Scalable Algorithms for Dealing with Missing Values

Yoshikazu Fujikawa and TuBao Ho

Japan Advanced Institute of Science and Technology
Tatsunokuchi, Ishikawa 923-1292 Japan

Abstract. The existence of missing values in datasets is a critical problem in knowledge discovery and data mining. Various methods to deal with missing values have been developed but it is not clear which methods can be appropriately used in a given application. Also, many methods are with high computation complexity and cannot be applicable to large datasets in data mining. In this paper, we first survey existing methods to deal with missing values and report the results of an experimental comparative evaluation in terms of processing cost and quality of imputing missing values. Based on the obtained results, we then propose three cluster-based mean-and-mode algorithms to impute missing values in different situations. Experimental results show that these algorithms with linear complexity can achieve comparative quality as other sophisticated algorithms and therefore are applicable to large datasets.

1 Introduction

The existence of missing values on datasets is a critical problem in knowledge discovery and data mining (KDD). Only a small number of data mining algorithms are designed with functions to deal with missing values while most of them deal poorly with missing values or required to fill-up missing values in the pre-processing step. Having efficient algorithms to deal with missing values (referred to as missing value algorithms) will extend the applicability of many data mining methods. On the other hand, different methods to deal with missing values have been developed but it is not clear which methods can be appropriately used in a given application. The objective of this research is two folds. One is to evaluate several well-known missing value methods in order to get a better understanding in their selection. The other is to develop scalable algorithms to deal with missing values for large datasets in data mining. The key idea of these algorithms is to divide a dataset with missing values into clusters beforehand and replace missing values on each attribute by mean or mode value of the corresponding cluster depending on the attribute is numeric or symbolic, respectively. The paper is organized as follows. Section 2 presents a classification and an evaluation of existing methods to deal with missing values. Section 3 introduces three cluster-based mean-and-mode algorithms that provide an efficient solution to impute missing values. Section 4 concludes the work and suggests future work.

2 Evaluation of existing algorithms for missing values

2.1 Classification of missing values cases

Generally, missing values can occur in data sets in different forms. By observing and analyzing different datasets in the UCI repository, we roughly classify missing values in datasets into three cases:

1. Missing values occur in several attributes (columns),
2. Missing values occur in a number of instances (rows),
3. Missing values occur randomly in attributes and instances.

The occurrence cases of missing values can affect the result of missing value methods, so the selection of suitable missing value methods in each case is significant. For example, a method that ignores instances having missing values cannot be used when most of instances have one or more missing values. Or the nearest neighbor estimator is generally good for the first case, but not good for the second one because it calculates distance between two instances by the values of attributes that do not contain missing values.

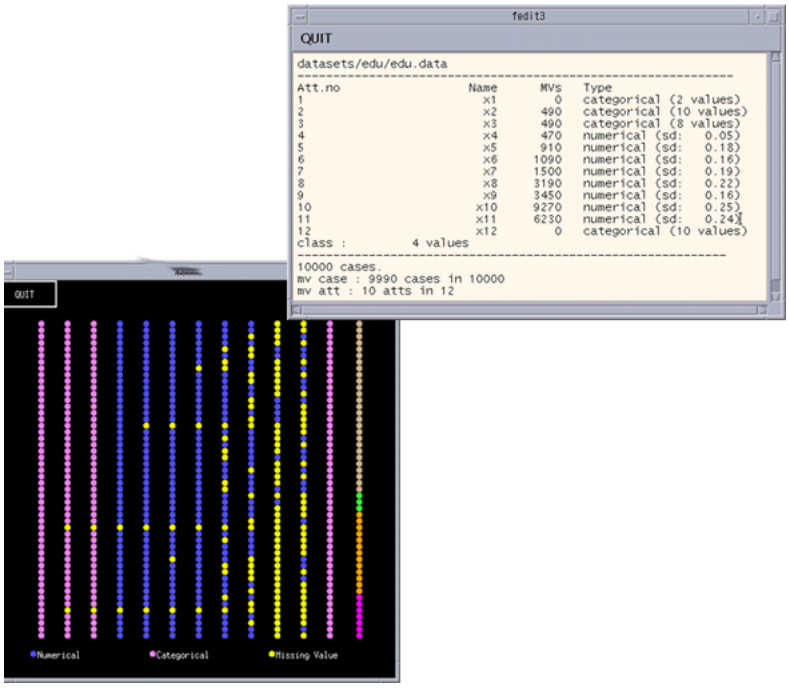

Fig. 1. Visualization of missing values in dataset *edu*

Table 1. Methods for dealing with missing values

Method	Reference	Group
Linear regression	Pyle, 1999	Pre-replacing
Standard deviation method	Pyle, 1999	Pre-replacing
mean-and-mode	Han, 2001	Pre-replacing
Nearest neighbor	Pyle, 1999	Pre-replacing
Autoassociative neural network	Pyle, 1999	Pre-replacing
Decision tree imputation	Quinlan, 1986	Pre-replacing
Case-wise deletion	Liu, 1997	Embedded
Lazy decision tree	Friedman, 1996	Embedded
Dynamic path generation	White, 1987	Embedded
C4.5	Quinlan, 1993	Embedded
Surrogate split (CART)	Breiman, 1984	Embedded

In order to identify the case of missing values for each dataset, we have created a tool that visualizes approximately the distribution of missing values in a data set. Fig. 1 illustrates this tool by showing missing values and types of attributes in dataset (*edu* of 10,000 from the UCI repository.

2.2 Existing methods for dealing with missing values

We classify methods to deal with missing values into two groups: (i) *pre-replacing* methods that replace missing values before the data mining process, and (ii) *embedded* methods that deal with missing values during the data mining process.

We roughly view two kinds of pre-replacing methods: (i) statistical methods, and (ii) machine learning methods. Among missing value methods from the literature that we consider in this work, the statistical ones include linear regression, replacement under same standard deviation [9] and mean-mode method [3]; and machine learning methods include nearest neighbor estimator, autoassociative neural network [9], decision tree imputation [10]. All of these are pre-replacing methods. Embedded methods include case-wise deletion [5], lazy decision tree [2], dynamic path generation [12] and some popular methods such as C4.5 [11] and CART [1]. Generally, statistical methods in our investigation are simpler than machine learning methods but machine learning methods are expected to achieve better quality because of their sophisticated processing. We summarized the above methods in Table 1.

2.3 Evaluation of existing methods

This section gives a summary of a comparative evaluation on effectiveness of the above mentioned algorithms by experiments with different datasets. These algorithms include standard deviation, mean-and-mode, nearest-neighbor estimator, decision tree imputation and See5 (the successor of C4.5). The datasets

Table 2. Comparative evaluation of methods to deal with missing values

Method	Computation Cost	Available kind of attributes	Suitable case of missing values
Mean-and-mode method	Low	Num & Sym	Case 2
Linear regression	Low	Num	Case 2
Standard deviation method	Low	Num	Case 2
Nearest neighbor estimator	High	Num & Sym	Case 1
Decision tree imputation	Middle	Sym	Case 1
Autoassociative neural network	High	Num & Sym	Case 1
Casewise deletion	Low	Num & Sym	Case 2
Lazy decision tree	High	Num & Sym	Case 1
Dynamic path generation	High	Num & Sym	Case 1
C4.5	Middle	Num & Sym	Case 1
Surrogate split	Middle	Num & Sym	Case 1

are taken from the UCI repository whose attributes are all numeric, all symbolic or mixed. Their size varies from several hundreds to several thousands.

We observed the methods on three aspects: (i) computation cost, (ii) applicable types of attributes, (iii) suitable cases of missing values. Because of the space limitation, we do not report in this paper the details of our experiments on existing methods.

On the first aspect, statistical methods are of low computation cost. Although the decision tree imputation approach is good for imputing missing values, it cannot be used for replacing missing values in large datasets because of its high complexity. Other methods with higher complexity of course are not applicable to large datasets.

On the second aspect, three methods can deal with missing values only for one type of attribute, numerical or symbolical. For example, linear regression and standard deviation methods can work only for numeric attributes while decision tree imputation method can work only for symbolic attributes. If these methods are used, discretization of numerical attributes is needed. Nearest-neighbor method is good at dealing with datasets whose attributes are almost numerical.

On the third aspect, statistical methods are generally appropriate when missing values are distributed on some instances (case 1). The others are suitable when missing values are distributed on some attributes (case 2).

Standard deviation method seems better than mean method, but it requires to select which value is suitable from two candidate values. Another problem occurs when the dataset is huge, to know the supplemental value is too complicated, especially if many values are missed in one attribute.

The nearest neighbor estimator in many cases can be very effective, however, its main drawbacks are that given a training dataset, it may require very big

storage. Looking up for neighbors can be very slow, so finding replacement values is slow, too.

The autoassociative neural network approach has been used with success in replacing missing values for datasets of modest dimensionality (up to several hundreds), but building such networks for moderate-dimensionality or high-dimensionality datasets is problematic and very slow. The amount of data required to build a robust network becomes prohibitive, and for replaced value generation of a robust network that actually reflects nonlinearities is needed.

In case-wise deletion, when a dataset has many instances with missing values, this method may cause a huge loss of data and, as a result, may not be used. The effect of surrogate method in CART obviously depends on the magnitude of the correlation in the database between the original attribute and its surrogate. Table 2 summarizes the features of considered methods.

3 Scalable algorithms to deal with missing values

3.1 Basic idea of scalable algorithms

For large datasets with missing values, complicated methods are not suitable because their high computation cost. We tend to find simple methods that can reach performance as good as complicated ones.

The results and experience obtained in the previous session suggested us that mean-and-mode method can be efficient and effective for large datasets with necessary improvements.

The basic idea of our method is the cluster-based filling up of missing values. Instead of using mean-and-mode on the whole dataset we use mean-and-mode in its subsets obtained by clustering. The method consists of three variants of the mean-and-mode algorithm:

1. Natural Cluster Based Mean-and-Mode algorithm (NCBMM),
2. attribute Rank Cluster Based Mean-and-Mode algorithm (RCBMM),
3. K-Means Clustering based Mean-and-Mode algorithm (KMCMM).

Table 3. Algorithm NCBMM

```
1. Divide instances into clusters by their class labels
2. For each cluster
3.    For each attribute which has missing values
4.       If it is numeric, calculate the mean value from all no-missing values,
5.       If it is symbolic, find the most probable value (mode),
6.       Fill up missing values on this attribute by the cluster mean or mode.
```

Table 4. Algorithm RCBMM

1. For each missing attribute a_i
2. Make a ranking of all n symbolic attributes $a_{j_1}, a_{j_2}, \ldots, a_{j_n}$ in decreasing order of distance between a_i and each attribute a_{j_k}
3. Divide all instances into clusters based on the values of a_h, where a_h is the attribute, which has the highest rank among $a_{j_1}, a_{j_2}, \ldots, a_{j_n}$.
4. Replace missing value on attribute a_i of an instance by the mode of each cluster to which it belongs to.
5. Repeat steps from 2 to 4 until all missing values on attribute a_i are replaced where h changes to the next number of ranking.

NCBMM uses the class attribute to divide instances into natural clusters and uses the mean or mode of each cluster to fill up missing values of instance belongs to the cluster depending on the attribute id numeric or symbolic, respectively. This algorithm is the simplest improvement of the mean-and-mode algorithm in case of supervised data but as shown in next sections it is very efficient if applicable. RCBMM and KMCMM divide a dataset into subsets by using the relationship between attributes. The starting point of these algorithms came from the question whether the class attribute is always the key attribute for clustering an arbitrary descriptive attribute? For some attributes, some of descriptive attributes may be better for clustering than the class attribute. The remark that to cluster instances concerning one missing attribute, the key attribute selected from all attributes may give better results than NCBMM.

3.2 The Proposed Algorithms

Natural Cluster Based Mean-and-Mode algorithm (NCBMM). NCBMM algorithm can be applied to supervised data where missing value attributes can be either symbolic or numeric. It produces a number of clusters equal to the number of values of the class attribute. At first, the whole instances are divided into clusters, where instances of each cluster have the same value of the class attribute. Then, in each cluster, the mean value is used to fill up missing values for each numeric attribute, and the mode value is used to fill up missing values for each symbolic attribute. We describe the algorithm in Table 3.

Attribute Rank Cluster Based Mean-and-Mode algorithm (RCBMM). RCBMM can be applied to filling up missing values for symbolic attributes independently with the class attribute. It can be applied to both supervised and unsupervised data. Firstly, for one missing attribute, this method ranks attributes by their distance to the missing value attribute. The attribute that has

Table 5. Algorithm KMCMM

1. For each missing attribute a_i,
2. Make a ranking of all n numeric attributes $a_{j_1}, a_{j_2}, \ldots, a_{j_n}$ in increasing order of absolute correlation coefficients between attribute a_i and each attribute a_{j_k}.
3. Divide all instances by k-means algorithm based on the values of a_h that is the attribute of highest rank among $a_{j_1}, a_{j_2}, \ldots, a_{j_n}$.
4. Replace missing value on attribute a_i by the mean of each cluster.
5. Repeat steps from 2 to 4 till all missing values on attribute a_i are replaced where h changes to the next number of ranking.

smallest distance is used for clustering. Secondly, all instances are divided into clusters each contains instances having the same value of the selected attribute. Thirdly, the mode of each cluster is used to fill up missing values. This process is applied to each missing attribute. We describe the algorithm in Table 4.

We can have several ways to calculate distances between attributes. Our idea is to measure how two attributes have similar distributions of values. For this purpose we employ the distance between partitions proposed by Mantaras [8]. Suppose that P_A and P_B are two partition of n and m subsets of values of attributes A and B. The distance between A and B is defined as follows ($1 \leq i \leq n$, $1 \leq j \leq m$)

$$d_N(P_A, P_B) = \frac{d(P_A, P_B)}{I(P_A \cap P_B)}$$

$$P_i = P(A_i), P_j = P(B_j), P_{ij} = P(A_i \cap B_j), P_{j/i} = P(B_j/A_i)$$

$$I(P_A \cap P_B) = -\sum_{i=1}^{n}\sum_{j=1}^{m} P_{ij} \log_2 P_{ij}$$

$$\begin{aligned} I(P_B/P_A) &= I(P_B \cap P_A) - I(P_A) \\ &= -\sum_{i=1}^{n}\sum_{j=1}^{m} P_{ij} \log_2 \left[\frac{P_{ij}}{P_i}\right] \\ &= -\sum_{i=1}^{n} P_i \sum_{j=1}^{m} P_{j/i} \log_2 P_{j/i} \end{aligned}$$

$$d(P_A, P_B) = I(P_B/P_A) + I(P_A/P_B)$$

Table 6. Properties of datasets used in experiments

dataset	inst.	miss inst.	atts	miss atts	class	type of miss att	miss case values
adt	22,747	1,335	13	2	2	sym	case 1&2
att	10,000	2,430	9	8	2	sym	case 1&2
ban	5,400	2,610	30	25	2	mixed	case 2&3
bcw	6,990	160	9	1	2	num	case 1
bio	2,090	150	5	2	2	num	case 3
bld	3,450	560	6	3	2	num	case 1
bos	5,060	860	13	3	3	num	case 1
bpr	3,600	75	16	3	3	sym	case 1
census	299,285	156,764	41	8	2	sym	case 1&2
cmc	14,730	2,002	9	3	3	mixed	case 1
crx	6,900	337	15	7	2	mixed	case 1&2
der	3,660	74	34	1	6	num	case 1
dna	2,372	354	60	3	3	sym	case 1
ech	1,310	230	6	5	2	num	case 3
edu	10,000	9,990	12	10	4	mixed	case 1&2
hab	3,060	562	3	3	2	num	case 3
hco	3,680	3,290	19	18	2	mixed	case 2
hea	3,030	60	13	2	2	sym	case 1
hep	1,550	750	19	15	2	mixed	case 2
hin	10,000	4,050	6	5	3	sym	case 1
hur	2,090	220	6	2	2	num	case 1
hyp	3,772	3,772	29	8	5	mixed	case 1
imp	2,050	80	22	5	5	mixed	case 1
inf	2,380	250	18	2	6	sym	case 1
lbw	1,890	240	8	3	2	mixed	case 1
led	6,000	1,770	7	7	10	sym	case 3
pima	7,680	3,750	8	4	2	num	case 1
sat	6,435	1,195	36	2	6	num	case 1
seg	23,100	3,240	11	3	7	num	case 1&3
smo	28,550	5,340	8	2	3	mixed	case 1
tae	1,510	120	5	5	3	sym	case 3
usn	11,830	10,402	27	26	3	num	case 1&2
veh	8,460	1,550	18	3	4	num	case 1
vot	4,350	420	16	1	2	sym	case 1
wav	3,600	674	21	3	3	num	case 1

K-Means Clustering based Mean-and-Mode algorithm (KMCMM). KMCMM can be applied to filling up missing values for numeric attributes independently with the class attribute. Therefore, it can be applied to both supervised and unsupervised data. We describe the algorithm for KMCMM in Table 5.

The *correlation coefficient* r used in KMCMM is calculated from p pairs of observations (x, y) according to the following formula:

$$r = \frac{S_{xy}}{\sqrt{S_{xx}}\sqrt{S_{yy}}}$$

$$S_{xy} = \sum (x - \bar{x})(y - \bar{y}), S_{xx} = \sum (x - \bar{x})^2, S_{yy} = \sum (y - \bar{y})^2$$

The k-means clustering algorithm proceeds as follows. First, it randomly selects k of the objects, each of which initially represents a cluster mean or center. Each remaining object is assigned to the cluster who mean is most similar with it. It then computes the new mean for each cluster. This process iterates until the criterion satisfied. Typically, the squared-error criterion is used, defined as

Table 7. Error rates of datasets after treating missing values

dataset	See5	NCBMM NCBMM	NCBMM RCBMM	KMCMM NCBMM	KMCMM RCBMM	NN NCBMM	NN RCBMM	decision tree	ANN
adt	**15.1**	*15.2*	**15.1**					*15.2*	
att	*2.2*	1.2	1.2					*2.3*	**1.1**
ban	*3.0*	2.5	2.5	2.6	2.5			2.5	
bcw	*0.9*	**0.7**		0.8		0.8		0.8	0.8
bio	*2.2*	1.8		1.8		1.8		1.8	1.8
bld	*4.7*	**3.3**		3.5		3.7		3.7	3.7
bos	*3.8*	2.9		2.9		2.9		2.9	2.9
bpr	3.8	**3.7**	3.8					3.8	*3.9*
cmc	6.7	5.2	5.5	5.2	5.5			*11.2*	**4.7**
crx	*2.5*	2.4	2.3	2.4	2.3	2.3	2.3	2.3	2.3
der	1.4	1.4						1.4	
dna	4.7	4.7	4.7					**4.6**	
ech	*5.4*	**3.5**		**3.5**				4.0	3.9
edu	5.2	**0.0**	**0.0**	**0.0**	**0.0**			7.7	1.8
hab	4.3	**2.9**		**2.9**				*7.6*	3.8
hco	0.9	**0.0**	**0.0**	**0.0**	**0.0**			*2.8*	*2.7*
hea	3.0	3.0	3.0					3.0	3.0
hep	*3.8*	**1.3**	**1.3**	**1.3**	**1.3**	2.5	2.6	2.2	2.2
hin	*12.9*	**9.0**	10.6					10.4	11.3
hur	*2.8*	**2.0**		**2.0**		2.2		2.2	2.1
hyp	0.4	0.4	0.4	0.4	0.4			0.4	0.4
imp	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	
inf	2.4	**2.3**	**2.3**					2.4	2.4
lbw	5.1	**3.9**	**3.9**	**3.9**	**3.9**	5.2	5.2	*8.6*	4.7
led	29.1	**25.1**	**25.1**					25.2	*32.0*
pima	*3.7*	**1.5**		**1.5**		3.0		2.9	2.9
sat	13.1	13.6		*14.1*		13.5		**9.0**	13.1
seg	*1.2*	**0.6**		**0.6**		0.7		0.7	0.7
smo	2.2	**1.3**	**1.3**	**1.3**	**1.3**	1.9	1.9	*7.3*	1.6
tae	**2.8**	**2.8**	4.0					*11.2*	
usn	*4.2*	**1.3**		2.4				2.8	2.9
veh	*3.0*	**2.7**		**2.7**		**2.7**		2.8	2.8
vot	*1.9*	**1.0**	**1.0**					1.1	1.4
wav	*24.5*	24.0		23.5		*24.6*		**13.1**	24.4

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} |p - m_i|^2$$

where E is the sum of square-error for all objects in the database, p is the point in space representing a given object, and m_i is the mean of cluster C_i.

3.3 Evaluation

Methodology. To evaluate the performance of pre-replacing methods our method consists of two phases: (1) filling up missing values on dataset by pre-replacing methods and measuring the executing time, (2) evaluating the quality of the replaced datasets in terms of error rate in classification. In our study, concretely each missing value dataset is filled up by suitable pre-replacing methods to yield datasets without missing values. Then the same data mining methods (See5) are applied to the non-missing value datasets as well replaced datasets and their results are compared in order to evaluate the quality of replacing methods.

Table 8. Error rates and time of processing on census-income dataset

Dataset	Error rate (%)	Processing time (sec.)
See5	4.6	200.00
NCBMM	4.6	171.41
RCBMM	4.6	651.38

Six replacing methods, NCBMM, RCBMM, KMCMM, nearest neighbor estimator, autoassociative neural network and decision tree imputation and one embedded method, C4.5, are used for this comparative evaluation. For evaluating the quality of replaced datasets, three classification methods, C4.5, Naive Bayesian classifier, k-nearest neighbor classifiers are used. Fifteen UCI datasets were replaced and classified. There are three types of datasets by the attributes which contain missing values: all missing attributes are numerical (except for class attribute), symbolic and mixture of symbolic and numeric. The properties of each dataset are summarized in Table 6, and the error rates after treating missing values are summarized in Table 7.

The experimenting result shows that NCBMM, RCBMM and KMCMM are much faster than other methods and higher accuracy than others (number in bold).

We also compare the methods on the large census dataset containing 299,285 instances with 156,764 instances having missing values. This set has eight numeric and thirty-three symbolic attributes except the class attribute and missing values occur on only symbolic attributes. We also used See5 for classification. We compared the error rate of See5 obtained directly, and that of See5 obtained after replacing missing values by NCBMM and RCBMM. And we measured the execution time for preparing at each replacing method. Experiments were done on a ultra spark machine and Sun-UNIX operating system. Table 8 shows that See5, NCBMM and RCBMM have the same accuracy. The time to replace missing values for each of our two methods is around 3 and 11 minutes respectively, i.e., they are scalable methods.

4 Discussion

The most important problem in our work is how to reduce processing cost in filling up missing values to deal with large datasets in data mining. It is showed that the proposed cluster-based algorithms could deal with missing values as good as other complicated methods and with low cost. Thus, they can be applied to large datasets. The followings are advantages of our methods:

– They are not specialized for one data mining method. In other words, they

are not 'built in' methods for dealing with missing values, and so have high applicability in KDD.
- They can deal with missing values with low cost and as accurate as other methods. Thus they can deal with large datasets that have missing values.
- RCBMM and KMCMM can be applied to unsupervised datasets, because they do not need to use the class attribute.

We carried out an experiment to evaluate NCBMM and RCBMM in the large census dataset consisting missing values. The result show that the proposed algorithms are scalable and achieve a comparable performance in filling up missing values.

Though the obtained results are interesting and very encouraging, it could be considered as an initial work and required to be pursued, refined and verified with a large number of datasets, and with other alternative methods of evaluation.

References

1. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and regression trees. Belmont: Wadsworth, 1984.
2. Friedman, J. H., Khavi, R., Yun, Y.: Lazy Decision Trees. Proceedings of the 13th National Conference on Artificial Intelligence, 717-724, AAAI Pres/MIT Press, 1996.
3. Han, J., Kamber, M.: Data Mining: Concepts and Techniques, Morgan Kaufmann Publishers, 2001.
4. Kononenko, I., Bratko, I., Roskar, E.: Experiments in automatic learning of medical diagnostic rules. Technical Report. Jozef Stefan Institute, Ljubjana, Yogoslavia, 1984.
5. Liu, W.Z., White, A.P., and Thompson S.G., Bramer M.A.: Techniques for Dealing with Missing Values in Classification. In IDAf97, Vol.1280 of Lecture notes, 527-536, 1997.
6. Lobo, O.O., Numao, M.: Ordered Estimation of Missing Values for Propositional Learning. Journal of Japanese Society for Artificial Intelligence, 2000.
7. Mannila, H.: Data mining: machine learning, statistics, and databases. Eight International Conference on Scientific and Statistical Database Management, Stockholm June 18-20, 1996, p. 1-8, 1996.
8. Mantaras, R. L.: A Distance-Based Attribute Selection Measure for Decision Tree Induction. Machine Learning, 6, 81-92, 1991.
9. Pyle, D.: Data Preparation for Data Mining. Morgan Kaufmann Publishers, Inc, 1999.
10. Quinlan, J.R.: Induction of decision trees. Machine Learning, 1, 81-106, 1986.
11. Quinlan, J. R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, Inc, 1993.
12. White, A.P.: Probabilistic induction by dynamic path generation in virtual trees. In Research and Development in Expert Systems III, edited by M.A. Bramer, pp. 35-46. Cambridge: Cambridge University Press, 1987.

This article was processed using the LaTeX macro package with LLNCS style