Proceedings of the Tenth International Symposium on Computer and Information Sciences, pp. 443-450, 1995

An Empirical Comparison of Discretization Methods

Dan Ventura and Tony R. Martinez

Computer Science Department, Brigham Young University, Provo, Utah 84602
e-mail: dan@axon.cs.byu.edu, martinez@cs.byu.edu

Many machine learning and neurally inspired algorithms are limited, at least in their pure form, to working with nominal data. However, for many real-world problems, some provision must be made to support processing of continuously valued data. This paper presents empirical results obtained by using six different discretization methods as preprocessors to three different supervised learners on several real-world problems. No discretization technique clearly outperforms the others. Also, discretization as a preprocessing step is in many cases found to be inferior to direct handling of continuously valued data. These results suggest that machine learning algorithms should be designed to directly handle continuously valued data rather than relying on preprocessing or *ad hoc* techniques.

1. Introduction

Many machine learning and neurally inspired algorithms are limited, at least in their pure form, to working with nominal data, where nominal data is defined as data from a small, finite, unordered domain. Examples include CN2 [7], ID3 [16], and AQ* [12][13]. However, for many real-world problems, some provision must be made to support processing of continuously valued data. One approach is discretization of continuously valued data as a preprocessing step. Discretization is the process of converting continuously valued data into nominal data by assigning ranges of real values to one nominal value. Discretizing continuously valued data produces inherent generalization by grouping data into several ranges, representing it in a more general way. Further, discretization has some psychological plausibility since in many cases humans apparently perform a similar preprocessing step representing naturally continuous data such as temperature, weather, and speed as nominal values (i.e. hot, cold, fair, cloudy, fast, slow, etc.). The main effort of this paper is to investigate the feasibility of discretizing continuously valued data as a preprocessing step for supervised learning systems. To do so, six different discretization methods are compared as preprocessors to three different supervised learning models. These comparisons are made over several real-world data sets and the results provide evidence that suggests that *machine learning algorithms should directly incorporate handling of continuously valued data rather than relying on a discretization preprocessing step*.

Section two briefly describes the discretization methods and supervised learning systems used. Section three presents the empirical results of the study and sections four and five analyze these results and draw conclusions.

2. Description of Discretization Methods and Supervised Learners Used

The discretization methods used are ChiMerge [10], equal-width-intervals[21], equal-frequency-intervals [21], maxi-min into k-means [8], Valley [19][20], and Slice [20]. This is by no means an exhaustive list of possible discretization techniques (see also [6], [9], [11], and [18]), but it is believed to be a representative sample.

ChiMerge. ChiMerge is based on the statistical χ^2 test. All examples are sorted and initially each is placed in its own interval. Then intervals are merged iteratively in the following manner. The χ^2 value is computed for each pair of adjacent intervals and the pair of intervals with the lowest χ^2 value is merged. This process is repeated until all pairs of adjacent intervals have a χ^2 value that exceeds some threshold. Two adjacent intervals whose χ^2 value exceeds this threshold are considered to be significantly different and thus should not be merged.

Equal-width-intervals and equal-frequency-intervals. Equal-width-intervals is a generic method that simply divides the data into some number of intervals all with equal width. This is completely arbitrary and does not seek to discover any information inherent in the data. Its main advantage is its simplicity. Equal-frequency-intervals uses a similar approach but employs a frequency metric, dividing the data into some number of intervals each of which contains the same number of data points. The same comments apply to this method as to the equal-width method.

Maxi-min into k-means (kmmm). This is a combination of two classical clustering algorithms, k-means and maxi-min. K-means depends on user input for the number of desired intervals but does not depend on the user to find interval boundaries. Maxi-min discovers the "proper" number of intervals but depends on user-defined parameters to find interval boundaries. In order to reduce (*not* eliminate) dependence on user-defined parameters, maxi-min is used to discover the "proper" number of intervals and then k-means is used to find the actual interval boundaries.

Valley. Valley is specific instantiation of a general discretization paradigm called BRACE. This method concentrates on finding the natural boundaries between intervals and creates a set of possible classifications using these boundaries. All classifications in the set are evaluated according to a criterion function and the classification that maximizes the criterion function is selected. Valley creates a histogram of the data, finds all local minima (valleys), and ranks them according to size. The largest is then used to divide the data into a two-interval classification. A three-interval classification is then created using the two largest valleys and so on until a *v*-interval classification has been created (where *v* is the number of local minima in the histogram). These classifications are then used to predict the output class of the data, and the classification with the best prediction rate is selected.

Slice. Slice is a generalization of Valley in which the boundary between every range (slice) in the histogram is a potential interval boundary. The best place to divide the data into two intervals is found by iteratively trying each range boundary and using the resulting classification to attempt to predict the data's output class. This is continued in a greedy fashion to divide the data into three intervals, and so on until prediction of the output class fails to improve.

The three supervised learners used are ID3 [16], CN2 [7], and Bounded Order Rule Sets [3][4]. They were chosen because they all require a discretizing preprocessor and represent different approaches to supervised learning. ID3 and CN2 are well established machine learning approaches, while BORS is a recently proposed technique.

ID3. ID3 is a decision tree approach to learning. A decision tree is built using information gain as the metric for splitting nodes. This is a greedy approach that attempts to maximize information gain at each node by splitting on the attribute that currently provides the most information gain. The final hypothesis is represented by a decision tree. C4.5 [15] is an extension of ID3 that, among its improvements, allows handling of continuously valued data by basically testing all possible splits over the range of the attribute to be discretized. This discretization is performed independently at each node in the tree. Therefore, discretizing the data before giving it to ID3 is similar to giving the undiscretized data to C4.5. This is convenient in the sense that we can compare results obtained by discretization and ID3 directly with results obtained from C4.5 which performs its own internal handling of continuously valued data (see Table 6 and accompanying discussion).

CN2. CN2 is a rule induction algorithm that incorporates ideas from both ID3 and from AQ* [12][13]. Thus CN2 attempts an eclectic combination of the efficiency and ability to cope with noisy data that ID3 exhibits, with the flexible search strategy and if-then rule format of AQ*. The algorithm attempts to build good if-then rules by considering different

combinations of variables and assessing how well each rule properly classifies the training instances. The search space is exponential and, therefore various heuristics are used to limit search time and space. The hypothesis is represented as an ordered list of if-then rules.

Bounded Order Rule Sets (BORS). This is an experimental approach that assumes that most interesting attribute correlations (features) are of a relatively low order. Thus, even though examining all higher order correlations is an exponentially expensive proposition in terms of both time and space, BORS proposes that most *interesting* correlations or features may be examined by brute force in bounded time. Therefore, the algorithm first considers all first order features and evaluates their ability to correctly classify the training set. It then considers all second order features and so on up to some relatively small *kth* order. Some subset of the features are then chosen to represent the hypothesis.

3. Empirical Results

To empirically compare the six discretization methods, each was used as a preprocessor to each of the three supervised learners for several different real-world data sets. For example, the glass data set was discretized with ChiMerge, and then fed to ID3, CN2 and BORS. It was then discretized with kmmm and again fed to all three learners, and so on. Then the whole process was repeated for each of the other data sets. The data sets used were all obtained from the UC Irvine Machine Learning Database [14], and each data set consists entirely of continuously valued inputs and has a single nominal output variable:

Glass. 9 inputs, 7 output classes. This is a collection of data from crime lab reports. The nine inputs are measures of mineral content (Mg, Al, etc.) and refractive index of different samples of glass. The problem is to determine what the glass was used for--windows, container, head lamp, etc.

Wine. 13 inputs, 3 output classes. These data are the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivators. Analysis determined the quantities of 13 constituents found in each of the three types of wines and these constitute the 13 input variables. The problem is to determine which of the three cultivators produced the wine.

Vowel. 10 inputs, 11 output classes. Speech signals from 15 different speakers were low pass filtered at 4.7kHz and then digitized to 12 bits with a 10kHz sampling rate. Twelfth order linear predictive analysis was carried out on six 512 sample Hamming windowed segments from the steady part of the vowel. The reflection coefficients were used to calculate 10 log area parameters, giving a 10 dimensional input space. Based on the ten inputs, the problem is to determine which vowel sound was spoken.

Sonar. 60 inputs, 2 output classes. Each of the sixty inputs represents a measure of the energy within a particular frequency band. Given these inputs, determine whether the sonar image is a rock or a mine.

Iris. 4 inputs, 3 output classes. Given sepal length and width and petal length and width, determine which type of iris the flower is.

Bupa. 6 inputs, 2 output classes. The six inputs represent the outcome of five blood test results and number of drinks (half-pint equivalents of alcoholic beverages consumed per day) for a group of male Bupa Indians. The problem, given the input data, is to determine whether or not the individual is likely to contract a liver disorder.

Pima. 8 inputs, 2 output classes. The inputs include age, number of times pregnant, and the results of various medical tests and are taken from a population of female Pima Indians. Given these, determine if the individual tests positive for diabetes.

Vehicle. 18 inputs, 4 output classes. The inputs represent silhouette features extracted by the HIPS (Hierarchical Image Processing System) extension BINATTS at the Turing Institute. Given these inputs the problem is to determine which of four vehicles the

silhouette represents.

Tables 1-3 summarize the results obtained by using each of the discretization methods as a preprocessor for each of the supervised learners. Results are averages obtained over ten runs by using 70% of the data as a training set for learning and the remaining 30% as a test set. The results reported in the tables are percent accuracy on the test set.

	Glass	Wine	Vowel	Sonar	Iris	Bupa	Pima	Vehicle	Average
Chi	61	92	61	74	95	62	73	67	73
Freq	53	88	62	65	93	52	67	62	68
Kmmm	64	89	62	68	93	58	73	61	71
Slice	59	93	36	78	95	61	73	62	70
Valley	65	81	47	65	96	60	74	66	69
Width	58	92	64	77	95	51	70	62	71
Chi	3	2	4	3	2	1	2	1	2.3
Freq	6	5	2	5	4	5	6	3	4.5
Kmmm	2	4	2	4	4	4	2	6	3.5
Slice	4	1	6	1	2	2	2	3	2.6
Valley	1	6	5	5	1	3	1	2	3.0
Width	5	2	1	2	2	6	5	3	3.3

Table 1. Performance and ranking of discretization methods as a preprocessor to ID3

In each of the tables the data sets are represented in columns and the discretization methods are represented in rows. The tables are split into two sections. The top matrix of each gives the percent accuracy of the supervised learner on a given data set using a particular discretization method. For example, Table 1 shows that ID3 achieved an accuracy of 61% on the glass data set with ChiMerge used as the preprocessor. Table 2 shows an

	Glass	Wine	Vowel	Sonar	Iris	Bupa	Pima	Vehicle	Average
Chi	64	75	48	78	95	63	71	45	67
Freq	51	83	57	79	64	57	69	53	64
Kmmm	64	88	57	78	93	56	72	59	71
Slice	59	72	33	80	96	62	74	55	66
Valley	53	75	35	74	92	61	72	43	63
Width	49	85	58	75	92	54	71	57	68
Chi	1	4	4	3	2	1	4	5	3.0
Freq	5	3	2	2	6	4	6	4	4.0
Kmmm	1	1	2	3	3	5	2	1	2.3
Slice	3	6	6	1	1	2	1	3	2.9
Valley	4	4	5	6	4	3	2	6	4.0
Width	6	2	1	5	4	6	4	2	3.8

Table 2. Performance and ranking of discretization methods as a preprocessor to CN2

accuracy of 64% for CN2 for the same data and discretization method, and Table 3 shows an accuracy of 62% for Bounded Order Rule Sets.

The bottom matrix of each table shows the relative rank of each discretization method for each data set. For example, Table 1 shows that for ID3's learning of the vowel data set,

ChiMerge was the 4th best discretization method. Table 2 shows that for CN2, ChiMerge was again the 4th best choice, but Table 3 shows that for Bounded Order Rule Sets, ChiMerge was the best discretization method of the six.

Examining the data in all three tables yields some interesting results. First, a perusal of the rankings for each discretization method shows no clearly best method. Although overall ChiMerge appears to be a little more robust than the others, the best method of discretization appears to be data dependent. Also, the best discretization method depends on the supervised learner for which the preprocessing is being performed. To see this, notice that ChiMerge does quite well for Bounded Order Rule Sets but not nearly so well for CN2. K-means combined with maxi-min (kmmm) seems to be the most robust choice for CN2.

	Glass	Wine	Vowel	Sonar	Iris	Bupa	Pima	Vehicle	Average
Chi	62	71	81	90	94	68	72	72	76
Freq	46	69	4	78	93	63	64	65	60
Kmmm	48	68	3	73	89	58	69	60	59
Slice	52	71	23	86	95	49	72	72	65
Valley	57	67	67	85	94	59	68	63	70
Width	39	65	8	78	94	53	70	65	59
Chi	1	1	1	1	2	1	1	1	1.1
Freq	5	3	5	4	5	2	6	3	4.1
Kmmm	4	4	6	6	6	4	4	6	5.0
Slice	3	1	3	2	1	6	1	1	2.3
Valley	2	5	2	3	2	3	5	5	3.4
Width	6	6	4	4	2	5	3	3	4.1

Table 3. Performance and ranking of discretization methods as a preprocessor to BORS

Table 4 summarizes the best, worst, and average accuracies obtained on the data sets by seven well-known learning algorithms that directly handle continuously valued data: C4.5 (both decision tree and rule list forms) [15], Cart [5], Bayes [5], Minimum Message Length [17], IB3 [1], and IB4 [2]. For details on the methods of obtaining these data see [22].

Dataset	Best	Average	Worst
Glass	68.6	66.0	61.6
Wine	93.8	92.7	90.5
Vowel	92.6	82.1	73.3
Sonar	78.9	75.1	72.6
Iris	95.3	94.8	94.0
Bupa	67.0	62.2	56.5
Pima	73.4	71.0	68.3
Vehicle	77.6	69.5	59.2

Table 4. Best, worst, and average performance for seven well-known learning algorithms that directly support handling of continuously valued data [22]

A second result of interest is obtained by comparing accuracy results from Tables 1-3 with the results in Table 4. Although each supervised learner performs well on some of the data sets (at least assuming discretization done by the best method), each one also performs quite poorly on others even using the discretization method with the best results.

Table 5 supports this claim by comparing the three supervised learners' best results with the average and best results of the seven algorithms mentioned above. AvgC is the average percent correct for all seven algorithms and MaxC is the maximum percent correct for any of them (these values are taken directly from Table 4). Bold table entries indicate relatively poor performance -- ID3 does comparatively poorly on the glass, vowel, and vehicle data sets; CN2 does comparatively poorly on the glass, wine, vowel, and vehicle data sets; and Bounded Order Rule Sets performs comparatively poorly on the glass and wine data sets.

	Glass	Wine	Vowel	Sonar	Iris	Bupa	Pima	Vehicle
ID3	**65**	93	**64**	78	96	62	74	**67**
CN2	**64**	**88**	**58**	80	96	63	74	**59**
BORS	**62**	**71**	81	90	95	68	72	72
AvgC	66	93	82	75	95	62	71	70
MaxC	69	94	93	79	95	67	73	78

Table 5. Discretization preprocessing vs. direct handling of continuously valued data

The claim that discretization as preprocessing is usually detrimental to performance is further substantiated by comparing the performance results of ID3 (and the best preprocessor for each data set) with C4.5 (see Table 6. For a discussion of why C4.5 was used for comparison, see the discussion of ID3 in section 2). As may be seen from the table, for all data sets except sonar, discretization as a preprocessing step did not significantly improve performance. Again, for four data sets -- glass, vowel, bupa, and vehicle -- the discretization was actually detrimental.

Dataset	ID3	C4.5
Glass	65	68
Wine	93	93
Vowel	64	80
Sonar	78	75
Iris	96	95
Bupa	62	65
Pima	74	73
Vehicle	67	70

Table 6. ID3 with (best) discretization preprocessing vs. C4.5

It should be noted that the three algorithms examined here -- ID3, CN2, and BORS -- were designed to explore the idea of concept learning. Concept learning in general does not address the issue of continuously valued data and often deals exclusively with nominal data. Therefore, the lower performance results for these algorithms on data sets comprised entirely of continuously valued attributes is not surprising. This observation further strengthens the position that consideration of continuously valued data should not be ignored in design of a learning algorithm.

These two results together, namely that 1) the best discretization method is dependent both on the application as well as on the supervised learner and that 2) in general, discretization as a preprocessing step can lead to severe performance degradation, provide evidence that discretization as a preprocessing step independent of the supervised learning system is not the appropriate approach to machine learning.

4. Problems with Discretization as a Preprocessing Step

There are at least two inherent problems in using discretization as a preprocessing step for a supervised learner. One of these problems results in the best discretization method being dependent on the application and supervised learner used. The other results in the sometimes poor overall performance observed.

The Independence Problem. This has to do with the fact that a general method of discretization has no knowledge of what information the supervised learning method requires to do its learning. Therefore, the discretizer may unwittingly eliminate some valuable (to the particular supervised learner in question) information in the preprocessing step thereby hampering the supervised system's ability to learn. On the other hand, the same discretization for another supervised learner may have essentially avoided eliminating important information for that particular case.

The Higher Order Correlation Problem. Obviously, there are often higher order correlations between input variables. That is, one input variable by itself may not directly correlate with the output class, but in combination with one or more of the other input variables, it may have a very high correlation with the output class. The idea behind discretizing the data as a preprocessing step has been to leave the discovery of these higher order correlations to the supervised learner. Unfortunately, in considering and discretizing each of the input variables independently, the discretizer may destroy the higher order correlation. For example, consider iris flowers. Perhaps sepal lengths fall into two general categories: long (those over 2 inches) and short (those 2 inches or less); however, perhaps if the sepal width is greater than .2 inches, then a sepal length is not considered long unless it is over 3 inches. Since the discretizer is not allowed to discover this higher order correlation, it will place the input variable sepal length into the "long" interval if it is over 2 inches, whether that instance's sepal width is greater than .2 or not. It will therefore incorrectly discretize the sepal length variable of any instance whose sepal length is between 2 and 3 inches long and whose sepal width is greater than .2 inches.

5. Conclusions

The empirical results of this paper indicate that in general, the choice of which discretization method to use depends both on the problem to be learned as well as on the choice of supervised learning algorithm. As a result, none of the discretization methods clearly out-performed the others, although ChiMerge appears to be somewhat more robust than the other methods. Also, the process of discretization as a preprocessing step suffered significantly when compared with algorithms that directly handle continuously valued data. Each of the supervised learners that employed preprocessing performed poorly on several of the problems, even assuming *a posteriori* choice of the best discretization method for the combination of problem and learner. These results as well as the two problems discussed in section 4 provide evidence that discretization as a separate preprocessing step is not the proper approach to machine learning. Instead, the handling of both nominal and continuously valued data should be addressed in the design of a learning algorithm.

This work was funded in part by a grant from Novell, Inc.

6. References

[1] Aha, D.W.; Kibler, D.; and Albert M.K, "Instance-Based Learning Algorithms", *Machine Learning*, vol. 6, Kluwer Academic Publishers, The Netherlands, pp.37-66, 1991.

[2] Aha, D. W., "Incremental, instance-based learning of independent and graded concept descriptions", *Proceedings of the Sixth International Workshop on Machine Learning*, pp. 387-391, 1989.

[3] Andersen, Timothy L. and Martinez, Tony R., "Learning and Generalization with Bounded Order Critical Feature Sets", *Proceedings of the 6th Australian Joint Conference on Artificial Intelligence*, p.

450, 1993.

[4] Andersen, Timothy L., “Learning and Generalization with Bounded Order Rule Sets”, Masters Thesis, Brigham Young University, April, 1995.

[5] Breiman, Leo; Friedman, Jorome; Olshen, Richard; and Stone, Charles, Classification and Regression Trees, Chapman and Hall, New York, 1993.

[6] Catlett, J., "On Changing Continuous Attributes Into Ordered Discrete Attributes", *Lecture Notes in Artificial Intelligence*, Ed. J. Siekmann, Springer-Verlag, Berlin, pp. 164-78, 1991.

[7] Clark, Peter and Niblett, Tim, "The CN2 Induction Algorithm", *Machine Learning*, vol. 3, Kluwer Academic Publishers, The Netherlands, pp. 261-83, 1989.

[8] Everitt, Brian, Cluster Analysis, Halsted Press, New York, 1974.

[9] Fayyad, Usama M. and Irani, Keki B., “On the Handling of Continuous-Valued Attributes in Decision Tree Generation”, *Machine Learning*, vol. 8, pp. 87-102, 1992.

[10] Kerber, Randy, "ChiMerge: Discretization of Numeric Attributes", *Proceedings of the 10th National Conference on Artificial Intelligence*, pp. 123-7, 1992.

[11] Lebowitz, Michael, “Categorizing Numeric Information for Generalization”, *Cognitive Science*, vol. 9 no. 3, pp. 285-308, 1985.

[12] Michalski, Ryszard S., "A Theory and Methodology of Inductive Learning", Readings in Machine Learning, eds. Jude W. Shavlik and Thomas G. Dietterich, Morgan Kaufman Publishers, Inc., San Mateo, California, pp. 70-95, 1990.

[13] Michalski, Ryszard S. and Stepp, Robert, E., "Learning From Observation: Conceptual Clustering", Machine Learning: An Artificial Intelligence Approach, eds. Ryszard S. Michalski, Jaime G. Carbonell, Tom M. Mitchell, Tioga Publishing Company, Palo Alto, California, pp. 331-63, 1983.

[14] Murphy, P. M. and Aha, D. W., *UCI Repository of machine learning databases*. Irvine, CA: University of California, Department of Information and Computer Science, 1992.

[15] Quinlan, J. Ross, C4.5: Programs For Machine Learning, Morgan Kaufman Publishers, San Mateo, California, 1993.

[16] Quinlan, J. R., "Induction of Decision Trees", Readings in Machine Learning, eds. Jude W. Shavlik and Thomas G. Dietterich, Morgan Kaufman Publishers, Inc., San Mateo, California, pp. 57-69, 1990.

[17] Rissanen, Jorma, “A Universal Prior for Integers An Estimation by Minimum Description Length”, *Annal of Statistics*, vol. 11 no. 2, pp. 416-31, 1983.

[18] Schlimmer, Jeffrey C., “Learning and Representation”, *Proceedings of the Sixth National Conference on Artificial Intelligence (AAAI87)*, v2, pp.511-515, 1987.

[19] Ventura, Dan and Martinez, T. R., "BRACE: A Paradigm For the Discretization of Analog Data", *Proceedings of the Seventh Florida Artificial Intelligence Research Symposium*, pp. 117-21, 1994.

[20] Ventura, Dan, “On Discretization as a Preprocessing Step for Supervised Learning Models”, Masters Thesis, Brigham Young University, April, 1995.

[21] Wong, A. K. C., and Chiu, D. K. Y., “Synthesizing statistical knowledge from incomplete mixed-mode data”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-9 no. 6, pp.796-805, November 1987.

[22] Zarndt, Frederick, "An Empirically Derived Test Suite for Machine Learning and Connectionist Algorithms", in preparation, 1994.