IDD: A Supervised Interval Distance-Based Method for Discretization

Francisco J. Ruiz, Cecilio Angulo, and Núria Agell

Abstract—This paper introduces a new method for supervised discretization based on interval distances by using a novel concept of neighborhood in the target's space. The proposed method takes into consideration the order of the class attribute, when this exists, so that it can be used with ordinal discrete classes as well as continuous classes, in the case of regression problems. The method has proved to be very efficient in terms of accuracy and faster than the most commonly supervised discretization methods used in the literature. It is illustrated through several examples, and a comparison with other standard discretization methods is performed for three public data sets by using two different learning tasks: a decision tree algorithm and SVM for regression.

Index Terms—Classification, ordinal regression, supervised discretization, interval distances.

1 Introduction

Discretization, also named quantization, is the process by which a continuous attribute is transformed into a finite number of intervals associated with a discrete value. The importance of discretization methods stems from interest in extending to continuous variable classification methods, such as decision trees or Bayesian networks, which were designed to work on discrete variables. The use of discrete variables, besides diminishing the computational cost of some automatic learning algorithms, also facilitates the interpretation of the obtained results [10], [3].

Discretization can be considered as a previous stage in the global process of inductive learning. In decision trees, discretization as a preprocessing step is preferable to a local discretization process as part of the decision tree building algorithm [4]. This stage can be carried out directly by an expert or automatically by means of a suitable methodology. In any case, the discretization process entails implicit knowledge of the data. This knowledge is introduced explicitly into the learning process by an expert or extracted implicitly from the data as a prior step to the global learning process, if discretization is carried out automatically.

On Qualitative Reasoning methods [19], discretization becomes a mandatory step when data present excess of precision. Excess of precision is, in general, caused by increasing interest in improving measurement processes. Precision may be desirable for some scientific purposes but not for others, since it supposes an excessive amount of information that also requires excessively large memory and calculation capacity. Computer programmers know that if it is not strictly necessary, they should otherwise avoid using double precision variables and resort, whenever possible, to integer variables, or better still, Boolean variables.

The existing methods of discretization can be classified mainly into two categories: unsupervised and supervised [3]. Unsupervised methods do not consider the class to which the training patterns belong. Among these, the most significant are the equal-width and the equal-frequency methods [1], [8], [3]. These methods are very simple to implement with a low computational cost. In addition, it has been pointed out [2] that these types of methods are vulnerable to outliers and the results obtained are rather unsatisfactory in most cases.

On the other hand, supervised methods consider the interdependence between the variable to be discretized and the class to which the patterns belong. Holte [7] presented the simplest example of a discretization method, the 1R algorithm. This method attempts to divide the domain of every continuous variable into pure bins, each containing a large majority of one particular class. Chi-merge [8], Chi2 [11], and StatDisc [15] provide statistical justification of discretization by using the χ^2 test to measure the independence between intervals. Other supervised methods such as the D2 method introduced by Catlett [1] and Minimum Description Length (MDLP) criterion [16] are based on recursive entropy minimization [4]. Finally, some supervised methods of discretization, such as the CAIM method [9], are based on information measures extracted from the quanta contingence matrix.

Most of the existing supervised methods are incremental. This means that they begin with a simple discretization and undergo an iterative process, adding news cutpoints (in top-down methods) or deleting cutpoints (in bottom-up methods). CAIM is an example of a top-down method, whereas Chi-merge is an example of a bottom-up method. Incremental methods need an additional criterion to identify when to stop the process. Unsupervised methods such as equal width or equal frequency are not incremental methods, as all cutpoints are found simultaneously: explaining why these methods are so fast.

- *F.J. Ruiz and C. Angulo are with the Department of Automatic Control, Technical University of Catalonia, EPSEVG-UPC. Avda Victor Balaguer, s/n 08800 Vilanova i la Geltrú, Spain. E-mail: {francisco.javier.ruiz, cecillo.angulo}@upc.edu.*
- *N. Agell is with the Department of Quantitative Methods Management, Universitat Ramon Llull, ESADE. Av. Pedralbes 62-65, 08034 Barcelona, Spain. E-mail: nuria.agell@esade.edu.*

Manuscript received 4 July 2007; revised 17 Dec. 2007; accepted 18 Mar. 2008; published online 4 Apr. 2008.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-2007-07-0328.
Digital Object Identifier no. 10.1109/TKDE.2008.66.

Published by the IEEE Computer Society

The automatic methods of discretization existing in the literature are specifically designed for the problem of Pattern Recognition or Classification, according to which the output variable is a categorical variable whose values belong to a finite set with no order relation. The objective of discretization in such methods is to obtain intervals of the continuous attribute so that patterns with values pertaining to the same interval belong to the same class. This requirement is complemented with a fixed number of intervals and an accepted error level.

There are two important negative aspects to be considered in classic discretization methods. On the one hand, such methods do not consider class proximity, i.e., they penalize patterns assigned to a similar class in the same way as patterns assigned to a very different class. On the other hand, such methods are ineffective when the output variable involves a large number of classes since, in this case, it is necessary to consider a large number of intervals with very few patterns in each.

These points reveal the limitations of the existing discretization methods when the output variable has a large number (or even infinity) of ordered different values. This is a frequent situation, which arises, for example, when the output variable is a qualification: an exam grade, a product satisfaction level, a program audience, a life expectancy after medical treatment, or a financial rating. In these cases, it would be advisable to use the full output variable information including the order identified.

This paper introduces a new method for supervised discretization, Interval Distance-Based Discretization (IDD), which avoids the limitations mentioned above, i.e., it takes into account the order of the output variable and can be used with any number of different output variable values. This new method is based on interval distances and a novel concept of neighborhood. In addition, the methodology proposed is also applicable when there is no order in the class variable, using a suitable distance in the output variable distribution.

In Section 2, the IDD is described in detail. Section 3 is devoted to presenting some interval distances that can be used in the proposed method. In Section 4, some examples are used to illustrate the IDD method, and in Section 5, the method is applied to demonstrate best performance over other methods in two different learning tasks. The final section is reserved for discussion and a summary of this work.

2 Interval Distance-Based Discretization

This section presents the novel method of IDD. This method considers the order of the output variable and can work with ordinal output variables with a large number of different values as well as with continuous variables. The IDD is neither a bottom-up nor a top-down method, but one which, unlike the usual supervised discretization techniques, finds the cutpoints in a single step, dramatically improving computational speed with respect to other techniques. In addition, the number of intervals can either be set previously by the user, as happens in other iterative algorithms, or obtained directly through using the novel concept of Δ-neighborhood.

Δ-neighborhood is the set of Δ nearest neighbors, with respect to the input variable, considered for analyzing whether or not a specific value is a suitable discretization cutpoint. The concept of Δ-neighborhood is key in the IDD method and can be considered as a measure of the quality of the borders. This concept will show that the effectiveness of a border depends on the size of the intervals separated by this border. A specific point may be an effective border for small intervals and ineffective for large intervals, in the same way that a geographic border may be good for separating regions and unsuitable for separating countries. The Δ-neighborhood is a concept that considers the kind of border it is looking for. This concept is highly related to the granularity of discretization. If Δ-neighborhood is large, the granularity will be small (there are fewer borders when looking for countries' borders than when looking for borders of towns or counties).

The IDD has, in addition, a new interesting feature that is different from the existing methods of discretization. In general, discretization forces patterns belonging to the same interval to have the same class. However, the IDD takes into consideration that the distribution of the classes of two contiguous intervals are as different as possible, allowing classes of the same interval to be distributed with a broad deviation. In order to understand the advantage of this characteristic, the discretization of the "*age*" variable can be considered when the output is the "*income*" variable. What makes it possible to distinguish, for example, the age interval (15, 25] from the age interval (26, 37] is not that the first have a low income and the second have a high income, but that most of the first have low income and the second have low, medium, and high incomes: the income deviation in each interval is significantly different.

2.1 Terminology

Let us start by establishing the formal terminology on discretization and some concepts related to the new method. A learning task requires a training data set consisting of N examples, where each example has a set of input variables and an output variable. Given a set of N sample patterns, let us consider a continuous input variable, X its domain and Y the domain of the output. In this method, such as in the majority of the supervised discretization methods, only one input variable is considered, so the training set can be partially represented by the set of couples:

$$\{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\} \subset X \times Y. \tag{1}$$

Definition (discretization). *A discretization D of granularity n of a continuous variable with domain X is a set of disjoint intervals:*

$$D = \{[d_0, d_1],]d_1, d_2], \dots,]d_{n-1}, d_n]\} \tag{2}$$

such that d_0 and d_n are the minimal and maximal values of X, respectively, and the rest of d_i is arranged in ascending order.

The set $\{d_0, d_1, \dots, d_n\}$ is named the set of *landmarks* or *cutpoints* of discretization D. Usually $d_i \in \{x_1, \dots, x_N\} = X$, but if N is small it is better to take

$$d_i \in \{(x_1 + x_2)/2, (x_2 + x_3)/2, \dots, (x_{N-1} + x_N)/2\}.$$

The criterion applied to decide whether d_i is a suitable landmark depends on the discretization method used.

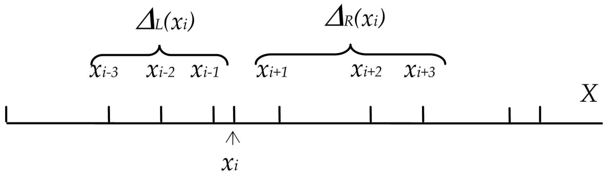


Fig. 1. Illustration of Δ-neighborhood for $\Delta = 3$.

2.2 Neighborhood

The criterion IDD consists of selecting a cutpoint if this locally separates the range of input variable into intervals which are different according to the specific learning problem being considered. When it is judged that a value $d_i \in X$ is a good candidate to be a cutpoint, the adjacent intervals formed by the same number, Δ, of data in both sides of d_i will be considered. This set of values will be named Δ-neighborhood. Most of the discretization methods analyze the suitability of a cutpoint by considering the intervals $]d_{i-1}, d_i]$ and $]d_i, d_{i+1}]$, so they need to consider the cutpoints d_{i-1} and d_{i+1}.

From here on, throughout the rest of this paper, it will be considered that $x_i < x_{i+1}\ \forall i$.

Definition (Set of right and left Δ-neighborhoods). *Let $x_i \in X$, the* right-Δ-neighborhood *and* left-Δ-neighborhood *of x_i are, respectively, the set of Δ values:*

$$\begin{aligned} \Delta_L(x_i) &= \{x_{i-\Delta}, \dots, x_{i-1}\} \subset X, \\ \Delta_R(x_i) &= \{x_{i+1}, \dots, x_{i+\Delta}\} \subset X. \end{aligned} \tag{3}$$

The definition of neighborhood is illustrated in Fig. 1 for the case of $\Delta = 3$. It is important to note that the left-Δ-neighborhood is not defined for the first Δ values $x_1, \dots, x_\Delta$ of the training set and the right-Δ-neighborhood is not defined for the last Δ values $x_{N-\Delta+1}, \dots, x_N$, resulting in these values not being selected as cutpoints. It is necessary to consider this circumstance when choosing the suitable Δ (usually a suitable value will verify $\Delta \ll N$). As mentioned above, the value of Δ is related to the granularity of discretization. A large Δ will obtain few cutpoint numbers, whereas a low value of Δ will obtain more cutpoints.

2.3 Output Sets and Intervals Associated

In order to decide if x_i is a suitable cutpoint, we will look at the output of the Δ-neighborhoods associated to x_i.

Definition (Output sets of right and left Δ-neighborhoods). *Let $x_i \in X$, the output sets of* right-Δ-neighborhood *and* left-Δ-neighborhood *of x_i are, respectively, the sets of Δ values:*

$$\begin{aligned} OS_{\Delta-}(x_i) &= \{y_{i-\Delta}, \dots, y_{i-1}\} \subset Y, \\ OS_{\Delta+}(x_i) &= \{y_{i+1}, \dots, y_{i+\Delta}\} \subset Y. \end{aligned} \tag{4}$$

If the output variable is an ordinal variable or a continuous variable, it is possible to associate an interval (IOS_Δ) to each Δ-neighborhood. The most direct association is the range of Δ-neighborhood outputs.

Definition (Range intervals associated to output sets of right and left Δ-neighborhoods). *Let $x_i \in X$, the range intervals associated to output sets defined above are, respectively,*

$$\begin{aligned} IOS^{range}_{\Delta-}(x_i) &= (\min(OS_{\Delta-}), \max(OS_{\Delta-})) \subset Y, \\ IOS^{range}_{\Delta+}(x_i) &= (\min(OS_{\Delta+}), \max(OS_{\Delta+})) \subset Y. \end{aligned} \tag{5}$$

Other associations are possible, for example, the interquartile ranges (IOS^{interq}_Δ) or intervals centered at the mean and radius equal to the standard deviation ($IOS^{meandesv}_\Delta$). These latter associations will be less sensitive to outliers than the range.

The output sets of the Δ-neighborhoods, or the intervals associated, do not depend on the input variable values but only on its order. In this form, the cutpoints selected remain unchanged if a normalization of this input variable is made.

2.4 The IDD Criterion

The distance between the two intervals associated to the output sets of Δ-neighborhoods will be the criterion to decide if $x_i \in X$ is suitable for being a cutpoint. It is possible to consider several kinds of distances between intervals. Section 3 will present two different distances defined in the set of intervals.

Once a distance measure has been chosen, a function d from $\{x_{\Delta+1}, \dots, x_{N-\Delta-1}\}$ to R (or, better, from $\{\Delta + 1, \dots, N - \Delta - 1\}$ to R) will be obtained:

$$d(i) = distance(IOS_{\Delta-}(x_i), IOS_{\Delta+}(x_i)). \tag{6}$$

The most significant local maxima of function d will be the cutpoint selected. This selection is achieved using a window of size Δ' (being Δ' an integer), which moves from the beginning to the end of the domain of d. The absolute maximum is voted in each window. The most-voted values are the cutpoints selected. The number of cutpoints selected depends on the size of the Δ' windows. It is reasonable for the size of the Δ' windows to be related with Δ. We have taken $\Delta' = \Delta$.

The pseudocode for this algorithm is given as follows:

```
votes = zeros(L)
for i = 1 to L − $\Delta'$ + 1
        index = argmax(d(i : (i + $\Delta'$ − 1)))
        votes(index)++
end for
votes
```

This algorithm returns the vector "*votes*" with a few nonzero elements. Each value is analyzed Δ' times, i.e., the maximum value possible for $votes(k)$ is Δ'. One criterion for selecting cutpoints is to choose the values elected Δ' times exactly (the maximum). This way, the number of cutpoints is indirectly chosen by the values of Δ and Δ'. It is better to consider the vector *votes* weighted by the vector *d*; this way, it is easier to select the cutpoints in order of importance.

3 Interval Distances

To complete the method, it is necessary to define the concept of interval distance. It is not evident what a distance between intervals is. Normally, the distance between two sets of a metric space is defined as the minimum distance between an element of one set and an element of the other set. However, this is not a correct distance, i.e., the properties of a true distance are not satisfied. A suitable distance must satisfy for all x, y:

1. $d(x,y) \geq 0$ and $d(x,x) = 0$,
2. $d(x,y) = d(y,x)$, and
3. $d(x,y) \leq d(x,z) + d(z,y)$.

Let us consider some of the most relevant distances defined on the set of closed and bounded real intervals.

3.1 Hausdorff Distance

By considering closed and bounded real intervals as compact subsets of a metric space, it is possible to employ the Hausdorff distance.

Definition (Hausdorff distance). *Let I_1 and I_2 be two closed real intervals, the Hausdorff distance between I_1 and I_2 is*

$$d_H(I_1, I_2) = \max\{\max_{x \in I_2} d(x, I_1), \max_{x \in I_i} d(x, I_2)\}, \tag{7}$$

where the distance from a point x to an interval I is defined as $d(x, I) = \min_{y \in I}(d(x,y))$.

Therefore, the Hausdorff distance represents the maximum distance of an arbitrary point, belonging to one of the intervals, to the other interval.

It is useful to characterize the Hausdorff distance as a function of the ends of the intervals and as a function of the center and the radius.

Proposition. *If $I_1 = [a_1, b_1]$ and $I_2 = [a_2, b_2]$, therefore $d_H(I_1, I_2) = \max\{|a_2 - a_1|, |b_2 - b_1|\}$.*

Proof. It is sufficient to consider that the distance from a real point x to the interval $I = [a, b]$ is 0 if $x \in I$ and $\min(d(a,x), d(b,x))$ elsewhere and to apply the Hausdorff distance definition to all the relative positions of two intervals I_1 and I_2. □

Proposition. *If c_1 and r_1 are, respectively, the center and the radius of I_1, and c_2 and r_2 are the center and the radius of I_2, respectively, i.e., $c_1 = (a_1 + b_1)/2$, $r_1 = (b_1 - a_1)/2$, $c_2 = (a_2 + b_2)/2$, and $r_2 = (b_2 - a_2)/2$, then*

$$d_H(I_1, I_2) = |c_1 - c_2| + |r_1 - r_2| = |\Delta c| + |\Delta r|,$$

where $\Delta c = c_2 - c_1$ and $\Delta r = r_2 - r_1$.

Proof.

$$\begin{aligned} d_H(I_1, I_2) &= \max\{|a2 - a1|, |b2 - b1|\} \\ &= \max\{|c2 - r2 - c1 + r1|, |c2 + r2 - c1 - r1|\} \\ &= \max\{|\Delta c - \Delta r|, |\Delta c + \Delta r|\} \end{aligned}$$

by using $\max\{\alpha, \beta\} = 1/2 \cdot (\alpha + \beta + |\beta - \alpha|)$

$$\begin{aligned} \max\Big\{|\Delta c - \Delta r|^2, |\Delta c + \Delta r|^2\Big\} &= 1/2 \cdot \big(|\Delta c - \Delta r|^2 + |\Delta c + \Delta r|^2 \\ &\quad + |\,|\Delta c - \Delta r|^2 - |\Delta c + \Delta r|^2|\big) \\ &= \Delta c^2 + \Delta r^2 + 2|\Delta c \cdot \Delta r| \\ &= (|\Delta c| + |\Delta r|)^2, \end{aligned}$$

therefore, $\max\{|\Delta c - \Delta r|, |\Delta c + \Delta r|\} = |\Delta c| + |\Delta r|$. □

The Hausdorff distance can be expressed very easily when using the center and the radius of the intervals. The proposition allows us to consider Hausdorff distance as a Manhattan distance or Minkowski distance (l^p) with $p = 1$.

3.2 Euclidean Distance

By identifying each interval $I = [a, b]$ as a point of the metric space R^2, it is possible to use any distance from R^2. The two matching intervals $\Phi_1(I) = (a, b) \in R^2$ and $\Phi_2(I) = (c, r) = ((b + a)/2, (a - b)/2) \in R^2$ will lead to similar distances.

Definition (Euclidean distance between intervals). *Let $I_1 = [a_1, b_1]$ and $I_2 = [a_2, b_2]$. The euclidean distance between I_1 and I_2 is defined as follows:*

$$d_E(I_1, I_2) = \Big((a_2 - a_1)^2 + (b_2 - b_1)^2\Big)^{1/2}. \tag{8}$$

Proposition. *If c_1 and r_1 are, respectively, the center and the radius of I_1 and c_2 and r_2 are the center and the radius of I_2, respectively, i.e., $c_1 = (a_1 + b_1)/2$, $r_1 = (b_1 - a_1)/2$, $c_2 = (a_2 + b_2)/2$, and $r_2 = (b_2 - a_2)/2$, then*

$$\begin{aligned} d_E(I_1, I_2) &= \sqrt{2} \cdot ((c_2 - c_1)^2 + (r_2 - r_1)^2)^{1/2} \\ &\equiv \sqrt{2} \cdot (\Delta c^2 + \Delta r^2)^{1/2}, \end{aligned}$$

where $\Delta c = c_2 - c_1$ and $\Delta r = r_2 - r_1$.

Both Hausdorff and euclidean distances give the same importance to the two basic features of an interval: the position associated with the center and the precision associated with the radius.

Other distances can be defined based on these definitions [5], for example, using the above function by giving different importance to the center and the radius.

4 Examples of Using the IDD Methodology

In order to illustrate the above concepts, let us consider a couple of synthetic examples: the first with an ordinal discrete output variable and the second with a continuous output. In the first case, it will be possible to compare the IDD method to other standard methods. In the second case, on the contrary, a comparison will not be possible because it is impossible to use such methods with continuous output.

4.1 First Example: Ordinal Discrete Output

A set of 1,000 patterns are generated in the first example. These patterns are characterized by only one input variable and one discrete output variable with five classes. As commented above, the value of the input variable is not important, only the order. For this reason, a sequence of integer numbers (from 1 to 1,000) has been taken as the input variable. The data have been generated using the R expression [13] that is indicated below, where $rnorm(n, \mu, \sigma)$ is the R function that generates n random numbers from a normal distribution with mean μ and standard deviation σ and $floor(x)$ returns the closest integer that is less than or equal to x:

```
data.frame(seq(1000), c(floor(rnorm(300, 3, 0.5),
floor(rnorm(150, 4, 0.2), floor(rnorm(250, 2, 0.6),
floor(rnorm(300, 3, 0.4)))).
```

The distribution of example 1 is represented in Fig. 2.

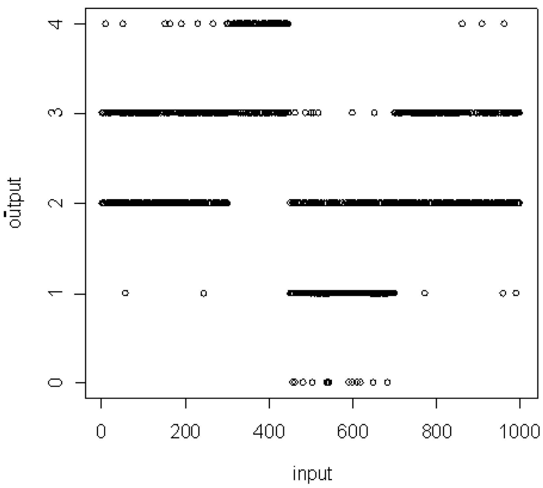


Fig. 2. Distribution of data in example 1.

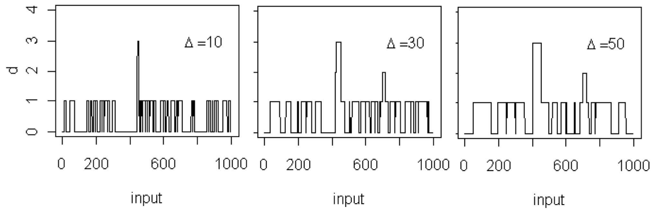


Fig. 3. Comparing function d for several values of Δ (using completed range and Hausdorff distance).

Figs. 3 and 4 represent the "distance" function presented in Section 2 for $\Delta = 10$, 30, and 50 by using the Hausdorff distance (Fig. 3) and the euclidean distance (Fig. 4) and by using the minimum-maximum range as intervals associated to output sets of neighborhoods. It can be seen that the two figures are similar. In this example, no significant differences were found when using the two distances. With regard to the dependence on the value of Δ, when increasing Δ, the number of local maxima of the "distance" function decreases.

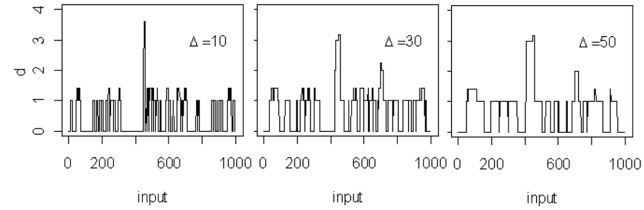


Fig. 4. Comparing function d for several values of Δ (using completed range and euclidean distance).

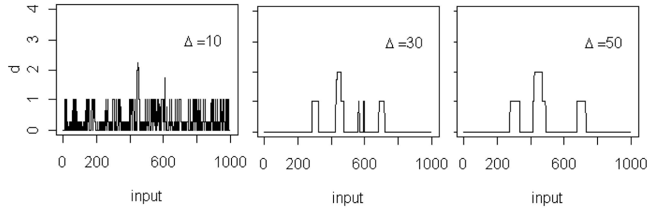


Fig. 5. Comparing function d for several values of Δ (using interquartile range and Hausdorff distance).

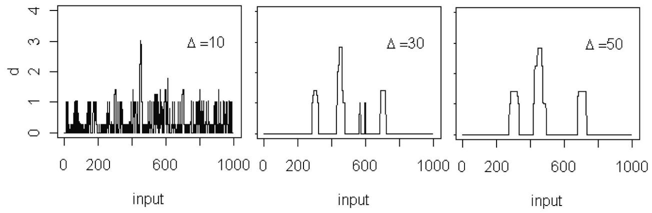


Fig. 6. Comparing function d for several values of Δ (using interquartile range and euclidean distance).

Figs. 5 and 6 represent the "distance" function for $\Delta = 10$, 30, and 50 by using the Hausdorff distance (Fig. 5) and the euclidean distance (Fig. 6) and by using the interquartile range instead of the minimum-maximum range. The differences are appreciated mainly when $\Delta \gg 10$, where the local maxima are more delimited than in the previous case. The interquartile range selects more suitable cutpoints, by ignoring the noise produced by the outliers.

The extraction of the cutpoints from the "distance" function has been done using the voting algorithm presented in Section 2. The results of this voting are represented in Fig. 7a. The *votes* function, weighted by the function *distance* (Fig. 7b), allows us to sort the cutpoints according to their importance.

The next part of this example is the comparison of the results obtained using the IDD method and other standard methods. Concretely, it will compare the IDD method to the bottom-up Chi-merge method [8] and the top-down CAIM method [9]. Table 1 contains a summary of this comparison. The most relevant difference between the IDD method and other methods is that the execution time is of a lower order of magnitude. The Chi-merge method found the expected cutpoints by using a very low value of the σ parameter, much lower than in most normal uses of this algorithm. The CAIM method used in this work is a variant of the original method proposed by Kurgan, where the number of final

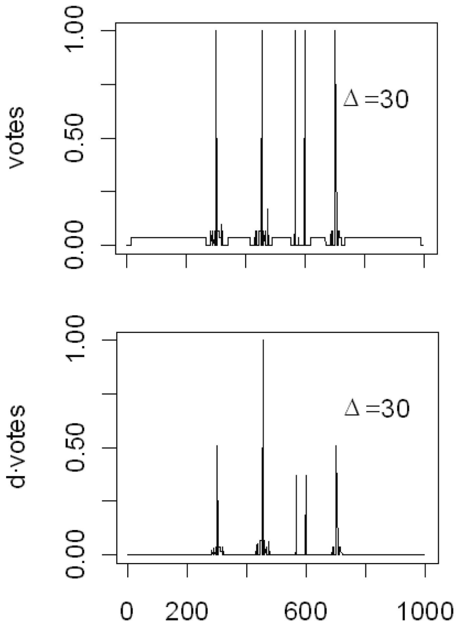


Fig. 7. Comparing the use of bare *votes* and weighted *votes* (using interquartile range, euclidean distance, and $\Delta = 30$).

TABLE 1
Comparison of Methods, Example 1

			Cutpoints	Time
Chi-merge		$\alpha=10^{-1}$	162 166 228 230 265 266 **301** 304 313 377 384 392 398 421 440 445 **450** 501 509 556 564 594 609 611 650 653 **700** 862 864 909 910 942 958 962	240 s
		$\alpha=10^{-3}$	228 230 265 266 **301** **450** **700** 862 864 909 910	240 s
		$\alpha=10^{-5}$	265 266 **301** **450** **700** 909 910	240 s
		$\alpha=10^{-7}$	**301** **450** **700** 909 910.5	240 s
		$\alpha=10^{-9}$	**301** **450** **700**	240 s
CAIM			**240** **451** **594**	13.9 s
IDD	Complete range	$\Delta=10$	**450** 237 **301** 466 501 527 590 646 657 692 **700**	0.08 s
		$\Delta=30$	**448** **700** **301** 582 619 649 229 139 505 756 787 196 901	0.07 s
		$\Delta=50$	**448** **710** 817 910 219 270 **325** 533 651	0.07 s
	Interquartile range	$\Delta=10$	**449** 608 **299** 565 594 **701** 909 403 797 534 545 67 847 16 136 171 182 251 262 314 328 758 837 885 895 920 942 953 972 982 78 502 748 817 57 392 491 515 637 648 670 681 806	4.38 s
		$\Delta=30$	**451** **300** **700** 564 595	4.16 s
		$\Delta=50$	**452** **300** **702**	4.03 s

intervals is fixed a priori. The first three cutpoints found using this algorithm are not the expected cutpoints.

Finally, it can be seen that the best results are found by using the interquartile range and $\Delta = 50$. The slight difference is caused by the way in which the max function works: If there are equal values, max function always selects the ones to the left.

4.2 Second Example: Continuous Output

In the second example, a set of 1,000 patterns is also generated but in this case with a continuous output variable. In this example, it is not possible to compare with standard discretization methods conceived for the classification problem. As in the previous example, a sequence of integer numbers has been used as input variable. The data have been generated using the following expression in R:

$$\texttt{data.frame(X} = \texttt{seq(1000)}, \texttt{Y} = \texttt{c(rnorm(300)} * 6 + 2, \texttt{rnorm(150)} * 3 + 7, \texttt{rnorm(250)} * 2 - 2, \texttt{rnorm(300)} * 4 + 2)).$$

The distribution of these data is represented in Fig. 8.

Figs. 9, 10, 11, and 12 represent the "distance" function for $\Delta = 10$, 30, and 50 using euclidean and Hausdorff distances and minimum-maximum and interquartile ranges. Although the shape of the graph is different from example 1, the same arguments with respect to the influence of Δ and the use of minimum-maximum and interquartile ranges can be considered.

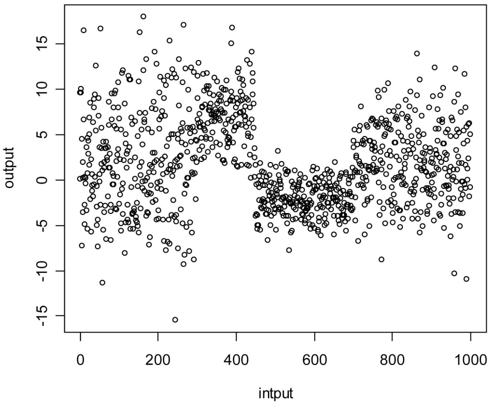


Fig. 8. Distribution of data in example 2.

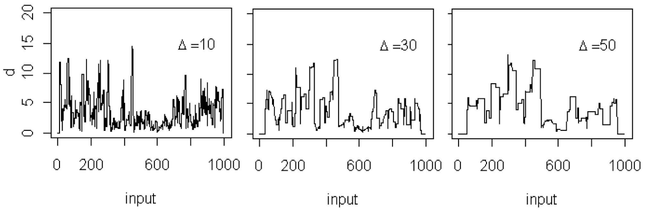


Fig. 9. Comparing function d for several values of Δ (using completed range and Hausdorff distance).

Finally, Fig. 13 represents the comparison between the function votes and the weighted version functions. In the second case, we obtain the cutpoints ordered by importance.

5 Comparing IDD with Other Discretization Methods by Means of Real Experimental Data

In this section, we evaluate the IDD method for three realistic data sets involving continuous inputs and continuous and ordered multiclass outputs by using two different learning tasks: the decision tree algorithm Recursive Partitioning (RPART) [18] and the Support Vector Machine (SVM) for regression [17]. The two tasks have been performed without

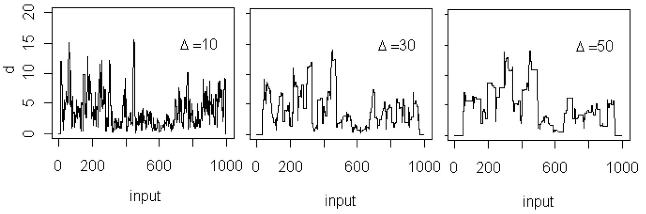


Fig. 10. Comparing function d for several values of Δ (using completed range and euclidean distance).

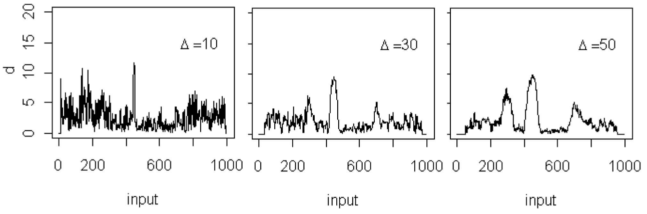


Fig. 11. Comparing function d for several values of Δ (using interquartile range and Hausdorff distance).

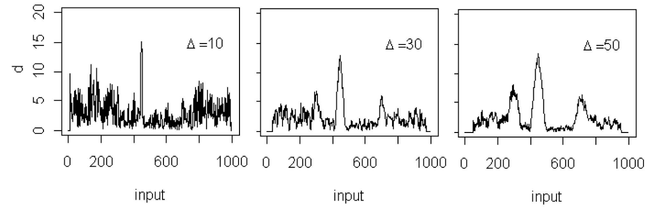


Fig. 12. Comparing function d for several values of Δ (using interquartile range and euclidean distance).

previous discretization and with a discretization preprocess on the continuous attributes using several discretization methods: equal width, Chi-merge, CAIM, and IDD.

The databases are selected from the UCI repository [12] ($auto\_mpg$ and $abalone\_m$) and from the DELVE repository [14] ($kin8nm$).

The database $auto\_mpg$ concerns city-cycle fuel consumption in miles per gallon, to be predicted in terms of three multivalued discrete and four continuous inputs. In the $abalone\_m$ data set, the goal is to predict the age of male abalones based on eight continuous inputs. Finally, the $kin8nm$ represents the forward dynamics of an eight-link all-revolute robot arm. The goal is to predict the distance of the end effector from a target, given the twist angles of the eight links as features. The outputs of $auto\_mpg$ and $kin8nm$ are purely continuous and the output of $abalone\_m$ is ordered multiclass (20 classes).

In order to use standard supervised discretization methods such as Chi-merge and CAIM in purely continuous output databases, an unsupervised discretization of the output variable was performed (using equal width and five classes). These methods cannot be used when output variable has many different values.

The standard errors of regression in the two tasks using all the discretization methods and without discretization are shown in Fig. 14. These errors are calculated by using the following formula:

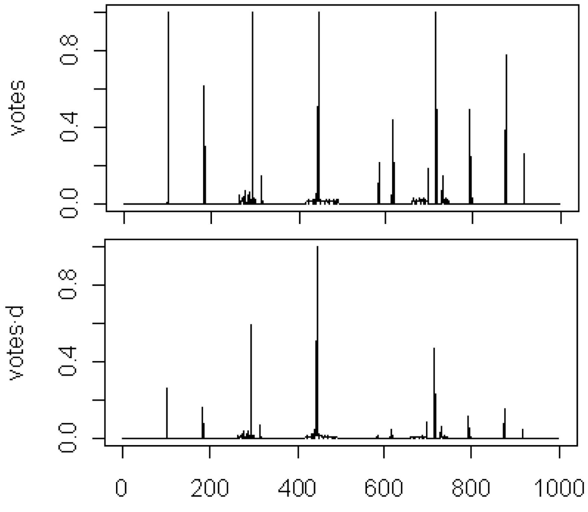
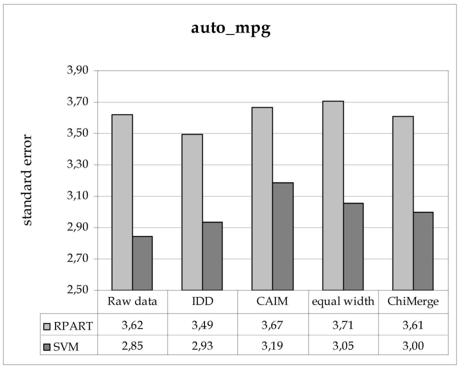


Fig. 13. Comparing the use of vote and $d \cdot vote$ (using interquartile range, euclidean distance, and $\Delta = 30$).



	Raw data	IDD	CAIM	equal width	ChiMerge
▨ RPART	3,62	3,49	3,67	3,71	3,61
■ SVM	2,85	2,93	3,19	3,05	3,00

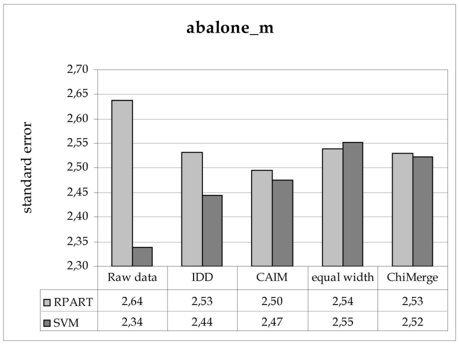
Fig. 14. Standard errors in RPART and SVM tasks in $auto\_mpg$ database using several preprocessed discretization methods (parameters used: IDD: $\Delta = 20$, euclidean distance, interquartile range. Equal width: $\mathrm{granularity} = 5$. Chi-merge: $\alpha = 0.005$).

$$Error = \sqrt{\frac{\sum_i (y_i - y_i')^2}{N}}, \quad (9)$$

where y_i' is the output predicted by the learning task, y_i is the correct output, and N is the number of test patterns. The parameters of discretization methods were taken in order to achieve similar level of granularity (approximately 5) and to facilitate the comparison. More precisely, the values of parameter Δ used were $\Delta = 20$ in $auto\_mpg$ database, $\Delta = 30$ in $abalone\_m$, and $\Delta = 25$ in $kin8nm$. In all cases, the interquartile range and the euclidean distance have been used. The standard error has been estimated using twofold cross-validation and averaging over 30 different training sets.

Starting from the results shown in Figs. 14, 15, and 16, it is evident that SVM for regression obtained better performance than RPART. In this last algorithm, discretization allows one to improve the results obtained with raw data. This is due to the fact that RPART does an implicit discretization in the algorithm that is less efficient than the explicit discretization.

IDD achieves the best performance in SVM task in all databases and good performance in RPART task, similar to the other methods.



	Raw data	IDD	CAIM	equal width	ChiMerge
▨ RPART	2,64	2,53	2,50	2,54	2,53
■ SVM	2,34	2,44	2,47	2,55	2,52

Fig. 15. Standard errors in RPART and SVM tasks in the $abalone\_m$ database using several preprocessed discretization methods (parameters used: IDD: $\Delta = 30$, euclidean distance, interquartile range. Equal width: $\mathrm{granularity} = 5$. *Chi-merge:* $\alpha = 0.08$).

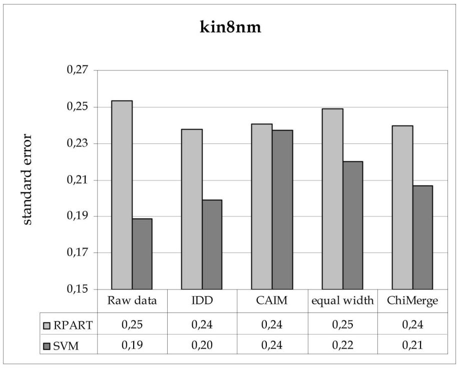


Fig. 16. Standard errors in RPART and SVM tasks in $kin8nm$ database using several preprocessed discretization methods (parameters used: IDD: $\Delta = 25$, euclidean distance, interquartile range. Equal width: $\mathrm{granularity} = 5$. *Chi-merge:* $\alpha = 0.05$).

6 Summary, Conclusions, and Future Work

This work describes IDD, a new discretization method, which, unlike other standard supervised methods of discretization, considers the order of the output variable. It can be applied when the granularity of the output variable is large and even with continuous output variables.

IDD is neither a bottom-up nor a top-down method, but one which, unlike the usual supervised techniques of discretization, finds the cutpoints in a single step, dramatically improving computational speed with respect to other techniques. In addition, the number of obtained intervals can be set previously by the user, as in other iterative algorithms, or obtained directly by using the novel concept of neighborhood, which can consider a measurement of the quality of the borders.

The method is based on the concept of interval distance. Two different distance measures that can be used in the method have been introduced.

The features of the method are presented with two illustrative examples, one with a discrete output variable and the other with a continuous output variable. In the first case, it has been possible to compare with other standard methods but not in the second case since the standard methods are conceived only for classification problems. An experiment with real database and involving two different learning tasks has also been shown. This experiment proves the efficiency of the presented method.

In this work, no significant differences were found in the performance between the two distance measures proposed, Hausdorff and Euclidean. In future works, we will explore the specific effect of each of these distances in the algorithm. In addition, the method will be extended to categorical classes by defining a suitable distance measure in the specific sets of Δ-$\mathrm{neighborhood}$ outputs.

Acknowledgments

This research has been partially granted by the projects AURA (TIN2005-08873-C02) and ADA-EXODUS (DPI2006-15630-C02) of the Spanish Ministry of Education and Science.

References

[1] J. Catlett, “On Changing Continuous Attributes into Ordered Discrete Attributes,” *Proc. European Working Session Learning (EWSL ’91),* Y. Kodratoff ed., pp. 164-178, 1991.

[2] J.Y. Ching, A.K.C. Wong, and K.C.C. Chan, “Class-Dependent Discretization for Inductive Learning from Continuous and Mixed Mode Data,” *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 17, no. 7, pp. 641-651, July 1995.

[3] J. Dougherty, R. Kohavi, and M. Sahami, “Supervised and Unsupervised Discretization of Continuous Features,” *Proc. 12th Int’l Conf. Machine Learning (ICML ’05),* pp. 194-202, 1995.

[4] U.M. Fayyad and K.B. Irani, “Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning,” *Proc. 13th Int’l Joint Conf. Artificial Intelligence (IJCAI ’93),* 1993.

[5] L. González, F. Velasco, J.A. Ortega, C. Angulo, and F.J. Ruiz, “Sobre Núcleos, Distancias y Similitudes Entre Intervalos,” *Inteligencia Artificial. RIIA,* no. 23, pp. 111-117, 2004.

[6] K.M. Ho and P.D. Scott, “Zeta: A Global Method for Discretization of Continuous Variable,” *Proc. Third Int’l Conf. Knowledge Discovery and Data Mining (KDD ’97),* pp. 191-194, 1997.

[7] R.C. Holte, “Very Simple Classification Rules Perform Well on Most Commonly Used Dataset,” *Machine Learning,* vol. 11, pp. 63-91, 1993.

[8] R. Kerber, “Chi-Merge: Discretization of Numeric Attributes,” *Proc. 10th Nat’l Conf. Artificial Intelligence (AAAI ’92),* pp. 123-128, 1992.

[9] L.A. Kurgan and K.J. Cios, “CAIM Discretization Algorithm,” *IEEE Trans. Knowledge and Data Eng.,* vol. 16, no. 2, pp. 145-153, Feb. 2004.

[10] H. Liu, F. Hussain, C. Lim, and M. Dash, “Discretization: An Enabling Technique,” *Data Mining and Knowledge Discovery,* vol. 6, no. 4, pp. 393-423, 2002.

[11] H. Liu and R. Setiono, “Chi2: Feature Selection and Discretization of Numeric Attributes,” *Proc. Seventh IEEE Int’l Conf. Tools with Artificial Intelligence (ICTAI),* 1995.

[12] D.J. Newman, S. Hettich, C.L. Blake, and C.J. Merz, “UCI Repository of Machine Learning Databases,” Dept. of Information and Computer Science, Univ. of California, http://www.ics.uci.edu/~mlearn/MLRepository.html, 1998.

[13] R Development Core Team, “R: A Language and Environment for Statistical Computing,” R Foundation for Statistical Computing, 2006.

[14] C.E. Rasmussen et al., *Delve Databases,* Univ. of Toronto, http://www.cs.toronto.edu/~delve, 2008.

[15] M. Richeldi and M. Rossotto, “Class-Driven Statistical Discretization of Continuous Attributes,” *Proc. Eighth European Conf. Machine Learning (ECML ’95),* pp. 335-338, 1995.

[16] J. Rissanen, “Modeling by Shortest Data Description,” *Automatica,* vol. 14, pp. 445-471, 1978.

[17] A. Smola and B. Sch, “A Tutorial on Support Vector Regression,” Technical Report NeuroCOLT NC.TR-98-030, Royal Holloway College, Univ. of London, 1998.

[18] T. Therneau and E. Atkinson, “An Introduction to Recursive Partitioning Using the RPART Routine,” technical report, Section of Biostatistics, Mayo Clinic, Rochester, http://www.mayo.edu/hsr/techrpt/61.pdf, 2008.

[19] L. Travé-Massuyès, *Le Raisonnement Qualitatif pour les Sciences de l’Ingénieur. Hermès,* 1997.

Francisco J. Ruiz received the MSc degree in physics from the University of Barcelona in 1988 and the PhD degree from the Technical University of Catalonia in 2006. He is currently an assistant professor in the Automatic Control Department, Technical University of Catalonia. His research interests are in qualitative reasoning and Kernel methods with applications in control systems and finances. He is a member of the Knowledge Engineering Research Group (GREC).

Cecilio Angulo received the MSc degree in mathematics from the Universitat de Barcelona in 1993 and the PhD degree in science from the Technical University of Catalonia in 2001. He is a lecturer in the Automatic Control Department, Universitat Politècnica de Catalunya. His research interests include machine learning, intelligent control, and ambient intelligence. He is a member of the Knowledge Engineering Research Group (GREC).

Núria Agell received the PhD degree in applied mathematics from the Technical University of Catalonia. She is a professor in the Quantitative Methods Department, ESADE, University Ramon Llull. Her main research activities are currently related to the 1) development of soft-computing models and technologies based on qualitative and fuzzy reasoning and 2) application of artificial intelligence techniques to finances, marketing, and knowledge management. She is a member of the Knowledge Engineering Research Group (GREC).