Discretization of Continuous Attributes in Supervised Learning algorithms

Ali Al-Ibrahim

Faculty of Information Technology
Department of Computer Information and Network Systems
The World Islamic Sciences and Education University (WISE)
+962 5600230, P.O. Box 1101, Amman 11947, Jordan
ali.alibrahim@wise.edu.jo
alikitim@yahoo.com

Abstract

We propose a new algorithm, called CILA, for discretization of continuous attribute. The CILA algorithm can be used with any class labeled data. The tests performed using the CILA algorithm show that it generates discretization schemes with almost always the highest dependence between the class labels and the discrete intervals, and always with significantly lower number of intervals, when compared with other state-of-the-art discretization algorithms. The use of the CILA algorithm as a preprocessing step for a machine learning algorithm significantly improves the results in terms of the accuracy, which are better than by using other discretization algorithms.

Keywords: Discretization methods, Machine learning, Continuous Inductive Learning, unsupervised learning methods, and supervised learning methods.

1 Introduction

The process of automation of processing and extraction of knowledge from data becomes an important task that is often performed by machine learning (ML) algorithms. One of the most common tasks performed by ML algorithms is generation of classification rules from class-labeled examples. The examples are described by a set of numerical, nominal, or continuous attributes. Many existing inductive ML algorithms are designed expressly for handling numerical or nominal data, while some algorithms perform better with discrete-valued attributes despite the fact that they can also handle continuous attributes [1][9].

This drawback can be overcome by using a discretization algorithm as a front-end for the learning algorithm. Discretization is a process of transforming a continuous attribute values into a finite number of intervals and associating with each interval a discrete, numerical value. The usual approach for learning tasks that use mixed-mode (continuous and discrete) data is to perform discretization prior to the learning process [1][7][8][12].

The discretization process first finds the number of discrete intervals, and then the width, or the boundaries for the intervals, given the range of values of a continuous attribute. Most often the user must specify the number of intervals, or provide some heuristic rule to be used [2]. The proposed CILA algorithm performs both tasks by automatically selecting the number of discrete intervals and finding width of every interval based on interdependency between class and attribute values.

Discretization algorithms can be divided into two categories:
1. Unsupervised (class-blind) algorithms that discretize attributes without taking into account respective class labels. The representative algorithms are equal-width and equal-frequency discretizations [3].
2. Supervised algorithms discretize attributes by taking into account the class-attribute interdependence. The representative algorithms are: maximum entropy [17],Patterson-Niblett algorithm [11], which is built-in as a front end into a decision trees algorithm [13], and other information-gain or entropy-based algorithms [7],[19], statistics based algorithms like Chi Merge [9] or Chi2 [10], class-attribute interdependency algorithms like CADD algorithm [2] and clustering-based algorithms like K-means Discretization algorithm [15].

Discretization should significantly reduce the number of possible values of the continuous attribute since large number of possible attribute values contributes to slow and ineffective process of inductive machine learning [1]. Thus, a supervised discretization algorithm should seek possibly minimum number of discrete intervals, and at the same time it should not weaken the interdependency between the attribute values and the class label. The proposed CILA discretization algorithm not only discretizes an attribute into the small number of intervals but also makes it easier for the subsequent machine learning task by maximizing the class-attribute interdependency. In this paper we discuss machine learning, especially inductive learning and its related algorithms, and a new information theoretic discretization method optimized for supervised and unsupervised learning methods is proposed and described, so we proposed improvement of ILA [14] which is called Continuous Inductive Learning Algorithm (CILA), and tested in inductive learning example to show how the discretization methods deal with continuous attributes.

Inductive learning system can be effectively used to acquire classification knowledge from examples; many existing symbolic learning algorithms can be applied in domains with continuous attributes when integrated with discretization algorithms to transform the continuous attributes into ordered discretization.

Most of the existing machines learning algorithms are able to extract knowledge from databases that store discrete attributes (features). If the attributes are continuous, the algorithms can be integrated with a discretization algorithm that transforms them into discrete attributes.

The attribute in learning problem may be nominal (categorical) , or they may be continuous (numerical) . The term “continuous” is used in the literature to indicate both real and integer valued attributes . Continuous-valued attribute must , therefore, be discretized prior to attribute selection. There are many ways to discretize a continuous attribute , so to partition a continuous variable two important decisions must be made [2]:

1- The number of discrete intervals must be selected, the selection of the optimal number of itervals is seldom addressed by existing discretization methods, and in most cases the human user selects the appropriate number of intervals.

2- The width of the intervale must be determined ,which mean the boundaries of the discretized intervals need to be defined.

The simplest discretization procedure is to divide the range of a continuous variable into equal-width intervals, which determined the range of values from the minimum and maximum observed attribute values, the range is then divided equally by a user-defined number of intervals, but the weekness of this procedure is that in case where the outcome observations are not distributed evenly, a large amount of important information can be lost after the discretization process, to solve this problem a method based on the concept of maximum marginal entropy has been developed, this method partitions a continuous attributes using a criterion to maximize shannon’s entropy and this minimize the loss of information. Once the number of interval is selected.

A variation on the entropy scheme determines the interval boundaries by making the total gain of information from the observed occurrences un each interval equal called the Even *Information Interval Quantization method*, this method determines the optimal interval boundaries by equalizung the total information gain in each interval, this method has been used with an Inductive learning (IL) system to decompose and classify continuous my electric signals.

2 Discretization process

Discretization is viewed as the partitioning of a continuous-valued attribute into an ordered discrete attribute with a number of discrete intervals, which equivalent to the process of reducing the number of states of an ordered discrete random variable by combining some of its states together.

In general discretization is s process that transforms the range of the continuous attributes into a discrets partion consisting of number of intervals associated with boundary set and quanta set [1],[3],[8],[12].
Smaller numbers of intervals are always preferred in inductive learning application because a large number of intervals means larger number of possible attribute values, and that contributes to slow and inefficient learning process.

The general discretization methods [2],[18]:

1- Class-Attribute Dependent Discretizer (CADD) .
2- Maximum Entropy (ME).
3- Equal Information Gain (EIG).
4- Equal Interval Width (EIW).

The CIL algorithm uses the class-attribute dependency information as the criterion for the optimal discretization, which has the minimum number of discrete intervals and minimum loss of the class-attribute interdependency. After Ching, Wong & Chan [2] we introduce several basic definitions. For a certain classification task, let us assume that we have a training data set consisting of M examples, and that each example belongs to only one of the S classes. F will indicate any of the continuous attributes from the mixed-mode data. Then there exists a discretization scheme D on F, which discretizes the continuous domain of attribute F into n discrete intervals bounded by the pairs of numbers:

D:{[d*0*,d*1*],[d*1*,d*2*],….,[d*n-1*, d*n*]}.
where d_0 is the minimal value and d_n is the maximal value of attribute *F*, and the values are arranged in the ascending order. These values constitute the boundary set {$d_0, d_1, d_2, \dots, d_{n-1}, d_n$} for discretization *D*.

In D each value belonging to attribute F can be classified into only one of the n intervals. With the change of discretization D, the membership of each value in a certain interval for attribute F may also change. The class variable and the discretization variable of attribute F can be treated as two random variables, thus a two-dimensional frequency matrix (called quanta matrix) can be set up as shown in Table 1.

In Table 1, qir is the total number of continuous values belonging to the ith class that are within interval (dr-1, dr]. Mi+ is the total number of objects belonging to the ith class, and M+r is the total number of continuous values of attribute F that are within the interval (dr-1, dr], for i=1,2…,S and, r= 1,2, …, n.

Table 1. 2-D frequency matrix for attribute F and discretization scheme *D* Interval

	$[d_0, d_1]$ …	$(d_{r-1}, d_r]$ …		$(d_{n-1}, d_n]$	Class Total
C_1	q11 …..	q1r	………..	q1n	m1+
:	: …..	:	……..	:	:
C_i	qi1 …..	qir	……..	qin	mi+
:	: …..	:	…..	:	:
C_S	qs1 …..	qsr	……	qSn	ms+
Interval Total	M_{+1}	M_{+r}	…	M_{+n}	M

Based on the quanta matrix the Class-Attribute Interdependence Redundancy (CAIR) criterion [20] has been desgnied. The CAIR has been used as a discretization criterion in the class-attribute dependent discretizer (CADD) algorithm [2]. In the nutshell, the CAIR criterion reflects the interdependence between classes and the discretized attribute, being at the same time independent of the number of class labels and the number of unique values of the continuous attribute. The larger the value of the CAIR the better correlated are the class labels and the discrete intervals. For details on the CAIR criterion the reader is referred to [6]. The CADD algorithm has several problems. It uses userspecified number of intervals and the maximum entropy discretization method to initialize the intervals, which may cause the algorithm to remain in the worst starting point in terms of the CAIR criterion. Finally, experience is required for selection of a confidence interval for the significance test used in the algorithm. The CIL algorithm has no disadvantages associated with the CADD algorithm.

3 Continuous Valued attributes

The initial definition of ID3 assumes discrete valued attributes , but continuous values attributes can be incorporated in the tree.

On continuous attributes:

1. Sort the data set on the real-valued attribute.
2. Dynamically create partitions using different thresholds.
3. The approach used here is to fix a threshold value when the output class changes.
4. Take mid-point of the two real values as the thresholds.
5. We have two partitions now, in any of which the example lie.
6. Get all possible thresholds and their gains or gain ratios.
7. Select as threshold the one with best information gain or gain ration.
8. One performing the tests , threshold calculated by considering gain gave more or less the same results as gain ratio.
9. Consider the best thresholds and split the training examples.
10. Only one partition is made and , so effectively attribute can take two values . Less than the thresholds or greater than or equal to threshold.
11. Compare the best gain with other gain of other attributes to get best attribute.

4 Continuous inductive learning Algorithm [CILA].

Figure 1 shows continuous inductive learning algorithm (CILA). Please note that we discuss algorithms that deal with continuous data.

CILA algorithm.

- Read the list of classes initially .
- Each class is given a class number and associated with it's name.
- All classes are sorted in a list.
- Read information about attributes next .
- Attributes could be DISCRETE or CONTINUOUS , flag them accordingly .
- If attribute is DISCRETE also get the possible values it could take.
- All possible values for DISCRETE attributes are stored in a list .
- Each attribute is flagged , given an attribute number , list of values initialized all stored in an attribute structure .
- Read each example incrementally.
- For each attribute of the example validate it and mark it DISCRETE , CONTINUOUS or UNKNOWN.
- Store all the examples in a table , where each row stands for a training example .
- The basic algorithm is same as ID3.

```
FUNCTION ID3( R: a set of non-goal attribute,
              C: the goal attribute ,
              S: a training set ) RETURN a decision tree,
    BEGIN
        IF  S is empty,
            Return  a single node with value failure;
        IF R is empty,
Return a single node with as value the most frequent of the vales of the goal attribute that are found in records if S;[note that then there will be errors,that is ,records that will be improperly  calssified;
LET D be the attribute with largest gain(D,S) aming attribute in R;
LET { dj | j=1,2,…..,m} be the value of atrribute D;
LET{ sj | j=1,2,…..,m} be the subsets of S consisting repectively of records with value dj for atrribute D;
RETURN a tree with rool labeled D and arcs labeled d1,d2,….,dm going respective to the trees
ID3(R-{D},C,S1),ID3(R-{D},C,S2),….,ID3(R-{D},C,Sm;
END ID3;
```

- Use other helper functions to calculate gain , best attribute , thresholds for continuous attributes , keep track of missing attributes , assign probabilities to missing values.
- Run the BUILDTREE routine until all examples are classified or all attributes in the examples are used up in that particular path.

Figure 1. CILA algorithm.

5 ID3 Algorithm

5-1 The idal area of the ID3 algorithm

ID3 is an algorithm introduced by Quinlan [13] for inducing Classification Models, also call Decision Trees, from data . We are given a set of records. Each record has the same structure , consisting of a number of attribute/value pairs. One of these attributes represents the goal of the record , i.e. the attribute whose values are most significant to us . The problem is to determine a decision tree that on basis of answers to questions about the non-goal attributes predicts correctly the value of the goal attribute [12].

ID3 is designed to handle training data with discrete and symbolic attribute values . In order to deal with continuous-valued attribute , ID3 must treat them as discrete attribute with many possible values, continuous-valued attribute can significantly by the arcs.

For example (in Table 2) which includes the manufacturer , state , city , product color , and the profitability is shown in the column Profit and has three distinct values : High , Average , Low.

So the categorical attribute specifies high , low and avg profit products.

Table 2. The non categorical attributes are :

ATTRIBUTE	POSSIBLE VALUES
Manufacturer	Adams,Johnson,Smith
State	AZ,CA,NY
City	LosAngeles,Flagstaff,NYC
Product Color	Blue , Green , Red

And the training data is:

Manufacturer	State	City	Product Color	Profit
Smith	CA	Los Angeles	Blue	High
Smith	AZ	Flagstaff	Green	Low
Adams	NY	NYC	Blue	High
Adams	AZ	Flagstaff	Red	Low
Johnson	NY	NYC	Green	Avg
Johnson	CA	Los Angeles	Red	*Avg

A decision tree is important not because it summarizes what we know , i.e. the training set , but because we hope it will classify correctly new cases . Thus when building classification models one should have both training data to build the model and test data to verify how well it actually works .

Definitions :

If we are given a probability distribution P=(p1,p2,……,pn) the Information conveyed by this distribution, also called the Entropy of P, is :

T(P) = -(p1*log(p1) | p2*log(p2) |…….. pn*log(pn))

If a set T of records is partitioned into disjoint exhaustive classis
C1,C2,…….,Ck on the basis of the value of the goal attribute, then the information needed to identify the class of an element of T is info(T) = I(P) , where P is the probability distribution of the partition(C1,C2,……Ck):

P = (|C1| / |T| , |C2| / |T| ,………., |Ck| / |T|)

In our example:
P1 = P(high) = 2/6
P1 = P(low) = 2/6
P1 = P(avg) = 2/6
P = (2/6 , 2/6 , 2/6)
I(p) = - ((0.33 * log2(0.33)) | ((0.33 * log2(0.33)) | ((0.33 * log2(0.33)))
= - ((0.33 * -1.6)) | ((0.33 * -1.6)) | ((0.33 * -1.6))
= 1.584
Info(T) = I(p)-1.584

If we first partition T on the basis of the value of a non-goal attribute X into sets T1,T2,……..,Tn then the information needed to identify the class of an element of T becomes the weighted average of the information needed to identify the class of an element of Ti, i.e. the weighted average of info(Ti):

Info(X,T)=∑|Ti|/|T| * info(Ti)

In our example , for the attribut state we have:

Info(State,T) = 2/6*I(1/2,1/2) | 2/6*I(2/2,0) | 2/6*I(1/2,1/2)
= 0.333 + 0 + 0.333
= 0.666

consider the quantity Gain (X,T) defined as :

Gain (X,T) = info(T) – info (X,T)

This represents the difference between the information needed to identify an element of T and the information needed to identify an element of T after the value of attribute X has been obtained , that is this is the gain in information due to attribute X. In our example , for the State attribute the gain is :

Gain (State , T) = Info(T) - Info(State , T)
= 1.584 – 0.666
= 0.918

We can use this function Of Gain to rank attribute and to build decision trees where at each node is located the attribute with greatest gain among the attributes not yet considered in the path from the root.

5-2 The ID3 Algorithm Code

The ID3 algorithm is used to build a decision tree , given a set of Non-goal attribute C1,C2,………Cn, The goal attribute c , and a training set T of records . As seen in Figure 2.

ID3 Algorithm.

```
FUNCTION ID3( R: a set of non-goal attribute,
              C: the goal attribute ,
              S: a training set ) RETURN a decision tree,
    BEGIN
        IF  S is empty,
              Return  a single node with value failure;
        IF R is empty,
Return a single node with as value the most frequent of the vales of the goal attribute that are found in records if S;[note that then there will be errors,that is ,records that will be improperly  calssified;
LET D be the attribute with largest gain(D,S) aming attribute in R;
LET { dj | j=1,2,…..,m} be the value of atrribute D;
LET{ sj | j=1,2,…..,m} be the subsets of S consisting repectively of records with value dj for atrribute D;
RETURN a tree with rool labeled D and arcs labeled d1,d2,….,dm going respective to the trees
ID3(R-{D},C,S1),ID3(R-{D},C,S2),….,ID3(R-{D},C,Sm;
END ID3;
```

Figure 2. ID3 algorithm.

In the our example we obtain the following decision tree (Figure 3):

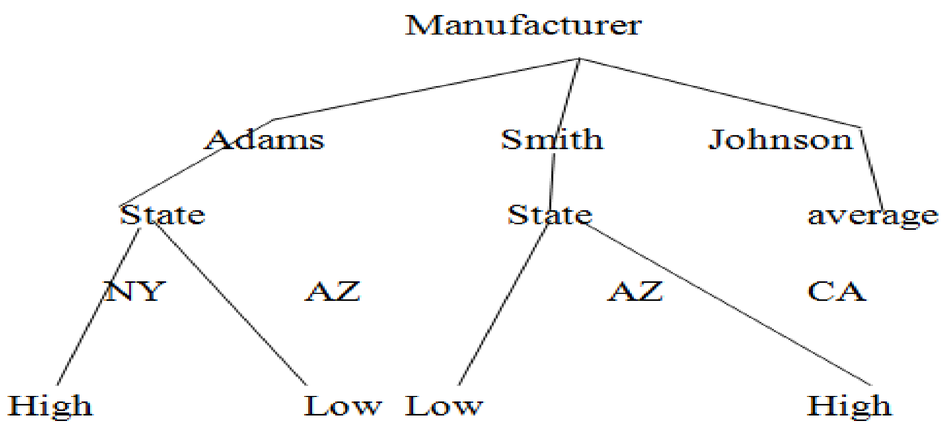

Figure -3 Decision tree

6 Experiments

To demonstrate how the propose discretization algorithm works in a real-word situation, we use a set of mixed-mode data from the microcomputer fault diagnosis domain [2]. Asset of 110 problem situation have been reclassified into one of six common PC power-up faults. Each problem record is characterized by 13 attribute, some of them continuous to represent the symptoms. The objective is to identify which subsystem is at fault causing the appearance of the problem symptoms.

One of the continuous –valued is the measured voltage between pin 2 and pin 4 from the power supply to the major peripherals. Those attributes provide important information since certain range of voltage measures can indicate power related problem as opposed to other types of problem, a knowledge-based approach was used to proper voltage range at pin 2 and pin 4 on a typical PC compatible computer is between 4.8 and 5.2 volts. A reasonable "common sense " partition scheme would be to divide the attribute into three ranges: too low (<4.8), normal (4.8-5.2), too high (>5.2).

In many application domains, knowledge about the attribute is not always easily available, and our proposed method is capable of utilizing the inherent class versus attribute range information to define the set of partition boundaries which emphasize the relation between the values assignment and the attribute range.

Suppose we wish to acquire diagnostic rules for classifying PC hardware fault using an IL algorithm, in this example, 109 samples are for training, and a single example is selected for testing. The teat cases are not generally available prior to classification training, and they will not be considered in the discretization analysis, instead test cases will be discretized prior to class predication procedure according to discretiaztion interval derived from the training sample.

There are only 85 training sample values available from this attribute due to missing values. Since there are six possible classes, it is determined that the maximum default number of intervals is 85/ (3*6) which is equal to 4. Therefore, the proposed algorithm begin with the default number of intervals 4, and proceeds to partition the attribute into 4 as evenly distributed intervals as possible in order to maximize H(AJ). This frequency matrix associated with the initial partition is presented in table 3. The initial mutual information between the class labels and the attribute intervals is 0.161820, and the calculated interdependence redundancy is 0.055825. after a few local boundary, the class-attribute mutual information is increased to 0.239157 , while the related interdependence redundancy measure is improved to 0.095790. the corresponding boundary set and the frequency distribution matrix are shown in table 4, note that the marginal frequency distribution for the classes dose not change.

Upon further test of interdependence on these two adjacent intervals the algorithm discovers that they may be combined without reducing the degree of class-attribute interdependence significantly. Therefore the original default number of intervals is now reduced to three and after an additional local boundary perturbation pass, the algorithm produce the final optimized boundary set and the corresponding frequency distribution matrix in table 5. Recall that human experts knew that voltages between 4.8 volts 5.2 volts are considered acceptable, and the proposed algorithm produced "normal" range of 4.8 to 5.3 volts is needed very similar...

So it is interesting that the final partition resulted in an optimal interdependence redundancy measure of 0.110098. As expected the absolute mutual information actually decreased slightly to 0.232157, as a result of reducing the number of intervals from 4 to 3. This result illustrates the importance of using normalize interdependence redundancy rather than the absolute mutual information as a discretization criterion.

Table 3. initial intervals and associated frequency matrix

		Attribute value intervals			
class	**0.0 - 4.7**	**4.8 - 4.9**	**5.0 - 5.0**	**5.1 - 7.1**	**Total**
1	7	**2**	**1**	**5**	**15**
2	2	6	6	5	19
3	**0**	**7**	**4**	**6**	**17**
4	**0**	**5**	**3**	**7**	**15**
5	**0**	**4**	**3**	**4**	**11**
6	**0**	**3**	**2**	**3**	**8**
total	**9**	**27**	**19**	**30**	**85**

Table 4. improved intervals and associated frequency matrix

		Attribute value intervals			
Class	**0.0 - 4.7**	**4.8 - 4.8**	**4.9 - 5.3**	**5.4 - 7.1**	**total**
1	7	**1**	**4**	**3**	**15**
2	2	2	12	3	19
3	**0**	**3**	**14**	**0**	**17**
4	**0**	**2**	**13**	**0**	**15**
5	**0**	**3**	**8**	**0**	**11**
6	**0**	**2**	**6**	**0**	**8**
Total	**9**	**13**	**57**	**6**	**85**

Table 5. final optimal intervals and associated frequency matrix

	Attribute	**Value**	**intervals**	
Class	**0.0 - 4.7**	**4.8 - 5.3**	**5.4 – 7.1**	**Total**
1	7	**5**	**3**	**15**
2	2	14	3	19
3	**0**	**17**	**0**	**17**
4	**0**	**15**	**0**	**15**
5	**0**	**11**	**0**	**11**

6	**0**	**8**	**0**	**8**
Total	**9**	**70**	**6**	**85**

After executing this example we will have the following rules:

Rule1:if voltage is too low and the class number is 1 then training sample value is 7
Rule2:if voltage is too low and the class number is 2 then training sample value 2
Rule3:if voltage is normal and the class number is 1 then training sample value is 5
Rule4:if voltage is normal and the class number is 2 then training sample value is 14
Rule5:if voltage is normal and the class number is 3 then training sample value is17
Rule6:if voltage is normal and the class number is4 then training sample value is 15
Rule7:if voltage is normal and the class number is 5 then training sample value is 11
Rule8:if voltage is normal and the class number is 6 then training sample value is 8
Rule9:if voltage is too high and the class number is 1 then training sample value is 3
Rule10:if voltage is too high and the class number is 2 then training sample value is 3

7 Conclusions

We found that current discretization methods are either limited to a particular learning algorithm or they tend to ignore the important associative information between the continuous- attribute and class assignment.

We proposed a new algorithm, called CILA, for discretization of continuous attribute. The CILA algorithm can be used with any class labeled data. The tests performed using the CILA algorithm show that it generates discretization schemes with almost always the highest dependence between the class labels and the discrete intervals, and always with significantly lower number of intervals, when compared with other state-of-the-art discretization algorithms. The use of the CILA algorithm as a preprocessing step for a machine learning algorithm significantly improves the results in terms of the accuracy, which are better than by using other discretization algorithms.

An important feature of the CILA algorithm is that it automatically selects the number of intervals, in contrast to many existing discretization algorithm. The CILA algorithm's execution time is comparable to the time of the simplest unsupervised discretization algorithms, and outperforms some supervised algorithms. The above advantages make the CILA algorithm suitable for discretization of data.

References

[1] Catlett, J.: On Changing Continuous Attributes into ordered discrete Attributes, Proc. European Working Session on Learning, pp.164-178, 1991

[2] Ching J.Y., Wong A.K.C. & Chan K.C.C.: Class-Dependent Discretization for Inductive Learning from Continuous and Mixed Mode Data, IEEE Transactions on Pattern Analysis and Machine Intelligence, v.17, no.7, pp. 641-651, 1995

[3] Chiu D., Wong A. & Cheung B.: Information Discovery through Hierarchical Maximum Entropy Discretization and Synthesis, In: Piatesky-Shapiro G., Frowley W.J. (Eds.) Knowledge Discovery in Databases,MIT Press, 1991

[4] Cios, K. J. & Kurgan, L.: CLIP4 – a hybrid algorithm generating production rules, 2001.

[5] Cios K. J. & Kurgan L.: Hybrid Inductive Machine Learning: An Overview of CLIP Algorithms. In: L. C. Jain, and J. Kacprzyk (Eds.) New Learning Paradigms in Soft Computing, Physica-Verlag (Springer), 2001

[6] Cios, K. J., Pedrycz, W. & Swiniarski, R.: Data Mining Methods for Knowledge Discovery. Kluwer, 1998, http://www.wkap.nl/book.htm/0-7923-8252-8

[7] Dougherty J., Kohavi R. & Sahami M.: Supervised and Unsupervised Discretization of Continuous Features, Proc. of the 12th International Conference on Machine Learning, pp.194-202, 1995

[8] Fayyad U.M. & Irani K.B.: On the Handling of Continuous-Valued Attributes in Decision Tree Generation, Machine Learning, v.8, pp.87-102, 1992

[9] Kerber R.: ChiMerge: Discretization of Numeric Attributes, Proc. AAAI-91, 9th International Conference on Artificial Intelligence, pp.123-128, 1992

[10] Liu H. & Setiono R.: Feature Selection via Discretization, IEEE Transactions on Knowledge and Data Engineering, v.9, no.4, pp.642-645, 1997 [11] Paterson, A. & Niblett, T.B.: ACLS Manual, Edinburgh: Intelligent Terminals, Ltd, 1987

[11] Paterson, A. & Niblett, T.B.: ACLS Manual, Edinburgh: Intelligent Terminals, Ltd, 1987

[12] Pfahringer B.: Compression-Based Discretization of Continuous Attributes, Proc. of the 12th International Conference on Machine Learning, pp.456-463, 1995

[13] Quinlan, J.R.: C4.5 Programs for Machine learning, Morgan-Kaufmann, 1993

[14]Tolun, M. and Abu-Soud, S.: ILA: AnInductive Learning Algorithm for Rule Extraction, Expert Systems with Applications, 14(3), (1998) 361-370.

[15] Tou J.T. & Gonzalez R.C.: Pattern Recognition Principles, Addison-Wesley, 1874

[16] Wong A.K.C. & Chiu D.K.Y.: Synthesizing Statistical Knowledge from Incomplete Mixed-Mode Data, IEEE Transactions on Pattern Analysis and Machine Intelligence, v.9, pp. 796-805, 1987

[17] Wong A.K.C. & Chiu D.K.Y.: Synthesizing Statistical Knowledge from Incomplete Mixed-Moe Data, IEEE Transactions on Pattern Analysis and Machine Intelligence, v.9, pp. 796-805, 1987

[18] Wong A.K.C. & Liu T.S.: Typicality, diversity and feature pattern of an ensemble, IEEE Trans

[19] Wu X.: A Bayesian Discretizer for Real-Valued Attributes, The Computer Journal, v.39, 1996

[20] University of California, UCI Machine Learning Repository, http://www.ics.uci.edu/~mlearn/MLRepository.html