Preprocessing Data to Mining a Power Generation Database

Manuel Mejía-Lavalle, Guillermo Rodríguez, Gustavo Arroyo
Instituto de Investigaciones Eléctricas / Gerencia de Sistemas Informáticos
Av.Reforma 113, Col. Palmira, Cuernavaca, Morelos, México 62490
Tel/Fax (01 777) 3 18 26 58
mlavalle@iie.org.mx

ABSTRACT
Four discretization methods were applied and evaluated on a real power generation database with hydroelectric mexican utilities information from years 1988 to 2001. In this paper we present the results obtained using the discretization methods "ChiMerge", "1R", "Equal Interval Width" and "Recursive Minimal Entropy" as preprocessing previous to data mining. The discretized data then was load to the "C4.5" mining tool to obtain production rules (hidden knowledge). We evaluated accuracy, knowledge amount reduction and processing time.

Keywords: Artificial intelligence, Data mining, Power system planning, Preprocessing data,

I. INTRODUCTION

It has been estimated [1] that the amount of information in the world doubles every 20 months. The size and number of databases probably increases even faster. To achieve the use this huge amount of information is necessary to apply algorithms and tools specialized to automatically discover the knowledge hidden in such information. The process of the non trivial extraction of implicit, previously unknown, and potentially useful information from data is known as data mining or knowledge discovery.

There exists a growing interest in the research community in the area of knowledge discovery from databases; not only at the academic institutions as MIT, UCLA, Stanford, Carnegie Mellon, Texas, CALTEC, Johns Hopkins, etc. but also, in the industry, they are interested in the topic, like strong companies as IBM, Mitsubishi, NASA, GTE and AT&T.

Data mining has been successfully applied to many different fields and problems of the real world, for example, in large-scale telecommunication networks [2], in the production process of a nuclear fuel power plant [3], in the flashover analysis of high tension insulators [4], or in the production of electrical energy [5].

Some of the techniques traditionally used to accomplish data mining are: artificial neural nets, induction of decision trees, decision rules and nearest-neighbor methods.

Data mining is applied mainly when large amounts of historical data have been stored and it is expected to exploit them by seeking the implicit knowledge hidden in this information; in other words, it is sought to determine trends or behavior patterns that allow to improve current procedures applied in marketing, production, operation or maintenance.

Such is the case of the power generation database developed and maintained by the Performance Control and Informatics Unit of the Electric Generation Division of CFE (Comisión Federal de Electricidad), the national electric utility company responsible for generating, transmitting and distributing electrical energy in Mexico. This database contains the basic data to compute the performance ratios or indexes for each generating unit (hydro or thermoelectric), power plant, generation region and for the whole CFE. These indexes are classified in seven groups: process performance, labor productivity, economic productivity, training, budget and process costs.

Before applying data mining tools to a data set, there are important reasons to employ discretization methods as a preprocessing phase [6, 7, 8]. Although continuous attribute discretization has received significant attention in the machine learning community (only recently), still there are a lot of research work to do in this topic, particularly from the application point of view.

This paper describes the reasons for attribute discretization preprocessing, the previous work on discretization and the results of an empirical evaluation of four discretization methods: "ChiMerge", "1R", "Equal Interval Width" and "Recursive Minimal Entropy"; we evaluated accuracy, knowledge amount reduction and processing time. Also describes the subset of the database used for knowledge discovery to which the discretization methods were applied before using the data mining algorithm "C4.5". The results are presented, and the advantages and disadvantages of each discretization method are discussed in the context of the application (power generation database). Although our work focuses on this power generation database, the conclusions can apply also to other similar problems.

II. DISCRETIZATION OF NUMERIC ATTRIBUTES

Discretization can turn continuous numeric attributes into discrete ones; in other words, discretization is performed by dividing the values of a numeric (continuous) attribute into a small number of intervals, where each interval is mapped to a discrete

J. Díaz de León, G. González, J. Figueroa (Eds.): Avances en Ciencias de la Computación, pp. 83-93, 2003.

(categorical or nominal) symbol; unless users are knowledgeable about the problem domain and understand the behavior of the classification algorithm, they won't know which discretization is best (something they expected the classification system to tell them) [9].

The reasons for attribute discretization are:

•Algorithmic requirements. Many algorithms developed in the machine learning community focus on learning in nominal attribute spaces [10]. However, many real-world classification task exist that involve continuous attributes where such algorithms could not be applied unless the continuous attributes are first discretized.

•Increasing the speed of induction algorithms. Processing time reduction is important in real-time systems where response time is a critical factor, and in any application it is desirable. Intuitively, the data mining algorithm processing time must be reduced when discretization is applied to the data since the algorithm has less attribute values to work with.

•Improve the performance (accuracy) of the induction tool. The rules, obtained by the discovery algorithm from data that has been discretized, use less values to model (represent) the knowledge. This is important to the final users in order to understand, interpret, but before applying the rules to take decisions, it is important to verify that no accuracy has been lost as compared to the rules obtained from the original data.

•Knowledge amount reduction. In the previous point, the number of values was the issue, here, intuitively, we expect that the number of rules is reduced if data is discretized before the mining algorithm is applied to it. And also, for the same reasons mentioned, this is important to the final users. We can also expect a reduction not only in the number of rules but in the number of conditions in the antecedents of the rules.

There are unsupervised and supervised discretization methods: the former do not make use of instance labels (for the class column) in the discretization process; the second type utilize the class labels. Generally, it has been observed that the supervised methods produced better results.

Discussing about previous work on discretization, Dougherty [11] points out that the simplest unsupervised discretization methods are the "Equal Interval Width" and the "Equal Frequency Interval" (they divide a continuous variable into k intervals where, given m instances, each interval contains m/k, possibly duplicated, adjacent values) which are vulnerable to out-liers that may drastically skew the range. As for the supervised methods, there exists a good amount of them, a fact that makes it difficult to select the best for a particular database. Some of these methods are: "Maximal Marginal Entropy" [12], "Vector Quantization" [13], "Hierarchical Discretization" [14], "Predicative Value Maximization" [15], "D-2" [16], "Adaptive Quantizers" [17], "ChiMerge" [9], "1R" [18], "Recursive Minimal Entropy" [19], "Monothetic Contrast Criterions" [20], "StatDisc" [21] and "Chi2" [22].

Some experimentation has been done with discretization methods, Catlett [16] has explored the use of entropy-based discretization in decision tree domains as a means of achieving an impressive increase in the speed of induction on very large data sets with many continuous attributes (he reports over a 10 to 50-fold speed-up). Kerber [9] compared the accuracy of the methods "ChiMerge", "D-2", "Equal Interval Width" and "Equal Frequency Interval" using two databases and the "Backpropagation" neural net as classifier algorithm. Dougherty [11] performed study to compare the methods "Equal Interval Width" "1R" y "Recursive Minimal Entropy Partitioning" applied to 16 databases using "C4.5" and "Naive Bayes" induction algorithms; it is mention how the accuracy improved in some cases, but no information is included about how the size of the extracted knowledge (number of rules and average number of conditions the antecedents of the rules), and the processing time are affected. For three databases, Liu [22] compared the accuracy and the number of attributes of the produced by "C4.5" without and with the "Chi2" algorithm discretization.

However, none of these studies has measured joint impact of discretization on the accuracy, the of the knowledge, and the processing time to produce it. The joint evaluation of these aspects is important since sometimes the accuracy can be improved the number of rules and the processing time increased. In this paper, we look for the ideal of finding a discretization method that causes to improve accuracy of the knowledge extracted by the mining algorithm, and also reduces the number of and the processing time.

III. PROBLEM DESCRIPTION

The data was selected by the personnel of Performance Control and Informatics Unit of CFE. table was built with 32 attributes or variables (columns) and 441 instances or records (rows) corresponding hydroelectric information for the years 1988 to 2001. The 32 variables include the following: power identifier, date, plate and effective capacity. unavailability by type of failure; outage equivalent hours due to decrements, number of outages outage hours due to failure and due to routine corrective maintenance and other causes; kilocalories; net and gross generation; permanent workers used in maintenance, in operation, and other activities; additional workers used maintenance, in operation, and other activitie equivalent substitution workers in maintenance, operation, and in other activities; total person positions; accidents that cause lost of time; accide in transit; days lost due to accidents; days lost

accidents in transit; sum of disabilities in percent; and various expenses.

With this data set, a supervised [23] data mining experiment was outlined as follows:

•To use the variable Power Plant Factor (PF), as the "class" or focus of attention for the experiment. To calculate the PF the following formula was used:

$$PF = \frac{\text{Gross Generation}}{HP * \text{Net Generation}} * 100$$

where HP (hours per period) is equal to 8,760 hours (365 days multiplied by 24 hours).

•To cluster the PF variable according to Table I (these intervals were defined by the users of the database).

TABLE I. Plant Factor Classes

Plant Factor Interval (%)	Class Value
0.00 - 19.99	Very-Low
20.00 - 39.99	Low
40.00 - 49.99	Regular
50.00 - 69.99	Average
70.00 - 79.99	Good
80.00 - 89.99	Very-Good
90.00 - 100.0	Excellent

•To substitute the four unavailability columns by its corresponding percentage values.

•To eliminate 9 columns that were found to be general variables of no interest.

The result was a database table with 21 columns of continuous numeric values and one discretized column containing PF class values.

We choose to use "C4.5" [24] since in [25] a comparison between the data mining tools "C4.5", "CN2", "FOIL", and "PEBLS" as applied to this power generation database was presented. After applying these tools it was appreciated that "C4.5" was the tool that obtains the more concise knowledge, since it generates the smaller number of production rules with an acceptable error, but with biggest processing time.

Considering that "C4.5" performs better (accuracy and rule amount), it was used as starting point for the discretization preprocessing experiments presented here. Other reasons for our choice are: "C4.5" works well for many problems and is well known; additionally, "C4.5" selects relevant attributes by itself in tree branching so it can be used as a benchmark, to verify the effects of discretization.

The "C4.5" data mining tool is based on the "ID3" algorithm [26], recursively divides the data set into homogeneous groups by selecting the attribute that best partitions (or explains) the data according to the focus of attention attribute; the selection is accomplished by calculating the entropy of each of the attributes and choosing that of smaller entropy. This tool presents the extracted knowledge in the form of decision trees and then in the form of production rules. This tool internally includes a "local discretization method", that produces partitions that are applied to localized regions of the instance space, it is say, discretization is performed not as a preprocessing step, but dynamically as the tool runs; the same measure used to choose the best attribute to branch on at each node of the decision tree is used to determine the best value for splitting a numeric attribute into two intervals; this value, called a cutpoint, is found by exhaustively checking all possible binary splits of the current interval and choosing the splitting value that maximizes the information gain measure (smaller entropy).

IV. DESCRIPTION OF METHODS EVALUATED

The discretization methods that were applied to the data described in the previous section were "ChiMerge" [9], "1R" [18], "Equal Interval Width" and "Recursive Minimal Entropy" [19]; the evaluation of the methods was made on the basis of: reduction of the amount of the discovered knowledge, accuracy (certainty) of the extracted knowledge and time to obtain results (processing time). A brief description of each of these discretization methods follows:

•"Equal Interval Width". It involves sorting the observed values of a continuous attribute and dividing the range of observed values of the attribute into k equally sized intervals, where k is a parameter supplied by the user. If an attribute x is observed to have values bounded by x_{min} and x_{max} then this method computes the interval width (W) as follows:

$$W = (x_{max} - x_{min}) / k$$

and constructs interval boundaries, or thresholds, at $x_{min} + (W * i)$ where $i = 1,..., k-1$. The method is applied to each continuous attribute independently. It makes no use of instance class information whatsoever and is thus an unsupervised discretization method.

•"Recursive Minimal Entropy". It is based on a minimal entropy heuristic: if we are given a set of instances S, an attribute A, and a partition boundary T, the class information entropy of the partition induced by T denoted $E(A,T;S)$ is given by:

$$E(A,T;S) = (|S_1| / |S|) * Ent(S_1) + (|S_2| / |S|) * Ent(S_2)$$

For a given attribute A, the boundary T_{min} which minimizes the entropy function over all possible partition boundaries is selected as a binary discretization boundary. This method can be applied

recursively to both of the partitions induced by T_{min} until some stopping condition is achieved, thus creating multiple intervals on the attribute *A*. Since the partition along each branch of the recursive discretization are evaluated independently using this stopping criteria, some areas in the continuous spaces will be partitioned very finely whereas others (which have relatively low entropy) will be partitioned coarsely.

- "1R". Induces one-level decision trees, sometimes called decision stumps. In order to properly deal with domains that contain continuous valued attributes, a simple supervised discretization method is given. This method, referred as 1RD (One-Rule Discretizer), sorts the observed values of a continuous attribute and attemps to greedily divide the domain of the attribute into intervals that each contain only instances of one particular class; since such a scheme could possibly lead to one interval for each observed real value, the algorithm is constrained to form intervals of at least some minimum size.
- "ChiMerge". Begins by placing each observed real value into its own interval and proceeds by using the X^2 (chi-square) test to determine when adjacent intervals should be merge. This method tests the hypothesis that the two adjacent intervals are statistically independent by making an empirical measure of the expected frequency of the classes represented in each of the intervals. The extent of the merging process is controlled by the use of a threshold, indicating the maximum X^2 value that warrants merging two intervals.

"ChiMerge" was implemented in language C by the authors of this paper and suggestions found [9] were incorporated to the algorithm. For the other three methods we use the Discretize Module from the MLC++ library [27] obtained through Internet.

V. APPLICATION AND EVALUATION

All the following discretization experiments were executed on a SPARC/20 workstation (32 bit bus, 64 Mb RAM, 66 MHz, RISC-SPARC).

A. Experiment 1

No discretization preprocessing was applied to the data set and "C4.5" was applied directly to the data using the default parameters (for example: 10 trials, weight 2, certainty factor 25%, verbosity level 0, etc.), the result is shown in Table II.

TABLE II. Results of applying "C4.5" without previous discretization

Observed Characteristic	Obtained Value
Error (Best decision tree)	33.4 %
Error (Best ruleset)	20.6 %
Number of production rules	27
Average antecedents per rule	5.11
Processing time minutes	20.9

In Table II, "Error (Best decision tree)" is the "Estimate error" as referred by the "C4.5" tool. The observed characteristics "Error (Best ruleset)", "Number production rules", and "Average antecedents per rule" correspond to the "composite ruleset" generated "C4.5". The "Processing time" refers to the time obtain the decision tree (18.04 minutes) plus the time to obtain the production rules from the tree (2.86 minutes).

B. Experiment 2

"Equal Interval Width" (referred to as "Binning" in MLC++ library) was used to discretize the data before the application of "C4.5". Actually, the experiment included 4 different discretizations preprocessing and 4 corresponding executions of "C4.5". The results shown in Table III where the parameter INIT_VALUE is the number of intervals used for each of the discretizations, and where: INIT_VAL = number intervals; Time = Processing time; # Rules = number of rules; Error Rules = Error (Best ruleset).

TABLE III. Results of applying "C4.5" (previous "Equal Interval Width")

INIT_VAL	Time	# Rules	Error Rules
0	8.0	17.45	51.6
10	5.9	13.55	56.9
50	11.2	17.27	50.8
100	5.2	17.45	51.6

C. Experiment 3

"1R" was used to discretize the data before application of "C4.5". Actually, the experiment included 5 different discretizations and corresponding executions of "C4.5". The results shown in Table IV where the parameter MIN_INT the minimum number of instances per label. (Holte suggest 6 as minimum number of instances per

TABLE IV. Results of applying "C4.5" (previous "1R" discretization)

MIN_INT	Time	# Rules	Error Rules
0	4.6	35.00	33.3
10	5.2	30.82	35.2
50	5.0	16.64	40.2
100	7.9	9.45	52.8
130	4.3	2.36	50.8

D. Experiment 4

"Recursive Minimal Entropy" (referred to as "Entropy" in the MLC++ library) was used to discretize the data before the application of "C4.5". The experiment included 5 different discretizations and 5 corresponding executions of "C4.5". The results are shown in Table V where the parameter MIN_INT is the minimum number of instances per label and we maintained constant to the default value of zero the parameter INIT_VALUE.

E. Experiment 5

In this experiment, "ChiMerge" was used to discretize the data before the application of "C4.5". A 0.95 significance level with 6 degrees of freedom was used since the class column has 7 different class values (this is equivalent to a 12.59 X^2-*threshold*). The results are shown in Table VI.

TABLE V. Results of applying "C4.5" (previous "Entropy" discretization)

MIN_INT	Time	# Rules	Error Rules
0	3.0	8.45	37.6
10	3.3	8.45	37.6
50	4.5	6.27	44.5
100	9.5	6.36	42.5
130	5.5	8.27	44.4

TABLE VI. Results of applying "C4.5" with previous "ChiMerge" discretization

Observed Characteristics	Obtained Value
Error (Best decision tree)	41.6%
Error (Best ruleset)	30.2%
Number of production rules	21
Average antecedents per rule	2.66
Processing time	3.08 minutes

VI. DISCUSSION

Some tradeoffs are analyzed for each discretizer considering as reference the experiment 1 where no discretizer was used.

- "Equal Interval Width". Even though this discretizer reduces the processing time (up to 20-fold speed-up), the number of rules (up to 4.5-fold) and the average antecedents per rule (up to 3.3-fold); the accuracy is not acceptable (between 50% and 57% error for the best ruleset). Because of this, this discretizer is not a good alternative for our database. This result agrees with the ones reported in the literature.
- "Recursive Minimal Entropy". The results are similar to the experiment with the "Equal Interval Width" discretizer, namely, the accuracy is not acceptable (between 37.6% and 44.5% error for the best ruleset), even though this discretizer reduces the processing time (up to 10-fold speed-up), the number of rules (up to 5-fold) and the average antecedents per rule (up to 2.3-fold). We observed that the error is almost the same for different values of MIN_INT, this dicretizer should be used with caution. Our results do not agree with the ones reported in [11]
- "1R". Our results agree with Holte's suggestion [18] of assigning a value of 6 to the minimum number of instances per label since we obtained the best result with a value of 7.

In this case (MIN_INT = 7), the error for the best ruleset was in the order of the error obtained in the experiment 1 (28.8% vs 20.6%) that has no previous discretization. Reductions in the processing time (up to 2.9-fold speed-up), and the average antecedents per rule (up to 2.8-fold) were obtained, but a little increase in the number of rules (up to 1.2-fold) was observed. This results are shown in Table VII.

TABLE VII. Results of applying "C4.5" with previous "1R" discretization (MIN_INT = 7)

Observed Characteristics	Obtained Value
Error (Best decision tree)	47.6%
Error (Best ruleset)	28.8%
Number of production rules	33
Average antecedents per rule	1.81
Processing time	7.1 minutes

- "ChiMerge". With this discretizer the error for the best ruleset was similar to the error obtained using "1R" (30.2% vs 28.8%) and as compared to experiment 1, significant reductions were obtained in processing time (3.08 minutes vs 20.9 minutes), average antecedents per rule (2.66 vs 5.11), and number of rules (21 vs 27).

VII. CONCLUSIONS

The knowledge discovery in large databases is an area of artificial intelligence that is an alternative to the analysis of data and that uses "intelligent" computational searching techniques to discover knowledge based on data mining algorithms that are capable of identifying regularities, tendencies or patterns hidden in the data sets that are not so easy to find using traditional methods, to improve the current procedures of a company for marketing, production, operation, maintenance, or other.

In this paper we evaluated four discretization methods and successfully obtained knowledge from a subset of the power generation database of the Performance Control and Informatics Unit of CFE for hydroelectric generating units.

The discretization preprocessing methods that performed better were "ChiMerge" and "1R" since they caused "C4.5" to improve its results when applied to power generation database: it was achieved to reduce processing time and the size of the extracted knowledge (number of rules and average antecedents rule). However, none of the discretization methods caused to improve accuracy since the error for the best ruleset always increased, but it can be appreciated that "ChiMerge" and "1R" the error is reasonable.

the future, research should be done about discretization methods that:

automatically adjust their parameter to obtain the best clustering of continuous numeric attributes, to avoid the execution of many experiments to find the best mix.

consider groups of two or more attributes of the table at the same time instead of just one attribute to perform the discretization. Intuitively, we think that two joint attributes may correlate better than just one with the classes and also the result would cause the mining algorithm to perform better.

perform more simulations with academic databases, to obtain general results and conclusions.

REFERENCES

Frawley, W., et.al., *Knowledge Discovery in Databases: An Overview*, Knowledge Discovery in Databases, Piatetsky-Shapiro, G. eds., Cambridge, MA, AAAI/MIT, 1-27, 1991.

[2]Sasisekharan, R., Seshadri, V., *Data Mining and forecasting in Large-Scale Telecommunications Networks*, IEEE Expert, 11, 1, Feb, 37-43, 1996.

[3]Leech, W., *A Rule Based Process Control Method with Feedback*, Proceedings of the International Conference and Exhibit, Instrument Society of America, 1986.

[4]Mejia, M., Rodriguez, G., Montoya, G., *Knowledge Discovery in High-Voltage Insulators Data*, Proceedings of The Tenth International Conference on Industrial & Engineering Applications of Artificial Intelligence & Expert Systems, Atlanta USA, Jun, 223-230, 1997.

[5]Hernández, V., Rodríguez, G., Morales, E., *Descubrimiento de información en una base de datos de generación eléctrica*, Novena Reunión de Verano de Potencia, Tomo IV, IEEE Sección México, Jul, 31-37, 1996.

[6] Weiss, S., Indurkhya, N., *Preditive data mining*, Morgan Kaufmann, San Francisco, CA, 1998.

[7]Dorian, P., *Data preparation for data mining*, Morgan Kaufmann, San Francisco, CA, 1999.

[8] Witten, I., Eibe, F., *Data mining*, Morgan Kaufmann, San Francisco, CA, 2000.

[9]Kerber, R., *ChiMerge: Discretization of Numeric Attributes*, Proceedings of the 10th National Conference on Artificial Intelligence, MIT Press, 123-128, 1992.

[10]Michalski, R., Stepp, R., *Learning from observations: Conceptual clustering*, Machine Learning: An Artificial Intelligence Approach, R. Michalski, ed., Tioga, Palo Alto USA, 1983.

[11]Dougherty, J., Kohavi, R., Sahami, M., *Supervised and Unsupervisad Discretization of Continuous Features*, Machine Learning: Proceedings of the Twelfth International Conference, A. Prieditis, ed. Morgan Kaufmann Publishers, San Francisco USA, 1995.

[12]Wong, A., Chiu, D., *Synthesizing statistical knowledge from incomplete mixed-mode data*, IEEE Transaction on Pattern Analysis and Machine Intelligence, 9, 796-805, 1987.

[13]Kohonen, T., *Self-Organization and Associative Memory*, Berlin Germany: Springer-Verlag, 1989.

[14]Chiu, D., Cheung, B., Wong, A., *Information synthesis based on hierarchical entropy discretization*, Journal of Experimental and Theoretical Artificial Intelligence, 2, 117-129, 1990.

[15]Weiss, S., Galen, R., Tadepalli, P., *Maximizing the predicative value of production rules*, Artificial Intelligence, 45, 47-71, 1990.

[16]Catlett, J., *On changing continuous attributes into ordered discrete attributes*, Proceedings of the European Working Session on Learning, Y. Kodratoff, ed., Berlin, Germany: Springer-Verlang, 164-178, 1991.

[17]Chan, C., Batur, C., Srinivasan, A., *Determination of quantization intervals in rule based model for dynamic systems*, Proceedings of the IEEE Conference on Systems, Man and Cybernetics, Charlottesvile USA, 1719-1723, 1991.

[18]Holte, R., *Very simple classification rules perform well on most commonly used datasets*, Machine Learning, 11, 63-90, 1993.

[19]Fayyad, U., Irani, K., *Multi-interval discretization of continuous-value attributes for classification learning*, Proceedings of the 13th International Joint Conference on Artificial Intelligence, Morgan Kaufmann, 1022-1027, 1993.

[20]Van de Merckt, T., *Decision trees on numerical attribute spaces*, Proceedings of the 13th International Joint Conference on Artificial Intelligence, 1016-1021, 1993.

[21]Richeldi, M., Rossotto, M., *Class-driven statistical discretization of continuous attributes*, Proceedings of the European Conference on Machine Learning, N. Lavrac, ed., Springer-Verlag, Berlin Germany, 335-338, 1995.

[22]Liu, H., Setiono, R., *Chi2: Feature Selection and Discretization of Numeric Attributes*, Proceedings of the IEEE 7th International Conference on Tools with Artificial Intelligence, Washington USA, Nov, 388-391, 1995.

[23]Langley, P., *Elements of Machine Learning*, Morgan Kaufmann Publishers Inc., 8-9, 1996.

[24]Quinlan, J., *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, 1989.

[25]Mejia, M., Rodriguez, G., *Obtaining Expert System Rules using Data Mining Tools from a Power Generation Database*, to appear in the Expert Systems with Applications An International Journal, J. Liebowitz, ed., Elsevier Science Ltd., 14, Jan, 1998.

[26]Quinlan, J., *Induction of Decision Trees*, Machine Learning, 1, 1, 81-106, 1986.

[27]Kohavi, R., John, G., Long, R., Manley, D., Pfleger, K., *MLC++: A machine learning library in C++*, Tools with Artificial Intelligence, IEEE Computer Society Press, 740-743, 1994.