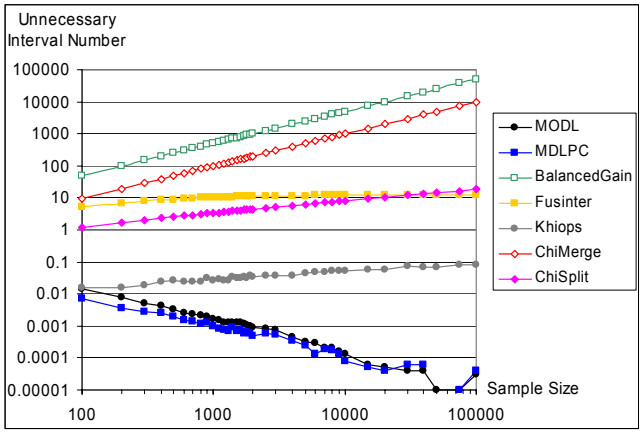

Figure **16**: Mean of the unnecessary interval number of the discretizations of an explanatory attribute independent from the class attribute

The BalancedGain, ChiMerge and ChiSplit methods always produce more than one interval, and the number of intervals increases with the sample size. The Fusinter method builds between 5 and 10 intervals with a maximum when the sample size is about 5,000. The Khiops method is designed to produce one single interval with probability greater than 0.95 when the explanatory attribute is independent from the class attribute. This is confirmed by the experiment with an average unnecessary interval number ranging from 0.02 to 0.08 (even a 0.1 unnecessary interval number corresponds to an average 5% of multi-interval discretizations, since most of these discretizations consist of 3 intervals). The MODL and MDLPC method are very resistant to noise and almost always produce a single interval. The experiments have been performed 100,000 times for these two methods in order to better approximate the result curves in figure 16. The percentage of multi-interval discretization is about 1% of the cases for sample size 100; it decreases with the sample size with an approximate rate of 1/n. This behavior seems consistent since the probability of finding an information pattern in a randomly generated attribute decreases with the sample size. In the few cases of multi-interval discretization, the MDLPC always constructs two intervals, since it is strongly biased by its top-down algorithm in favor of information patterns in the borders of the numerical domain. On the opposite, the MODL method builds discretization containing either two or three intervals, since its bottom-up heuristic allows to identify information patterns surrounded by two noise patterns. This behavior explains why the average unnecessary interval number is slightly larger for the MODL method than for the MDLPC method.

4.4.2 Crenel pattern

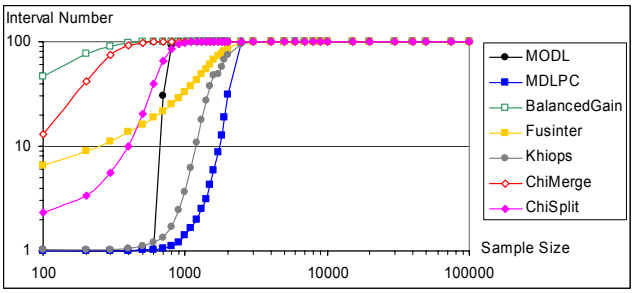

Figure **17**: Mean interval number for the discretization of the "crenel pattern" $Class = Sign(Sinus(100\pi x))$ on the numerical domain [0, 1], in a noise free context

The crenel pattern experiment is based on a more complex dataset family, built with a "crenel pattern". The explanatory attribute is uniformly distributed on the [0, 1] numerical domain and the class attribute consists of two equidistributed class values '+' and '-' which alternate on the explanatory value *x* according to the function $Class = Sign(Sinus(100\pi x))$. In a noise free context, the optimal discretization consists of 100 intervals whose bounds are equidistant on the [0, 1] numerical domain. For small sample sizes, where the number of instances is comparable to the optimal interval number, the distribution of the class values on the numerical domain is similar to noise. In this case, the discretization methods should produce a single interval, related to a test error of 0.5. When the sample size increases, each optimal interval contains more and more instances and can correctly be identified by the discretization methods. Thus, the test error asymptotically decreases towards 0 when the sample size increases. The experiment points out the threshold of the sample size above which all the optimal intervals are correctly detected.

The experiment is performed for a large number of sample sizes ranging from 100 instances to 100,000 instances. The evaluated criterion is the mean of the number of intervals on 1,000 randomly generated samples, for each sample size. We can notice that the 1R discretization method (Holte, 1993) which produces one interval for each sequence of instances belonging to the same class is optimal for this noise-free problem. The results for the tested discretization methods are displayed in figure 17.

The BalancedGain and ChiMerge methods which produce many intervals (cf. UCI experiments) are favored in this noise free context: they are the first methods that correctly identifies the 100 intervals, with about 500 instances. The Fusinter, ChiMerge and ChiSplit methods overfit the data with too many intervals for very small sample sizes. On the opposite, the MODL, Khiops and MDLPC methods produce one single interval for small sample sizes. The transition between 1 interval and 100 intervals happens very sharply at sample size 700 for the MODL method. The MDLPC, Fusinter and Khiops methods need between 2,000 and 3,000 instances to identify all the intervals.

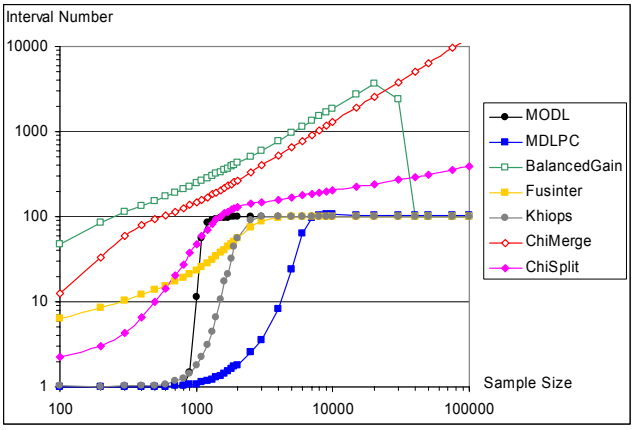

Figure **18**: Mean interval number for the discretization of the "crenel pattern" $Class = Sign(Sinus(100\pi x))$, with 10% misclassified instances

The same experiment is performed in a noisy context, with 10% of randomly misclassified instances. The results are displayed in figure 18. Once again, the BalancedGain, Fusinter, ChiMerge and ChiSplit methods overfit the data with too many intervals, but the Fusinter method does not asymptotically build more than the necessary number of intervals. The BalancedGain method, whose criterion compromises the InformationGain and the number of intervals owing to a ratio, displays an abrupt change in its behavior beyond 30000 instances. The ChiSplit method produces several times the number of necessary intervals when the sample size is large. However, the difference in test accuracy (shown in Figure 19) between the ChiSplit method and the robust methods is asymptotically small. The robust methods MODL, MDLPC and Khiops produce one single interval for small sample sizes and identify the correct intervals for large sample sizes. The transition is still abrupt for the MODL method, at sample size 1,000, whereas it is smoother for the other methods. The correct discretization is found at sample size 3,000 for the Khiops method, 4000 for the Fusinter method and 7,000 for the MDLPC method. When the sample size increases, the MDLPC method slightly overfits the data, with discretizations containing about 103 intervals for sample size 100,000.

To summarize, the MODL method is both robust and sensitive. It produces as few intervals as the most robust methods when the sample size is small, and as many interval as necessary once there is enough instances in the sample.

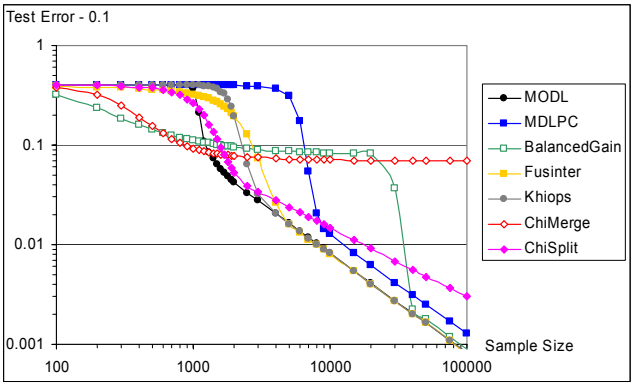

Figure **19**: Mean distance to the true test error for the discretization of the "crenel pattern" $Class = Sign(Sinus(100\pi x))$, with 10% misclassified instances

4.4.3 Peak pattern

The peak pattern experiment focuses on the detection of a pure interval (with instances belonging to one single class) hidden in the center of a noise attribute. In order to prevent superfluous over-splitting, the explanatory attribute consists of only three values: one for the first noisy interval, one for the center pure interval and one for the last noisy interval. The class attribute is composed of two classes. The pure interval contains only instances of the first class and the other intervals share the remaining instances with the same class proportion. The experiment points out the threshold of the peak interval size above which the pattern is correctly identified. Three intervals are necessary when the peak is in the center, and two intervals when the peak is in the head.

In order to get an idea of the frequency of such patterns in randomly distributed attributes, we propose below an approximation of such a threshold. Let p_1 and p_2 be the two class probabilities, n the number of instances in the sample and k the size of the peak interval. The probability that a sequence of k instances contains only instances of the first class is p_1^k. If $k << n$, there are about n such sequences in the sample. Although they intersect on a small scale, we assume that they are independent for approximation reasons. The probability of observing no pure sequence of k instances in the sample is:

$$\left(1 - p_1^k\right)^n.$$

Thus, observing at least one pure sequence of length k in a random sample of size n becomes probable beyond $k \sim -\log(n)/\log(p_1)$.

Figure 20 displays the threshold of the pure interval size for the evaluated discretizations under variation of the sample size, when the two classes are equidistributed. The UnbalancedGain method always produces multi-split discretizations, even when there is one single instance in the peak interval. The ChiMerge and ChiSplit methods are identical when the peak is in the head: they overfit the data. When the peak is in the center, the ChiSplit method suffers from its top-down algorithm and requires an increasing number of instances with the sample

size: it needs about 10 times the theoretical peak size approximation for sample size 10000. The Fusinter method exploits a regularization technique that penalizes intervals with small frequency. The penalty is heuristically adjusted, resulting in an overfitting behavior for small sample sizes and an underfitting behavior for larger sample sizes, wherever the peak is located. The MODL, MDLPC and Khiops methods exhibit a very similar behavior in the case of a head peak, quite close from the theoretical threshold approximation. When the peak is in the center, the MDLPC methods has the same drawback that the ChiSplit method, due to its top-down approach. The Khiops methods has exactly the same behavior for head and center peaks, whereas the MODL methods requires slightly larger peaks in the center, when more complex discretizations (with three intervals instead of two) are necessary.

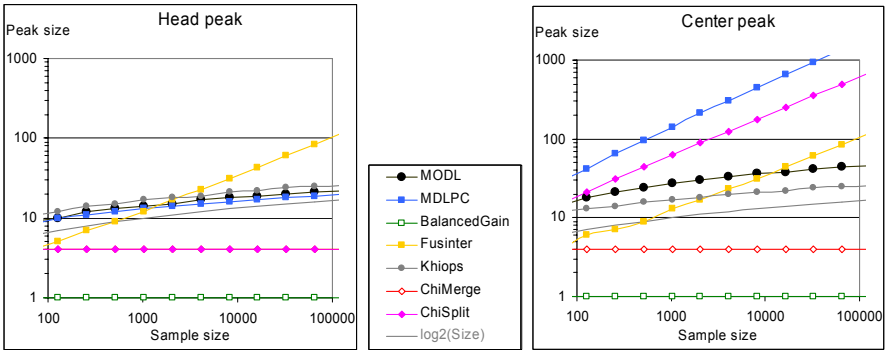

Figure **20**: Threshold for the detection of a pure peak interval hidden into a noise attribute, under variation of the sample size

We exploit the same experiment to study the asymptotic behavior of the methods, when the class distribution becomes more and more unbalanced. The Khiops, ChiMerge and ChiSplit methods are based on chi-square statistics. The chi-square value is not reliable to test the hypothesis of independence if the expected frequency in any cell of the contingency table is less than some minimum value. This is equivalent to a minimum frequency constraint for each row of the contingency table. The tuning of this constraint is a compromise between reliability and fine pattern detection: no optimal trade-off can be found.

Figure 21 displays the threshold of the pure interval size for the evaluated discretizations under variation of the probability of first class, for a sample size fixed to 10000 instances. In addition to the theoretical threshold approximation, we report the chi-square theoretical minimum interval size that corresponds to an expected frequency of at least 1 instance in each cell of the contingency table. The most notable result is a quasi-identical behavior of the MODL and MDLPC methods in the case of head peaks, very close from the theoretical threshold approximation. The chi-square based methods ChiMerge and ChiSplit do no exploit a minimum frequency constraint: their use of the chi-square statistics is not reliable as soon as the first class probability is lower than 0.1. The Khiops method heuristically incorporates the minimum frequency constraint: it is still both too aggressive to get

a reliable use of the chi-square statistics and to conservative to allow fine grain detection when the class distribution is unbalanced. The top-down based algorithms MDLPC and ChiSplit fail to correctly identify pure peaks when they are located in the center. The MODL method obtains the finest results for peak detection wherever they are located. In extremely unbalanced class distributions, the MODL method requires only two instances in head pure peak intervals and fours instances in center pure peak intervals.

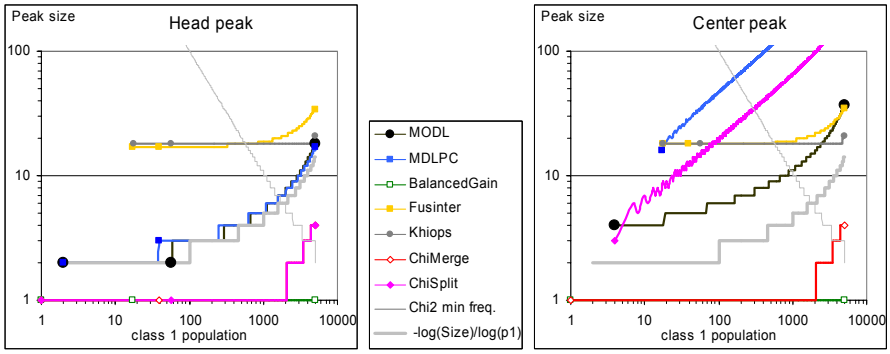

Figure **21**: Threshold for the detection of a pure peak interval hidden into a noise attribute, under variation of the first class population, for sample size 10000

4.4.4 Discussion

The experiments on synthetic data are a way of characterizing each discretization methods and its bias. Some of these methods are very robust to noise, some of them are very accurate in noise free context. It is interesting to see that the widely used accuracy criterion is not suitable to precisely evaluate and compare different inductive methods. It does not penalize the methods which build patterns from noise. Furthermore, the methods which overfit the data have a minor penalty when the detected pattern is close to noise and a large reward when it is a true pattern. Finally, the accuracy criterion does not clearly favor the methods which are more and more accurate as the sample size increases. The difference in accuracy is often small while the difference in correctness of the model might be important. These synthetic experiments are a first step towards a better evaluation of the robustness of the discretization methods and their ability to find complex patterns even in a noisy context. There is still a need for a more complete set of synthetic benchmarks and perhaps for the design of new criteria in order to consistently evaluate the inductive methods on real datasets.

Overall, the synthetic experiments enlighten a large diversity of discriminating behaviors among the evaluated methods. The two MDLPC and Khiops methods are the closest from the MODL method. Compared with the MDLPC method, the MODL method extends the recursive binary-split approach towards a true multi-split discretization schema and largely augments the capacity of fine grain pattern detection. Compared with the Khiops method, the MODL method does not require any parameter for the level of resistance to noise and removes the asymptotical limitations caused by the use of chi-square statistics.

5 Optimization criterion versus search strategy

In this section, we study the relative contribution of the optimization criterion versus the search strategy to the quality of the discretizations. We first focus on computation time and its relation to the numerical efficiency of the optimization. We then examine the connection between the criterion optimality and the discretizations quality.

5.1.1 Computation times of the evaluated algorithm

The evaluated discretization methods have the same computational complexity of $O(n \log(n))$. They first sort the attribute values and identify boundary instances in a preprocessing step. The time efficiency of the methods mainly relies on the search direction (top-down or bottom-up), the simplicity of the mathematical criterion and the use of a post-optimization algorithm. The actual running time on real datasets depends on the final number of intervals found in the attribute discretizations.

The UCI dataset experiments are conducted on a PC with a Pentium IV 2.5 Ghz, using the ten times tenfold cross-validation protocol. Figure 22 reports the average discretization running time per attribute (in seconds) for the ten larger datasets, ranging from 50000 instances on the left to 700 instances on the right. All the supervised discretization methods obtain comparable running times. They are on average between 50% and 150% longer than the unsupervised discretization methods.

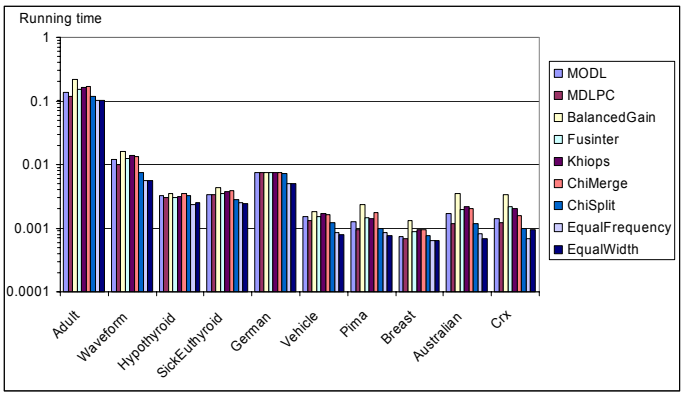

Figure **22**: Computation time of the discretization methods on ten UCI datasets

5.1.2 Computation time versus criterion optimality

The MODL, BalancedGain and Fusinter methods are the only evaluated methods whose criterion can be globally optimized. The other methods MDLPC, ChiMerge and ChiSplit use a recursively applied binary-split criterion. The Khiops method exploits a regularization technique that is intrinsically coupled with its search strategy.

The UCI experiments are completed with the three MODL, BalancedGain and Fusinter methods, using three search strategies: the greedy bottom-up optimization

algorithm, the new evaluated search algorithm that incorporates an additional post-optimization step and the optimal dynamic-programming based algorithm. In order to obtain all the results in less than one week of CPU time, the maximum number of intervals is set to 100 for all the methods and the tenfold cross-validation is performed just one single time. Figure 23 presents the average discretization running times per attribute. The results show a consistent behavior of the algorithms for the three criteria. The post-optimization algorithm presents a 20% running time overhead compared to the greedy algorithm. As expected, the optimal algorithm, whose computational complexity is $O(n^3)$, requires much more running time than the two search heuristics.

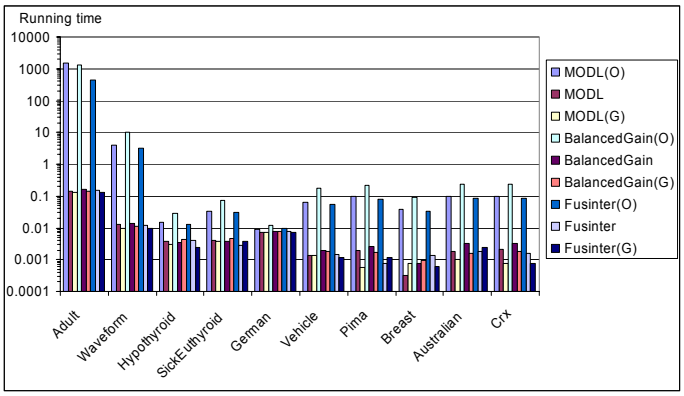

Figure **23**: Computation time of the MODL, BalancedGain and Fusinter discretization methods on ten UCI datasets, using the optimal algorithm (O), the new evaluated algorithm and the greedy algorithm (G)

We report in Table 8 the efficiency of the three search strategies, in terms of their ability to reach the optimal solution or to improve the relative distance to the optimal solution. The "Time" column in Table 8 stands for the geometric mean (by dataset) of the computation time normalized by the Equal Frequency computation time. For example, the MODL method with the post-optimization algorithm takes on average 1.7 more computation time than the Equal Frequency algorithm. The "Diff" column represents the mean (by attribute) of the relative distance of the method criterion to the optimal solution $\left(criterion_{algorithm} - criterion_{optimal}\right) \Big/ criterion_{optimal}$. The "% opt" column presents the percentage of optimal solution found by the algorithm.

The results are still consistent for the three evaluated criteria. The post-optimization algorithm reaches the optimal solution about twice as many times than the greedy algorithm and the relative distance to the optimal solution is improved by a factor 100. From a strict optimization point of view, the post-optimization algorithm radically improves the results of the greedy algorithm with a small computational overhead.

Table **8**: Dataset geometric mean of the normalized computation time (Time), mean of the relative distance to the optimal solution (Diff) and percentage of optimal solution (% opt) for the MODL, BalancedGain and Fusinter discretization methods, using three search strategies

Algorithm	MODL			BalancedGain			Fusinter		
	Time	Diff	% opt	Time	Diff	% opt	Time	Diff	% opt
Optimal	82.3	0.0	100%	161.7	0.0	100%	64.0	0.0	100%
Post-opt	1.7	8.6E-5	95%	2.2	2.4E-3	90%	1.7	8.3E-5	91%
Greedy	1.3	3.8E-3	55%	1.9	1.3E-1	41%	1.4	4.0E-3	40%

5.1.3 Impact on the quality of the discretizations

Reaching the optimal value of a mathematical criterion does not mean finding the best discretization. The notion of best discretization is hard to define: we restrict this to the evaluation of the accuracy, robustness and number of intervals of the discretizations, as reported in Table 9. The optimal and post-optimized search strategies obtain statistically the same results: there are only 5 significant differences (not reported here) between the two search strategies among 1629 individual comparison experiments (181 attributes, 3 evaluations, 3 criteria). Compared to the greedy search strategy, they both bring a slight enhancement in accuracy and robustness, and a more significant reduction in the number of intervals. The Fusinter criterion is the less sensitive one to the quality of the optimization. The BalancedGain criterion highly benefits from the optimization by escaping from local optima with numerous intervals in the case of informative attributes.

Overall, the evaluated quality of the discretizations is improved by better search strategies for all the criteria. However, the criterion is far more important than the search strategy to obtain high quality discretizations.

Table **9**: Dataset geometric means of the accuracy, robustness and number of intervals for the MODL, BalancedGain and Fusinter discretization methods, using three search strategies

Algorithm	MODL			BalancedGain			Fusinter		
	Accur.	Robus.	Int.	Accur.	Robus.	Int.	Accur.	Robus.	Int.
Optimal	69.14	98.17	2.62	66.42	94.83	4.12	69.11	96.43	4.37
Post-opt	69.18	98.23	2.60	66.48	94.84	4.19	69.16	96.57	4.27
Greedy	69.06	98.09	2.80	66.01	90.18	11.04	68.99	96.52	4.29

5.1.4 Impact on the quality of the explanation

In the case of the MODL criterion, we can propose an interpretation of the improvements in the optimized criterion owing to Theorem 5. The absolute difference between the evaluation of a discretization and the optimal evaluation is directly connected to the probability that the discretization explains the data.

Theorem 5: Let M be a MODL discretization of a sample data D, $Cost(M)$ its evaluation with the MODL criterion. Then optimal discretization M_{opt} is a more probable MODL explanation of D than M the with the following factor:

$$\exp\left(Cost(M) - Cost(M_{opt})\right).$$

Proof:

The probability that M explains D is $p(M/D)$. According to the proof of Theorem 1, this probability is related to the MODL evaluation cost by:

$$Cost(M) = -\log\left(p(M/D)\,p(D)\right).$$

Thus,

$$\log\left(\frac{p(M_{opt}/D)}{p(M/D)}\right) = Cost(M) - Cost(M_{opt}).$$

The claim follows. ■

In Figure 24, we report the distance to the optimal discretization for the 1810 discretization performed on the UCI datasets. For example, with the greedy heuristic, 20% of the discretizations are at least 10 times less probable than the optimal discretization to explain the data. The quality of the discretization-based explanation of the data is thus significantly improved by the post-optimization algorithm.

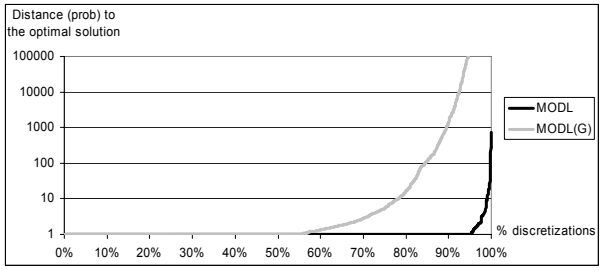

Figure **24**: Repartition function of the distance to the optimal discretization evaluated on 1810 discretizations, for the new evaluated algorithm and the greedy heuristic (G)

We finally compare the two search heuristics in the case of the noise pattern experiment. The optimal algorithm is not applicable due to the size of the datasets and to the number of repetitions of the experiments. Figure 25 shows no differences between the two search strategies for the BalancedGain and Fusinter criteria. On the opposite, the change of behavior is very important for the MODL criterion. The greedy algorithm produces on average 0.2 unnecessary intervals instead of almost exactly the expected number of intervals for the post-optimized algorithm.

This last experiments enlightens the main conclusions of this section. The criterion clearly comes first, before the search strategy. The impact of better optimized discretization is small on common evaluations such as accuracy, robustness or number of intervals. It is still important when the quality of the

explanation of the data is considered.

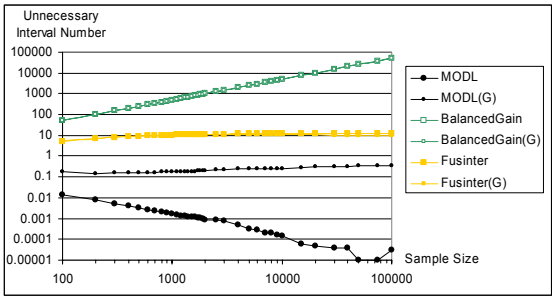

Figure **25**: Mean of the unnecessary interval number of the discretizations of an explanatory attribute independent from the class attribute, for the post-optimized search strategy and the greedy algorithm (G)

6 Conclusion

The MODL discretization method takes advantage of the precise definition of a family of discretization models with a general prior. This provides a new evaluation criterion which is minimal for the Bayes optimal discretization, i.e. the most probable discretization given the data sample. A new optimization heuristic is proposed in this paper to optimize the discretization with super-linear time complexity. This algorithm allows to find the optimal discretization in most cases.

Extensive evaluations both on real and synthetic data enlighten the key features of the MODL method. It is time efficient and does not require any parameter setting. It builds discretizations that are both robust and accurate, resistant to noise and sensitive to fine grain patterns. It correctly identifies complex patterns provided that there is enough data, needing less data than the alternative robust methods. It produces fewer intervals than most alternative accurate methods and has no asymptotic limitation.

The most valuable characteristic of the MODL method potentially resides in the robust explanation of the data it provides. Even though this is restricted to choice of its model space and limited by the bias of its model prior, the MODL method builds the most probable discretization-based explanation of the data.

In future work, we plan to extend this approach to the problem of grouping the values of categorical attributes.

Acknowledgments

I am grateful to the editor Prof. Tom Fawcett and the three anonymous reviewers for their numerous beneficial comments, especially concerning the experimental study and the comparative evaluation of the contribution of the criterion and the search strategy.

References

Bay, S. (2001). Multivariate Discretization for Set Mining, *Knowledge and Information Systems, 3(4)*: 491:512.

Bertier, P. & Bouroche, J.M. (1981). *Analyse des données multidimensionnelles*. Presses Universitaires de France.

Blake, C.L. & Merz, C.J. (1998). UCI Repository of machine learning databases [http://www.ics.uci.edu/~mlearn/MLRepository.html]. Irvine, CA: University of California, Department of Information and Computer Science.

Boullé, M. (2003). Khiops: a Discretization Method of Continuous Attributes with Guaranteed Resistance to Noise. *Proceeding of the Third International Conference on Machine Learning and Data Mining in Pattern Recognition*, 50-64.

Boullé, M. (2004). Khiops: a Statistical Discretization Method of Continuous Attributes. *Machine Learning, 55:1*: 53-69.

Breiman, L., Friedman, J.H., Olshen, R.A. & Stone, C.J. (1984). *Classification and Regression Trees*. California: Wadsworth International.

Catlett, J. (1991). On Changing Continuous Attributes into ordered discrete Attributes. In *Proceedings of the European Working Session on Learning.* Springer-Verlag, 87-102.

Dougherty, J., Kohavi, R. & Sahami, M. (1995). Supervised and Unsupervised Discretization of Continuous Features. In *Proceedings of the 12th International Conference on Machine Learning*. San Francisco, CA: Morgan Kaufmann. 194-202.

Elomaa, T. & Rousu, J. (1996). Finding optimal multi-splits for numerical attributes in decision tree learning. *Technical report, NeuroCOLT Technical Report NC-TR-96-041*, Royal Holloway, University of London.

Elomaa, T. & Rousu, J. (1999). General and Efficient Multisplitting of Numerical Attributes. *Machine Learning, 36*: 201-244.

Fayyad, U. & Irani, K. (1992). On the handling of continuous-valued attributes in decision tree generation. *Machine Learning, 8*: 87-102.

Fischer, W.D. (1958). On grouping for maximum of homogeneity. *Jounal of the American Statistical Association.* 53. 789-798.

Fulton, T., Kasif, S. and Salzberg, S. (1995). Efficient algorithms for finding multi-way splits for decision trees. In *Proc. of the Twelfth International Conference on Machine Learning*.

Holte, R.C. (1993). Very simple classification rules perform well on most commonly used datasets. *Machine Learning, 11*: 63-90.

Kass, G.V. (1980). An exploratory technique for investigating large quantities of categorical data. *Applied Statistics, 29(2)*: 119-127.

Kerber, R. (1991). Chimerge discretization of numeric attributes. In *Proceedings of the 10th International Conference on Artificial Intelligence*, 123-128.

Kohavi, R. & Sahami, M. (1996). Error-Based and Entropy-Based Discretization of Continuous Features. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, AAAI Press/MIT Press, Menlo Park, CA, 114-119.

Kononenko, I., Bratko, I. & Roskar, E. (1984). *Experiments in automatic learning of medical diagnostic rules* (Technical Report). Ljubljana: Joseph Stefan Institute, Faculty of Electrical Engineering and Computer Science.

Lechevallier, Y. (1990). Recherche d'une partition optimale sous contrainte d'ordre total. *Technical report N° 1247*, INRIA.

Liu, H., Hussain, F., Tan, C.L. and Dash, M. (2002). Discretization: An Enabling Technique. *Data Mining and Knowledge Discovery, 6(4)*: 393-423.

Quinlan, J.R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann.

Rissanen, J. (1978). Modeling by shortest data description. *Automatica, 14*: 465-471.

Vitanyi, P.M.B. and Li, M. (2000). Minimum Description Length Induction, Bayesianism, and Kolmogorov Complexity. *IEEE Trans. Inform. Theory, IT-46:2*: 446-464.

Zighed, D.A., Rabaseda, S. & Rakotomalala, R. (1998). Fusinter: a method for discretization of continuous attributes for supervised learning. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 6(33)*: 307-326.

Zighed, D.A., Rabaseda, S., Rakotomalala, R & Feschet F. (1999). Discretization methods in supervised learning. In *Encyclopedia of Computer Science and Technology, 40*, Marcel Dekker Inc., 35-50.

Zighed, D.A. & Rakotomalala, R. (2000), *Graphes d'induction*. HERMES Science Publications, 327-359.

Appendix

Table **10**: Geometric mean of the accuracy per attribute discretization

Dataset	MODL	MDLPC	BGain	Fusin	Khiops	ChiM	ChiS	EqFr	EqWi
Adult	77.34	77.33	74.07	77.27	77.30	75.63	77.32	76.62	76.76
Australian	64.31	64.39	62.40	63.46	64.42	63.86	64.31	65.14	60.72
Breast	85.35	85.55	85.08	85.40	85.35	85.12	85.56	85.24	85.63
Crx	64.41	64.46	62.39	63.41	64.55	63.45	64.49	64.99	60.62
German	70.05	70.00	69.98	70.03	70.04	69.99	70.02	69.90	70.07
Heart	63.57	63.19	63.26	62.68	63.62	62.77	63.00	64.12	63.09
Hepatitis	79.13	79.15	75.98	79.42	79.52	77.92	78.85	80.15	79.99
Hypothyroid	96.00	96.03	95.20	96.04	96.04	96.02	96.02	95.21	95.40
Ionosphere	79.64	77.37	76.21	79.19	79.40	75.73	79.23	73.93	73.39
Iris	76.86	72.91	60.92	77.68	75.98	75.49	76.54	73.42	74.69
Pima	66.10	65.98	64.94	66.01	66.12	65.97	66.08	66.50	66.31
SickEuthyroid	91.29	91.29	91.24	91.28	91.30	91.28	91.31	91.18	90.72
Vehicle	40.21	39.58	35.72	40.53	40.56	40.52	40.82	39.75	39.95
Waveform	48.71	48.77	48.64	48.57	48.75	47.92	48.51	48.79	48.52
Wine	58.97	58.74	55.44	61.21	58.66	58.26	59.15	60.23	59.48
All attributes	66.24	65.67	63.86	66.28	66.25	65.26	66.22	65.35	64.88
All datasets	69.14	68.67	66.26	69.17	69.14	68.38	69.12	68.77	68.04

Table **11**: Number of MODL significant wins and losses for the accuracy criterion

Dataset	MDLPC	BGain	Fusin	Khiops	ChiM	ChiS	EqFr	EqWi
Adult	2/0	5/0	2/0	4/0	3/2	2/2	2/0	3/0
Australian	1/2	2/1	3/1	1/1	2/0	3/2	0/4	5/1
Breast	0/1	2/1	2/2	1/1	2/2	1/2	2/2	1/2
Crx	1/2	2/1	4/1	1/1	4/0	2/2	0/4	5/1
German	1/0	1/0	1/1	0/0	1/0	1/1	2/0	1/2
Heart	1/0	1/0	1/0	0/0	3/0	2/0	2/3	2/1
Hepatitis	0/0	3/1	3/2	0/1	3/3	2/2	0/2	3/3
Hypothyroid	1/2	4/0	1/1	1/2	3/2	2/2	3/0	3/0
Ionosphere	17/2	23/1	14/2	5/5	29/0	16/2	29/0	26/0
Iris	2/0	4/0	0/1	1/1	1/1	1/0	2/0	2/1
Pima	1/1	3/1	2/2	1/1	2/2	1/1	1/2	3/2
SickEuthyroid	0/0	2/0	1/0	0/0	3/2	2/1	1/0	2/0
Vehicle	8/1	14/0	4/5	3/6	4/5	3/7	7/5	6/6
Waveform	2/4	4/9	6/2	1/2	14/0	11/1	3/7	9/5
Wine	4/3	6/2	0/7	5/3	3/1	4/2	1/3	3/5
Total	41/18	76/17	44/27	24/24	77/20	53/27	55/32	74/29

Table **12**: Geometric mean of the robustness per attribute discretization

Dataset	MODL	MDLPC	BGain	Fusin	Khiops	ChiM	ChiS	EqFr	EqWi
Adult	99.99	99.99	90.63	100.0	99.96	94.98	99.91	100.0	100.0
Australian	97.12	97.55	84.95	94.18	97.40	92.34	96.10	98.88	99.00
Breast	99.74	99.80	95.20	98.67	99.46	96.91	99.18	99.67	99.69
Crx	97.35	97.68	84.94	94.20	97.55	91.66	96.39	98.65	98.81
German	99.92	99.89	99.66	99.81	99.91	99.71	99.79	99.84	99.86
Heart	98.78	98.43	92.69	94.22	98.85	93.45	96.22	97.50	97.00
Hepatitis	98.40	99.25	90.26	96.71	98.42	92.70	97.11	99.39	98.23
Hypothyroid	99.91	99.91	99.94	99.89	99.86	99.79	99.84	100.0	99.99
Ionosphere	97.57	97.77	90.52	96.66	97.63	87.36	96.06	98.65	98.87
Iris	96.94	94.55	97.14	97.49	96.76	95.54	96.49	94.92	96.27
Pima	99.14	98.92	92.57	97.23	99.04	95.24	97.55	98.92	98.10
SickEuthyroid	99.96	99.96	99.94	99.95	99.94	99.78	99.93	100.0	99.99
Vehicle	94.87	95.16	96.10	92.40	95.40	90.98	92.64	93.82	94.47
Waveform	98.68	98.76	94.30	96.65	98.68	91.99	96.76	98.12	98.57
Wine	94.29	95.30	96.49	91.17	93.31	86.18	89.65	92.84	93.13
All attributes	98.01	98.14	94.03	96.46	98.00	92.83	96.54	97.93	98.06
All datasets	98.16	98.18	93.57	96.58	98.13	93.82	96.87	98.05	98.11

Table **13**: Number of MODL significant wins and losses for the robustness criterion

Dataset	MDLPC	BGain	Fusin	Khiops	ChiM	ChiS	EqFr	EqWi
Adult	0/0	2/0	0/0	2/0	3/0	2/0	0/0	0/0
Australian	1/2	3/1	5/1	1/1	5/0	4/1	0/4	0/4
Breast	0/0	2/0	2/0	2/0	2/0	2/0	1/0	1/0
Crx	1/2	3/1	5/1	1/1	5/0	4/1	0/4	1/2
German	1/0	3/0	1/0	0/0	3/0	1/0	1/0	1/1
Heart	1/0	2/0	5/0	0/0	4/0	5/0	2/1	3/0
Hepatitis	0/1	3/0	3/1	1/1	4/0	2/0	0/1	3/1
Hypothyroid	1/0	1/0	1/0	1/0	4/0	2/0	0/1	0/1
Ionosphere	1/5	17/3	17/1	2/4	32/0	25/0	0/12	2/14
Iris	2/0	1/2	0/1	1/1	1/2	1/0	1/2	1/2
Pima	1/1	4/1	4/0	1/0	7/0	5/0	1/0	4/0
SickEuthyroid	0/0	1/0	1/0	0/0	4/0	4/0	0/0	1/0
Vehicle	2/4	2/7	7/1	2/5	12/0	10/0	5/3	3/4
Waveform	2/3	4/9	17/0	2/2	21/0	20/0	5/6	2/8
Wine	1/4	1/5	5/0	5/1	11/0	10/0	2/1	4/5
Total	14/22	49/29	73/6	21/16	118/2	97/2	18/35	26/42

Table **14**: Geometric mean of the number of intervals per attribute discretization

Dataset	MODL	MDLPC	BGain	Fusin	Khiops	ChiM	ChiS	EqFr	EqWi
Adult	5.56	5.39	25.05	8.42	6.84	93.47	17.15	4.62	9.29
Australian	2.13	1.95	15.82	6.28	2.12	13.49	4.86	7.74	7.80
Breast	2.58	2.73	3.24	3.80	2.50	5.83	4.51	4.79	8.74
Crx	2.17	1.97	15.99	6.23	2.14	13.39	4.83	7.75	7.80
German	1.18	1.16	2.35	1.94	1.21	1.92	1.80	2.45	3.22
Heart	1.62	1.60	3.75	3.06	1.62	3.38	2.32	4.54	4.79
Hepatitis	1.47	1.35	6.47	3.21	1.51	6.02	2.55	8.08	8.70
Hypothyroid	2.31	2.53	3.77	2.19	3.82	8.52	4.36	6.94	9.60
Ionosphere	4.49	3.65	8.05	5.80	4.13	25.76	7.31	7.73	8.84
Iris	2.99	2.75	2.00	3.10	2.82	3.68	3.60	7.43	9.70
Pima	2.13	1.91	6.67	5.01	2.19	9.50	4.89	8.11	9.43
SickEuthyroid	2.84	2.83	3.06	3.41	3.11	12.79	5.26	6.94	9.59
Vehicle	4.12	3.69	2.20	5.75	3.70	8.48	7.44	8.29	9.57
Waveform	4.37	4.46	4.27	7.83	4.07	48.14	12.20	10.00	10.00
Wine	2.72	2.62	2.01	4.00	2.53	6.15	4.56	9.40	9.83
All attributes	2.79	2.62	4.50	4.40	2.77	10.71	5.18	6.44	7.73
All datasets	2.60	2.48	4.96	4.27	2.68	9.96	4.91	6.63	8.15

Table **15**: Number of MODL significant wins and losses for the number of intervals

Dataset	MDLPC	BGain	Fusin	Khiops	ChiM	ChiS	EqFr	EqWi
Adult	1/1	2/5	4/2	3/3	7/0	7/0	2/5	5/2
Australian	0/3	3/2	6/0	2/1	6/0	6/0	6/0	6/0
Breast	4/0	1/8	7/0	1/5	10/0	10/0	10/0	10/0
Crx	0/4	3/2	6/0	1/2	6/0	6/0	6/0	6/0
German	0/4	20/1	17/0	3/0	15/0	15/0	19/1	24/0
Heart	0/1	3/0	6/0	0/0	6/0	6/0	8/0	8/0
Hepatitis	0/1	4/0	6/0	2/0	6/0	6/0	6/0	6/0
Hypothyroid	4/0	2/3	2/3	6/0	7/0	7/0	7/0	7/0
Ionosphere	1/21	14/11	29/0	0/17	32/0	31/0	32/0	32/0
Iris	1/2	0/4	2/0	0/1	4/0	4/0	4/0	4/0
Pima	2/3	4/2	8/0	3/1	8/0	8/0	8/0	8/0
SickEuthyroid	0/1	1/4	5/0	4/0	6/0	6/0	6/0	7/0
Vehicle	0/12	0/16	17/1	4/9	18/0	18/0	18/0	18/0
Waveform	9/2	4/17	21/0	5/13	21/0	21/0	21/0	21/0
Wine	1/5	0/10	12/0	1/7	13/0	13/0	13/0	13/0
Total	23/60	61/85	148/6	35/59	165/0	164/0	166/6	175/2