MVC - A Preprocessing Method to Deal With Missing Values

Arnaud Ragel and Bruno Cremilleux
GREYC - CNRS UPRESA 6072
Universit de Caen
14032 Caen Cedex, France
ragel,cremilleux@info.unicaen.fr

Abstract

Many of analysis tasks have to deal with missing values and have developed specific and internal treatments to guess them. In this paper we present an external method, MVC (Missing Values Completion), to improve performances of completion and also declarativity and interactions with the user for this problem. Such qualities will allow to use it for the data cleaning step of the KDD[1] process[6]. The core of MVC, is the RAR[2] algorithm that we have proposed in [15]. This algorithm extends the concept of association rules[1] for databases with multiple missing values. It allows MVC to be an efficient preprocessing method: in our experiments with the c4.5[13] decision tree program, MVC has permitted to divide, up to two, the error rate in classification, independently of a significant gain of declarativity.
keywords: Association rules, Missing Values, Preprocessing, Decision Trees.

1 Introduction

The missing values problem is an old one for analysis tasks[9] [12]. The waste of data which can result from casewise deletion of missing values, obliges to propose alternatives approaches. A current one is to try to determine these values [10],[4]. However, techniques to guess the missing values must be efficient, otherwise the completion will introduce noise. With the emergence of KDD for industrial databases, where missing values are inevitable, this problem has become a priority task [6] also requiring declarativity and interactivity during treatments. At the present time, treatments are often specific and internal to the methods, and do not offer these qualities. Consequently the missing values problem is still a challenging task of the KDD research agenda [6].

To complete missing values a solution is to use relevant associations between the attributes of the data. The problem is that it is not an easy task to discover relations in data containing missing values. The association rules algorithms [2] [1] are a recent and efficient approach to extract quickly all the associations out of large databases but they are unable, like the majority of analysis methods,

[1]Knowledge Discovery in Databases
[2]Robust Association Rules

to work when missing values are embedded in the data [15]. We have proposed in [15] the RAR algorithm to correct this weakness. It is fully compatible with the core of the fast association rules algorithms [2]. The efficiency of RAR to mine databases containing multiple missing values, allows to use it for the missing values problem. This is what we propose in this paper with the MVC (Missing Values Completion) method. MVC uses the association rules, discovered with RAR, to fill the missing values, independently of any analysis tasks, during a preprocessing, declarative and interactive process with the user. This approach also allows to correct some faults of usual completion methods and so performances of analysis tasks are increased: the error rate of the classification software c4.5[13] can be divided by two in our experiments if MVC is used rather than the missing values treatment of c4.5.

In this paper, we focus on the missing values problem for decision trees but we will see that MVC can be interesting for any analysis tasks. Section 2 briefly reviews usual approaches to deal with missing values in decision trees and points out problems. In Sect. 3 we present the RAR algorithm[15] and show in Sect. 4 its interest for the missing values problem in MVC. In Sect. 5 we present experiments using MVC for classification tasks with the decision tree program c4.5[13] and conclude in Sect. 6, on the possibility with MVC to control the noise that may be introduced during the completion step.

2 Missing Values and Decision Trees

A decision tree is a divide and conquer classification method: it is either a leaf node containing a classification or an attribute test, with a branch to a decision tree for each value of the attribute. The main problem of the missing values in decision tree is to know in which branch we should send a case with an unknown value ? Considering all possible trees, to avoid the missing values problem, is computationally infeasible [7], and deleting data with missing values may cause a huge waste of data. Consequently, a current approach is to fill them [10], [14].

For no having universal and independent method to deal with missing values, many programs used specific and internal treatments that we will briefly review here.

A current approach is to fill the missing values with the most common value. This value can be chosen either in the whole dataset either in datasets constructed for the classification task. For example, if in a dataset the known values for an attribute X are a for 70% of the cases, b for 10% of the cases and c for 20% of the cases, missing values for X will be filled with a. An important critic which can be said is that the common value is chosen in datasets constructed to decide the class variable and not to decide the missing values. Thus such treatments may be irrelevant. There are other variants, like c4.5[13], where missing values are sent in each subset, with a weight proportional to the known values in the subset. In this case, it is equivalent to fill missing values with several weighted values. The precedent critic can be done also for c4.5.

Another technique, proposed by Breiman et al. in [3], is to use a surrogate split to decide the missing value for an attribute. The surrogate attribute is the one with the highest correlation with the original attribute. If this method uses more information than the precedent ones, its efficacy depends on the possibility to find correlations between two attributes. Looking for more associations have been proposed by Quinlan and Shapiro in [11] where they use a decision tree approach to decide the missing values for an attribute. If S is the training set and X_1 an attribute with missing values, the method considers the subset S', of S, with only cases where the attribute X_1 is known. In S' the original class is regarded as another attribute while the value of attribute X_1 becomes the class to be determined. A classification tree is built using S' for predicting the value of attribute X_1 from the other attributes and the class. Then, this tree can be used to fill the missing values. Unfortunately it has been shown that difficulties arise if the same case has missing values on more than one attribute[10]: during the construction of a tree to predict X_1, if a missing value is tested for an attribute X_2 another tree must be constructed to predict X_2 and so on. So this method cannot be used practically. More recently, in 1996, K. Lakshminarayan et. al proposed in [8] a similar approach where they use machine learning techniques (Autoclass[5] and c4.5[13]) to fill missing values. They avoid the problem of multiple missing values on a same data using the internal missing values treatments of c4.5 or Autoclass. Consequently, a declarative and association approach is only used to decide missing values for one attribute: missing values of the others attributes are treated with the missing values treatment of c4.5 or the one of autoclass.

With this brief overview of missing values treatments for decision trees, which can be extended to other analysis methods, we see that the main difficulty is to find declarative associations between data in presence of multiple missing values, which are relevant to guess missing values. In the next section, we present a method which can be used for this task.

3 The Robust Association Rules Algorithm

The RAR algorithm, that we have proposed in [15], corrects a weakness of usual association rules algorithms: a lot of relevant rules are lost by these algorithms when missing values are embedded in databases. We briefly recall principles of association rules and we point out the main characteristics of the RAR algorithm.

3.1 Association Rules

An association rule[1] is an expression of the form $X \longrightarrow Y$, where X and Y are sets of items, and is evaluated by a support and a confidence. Its support is the percentage of data that contains both X and Y. Its confidence is the percentage of data that contains Y among those containing X. For example, the rule *fins=y and tail=y* $\longrightarrow$ *hairs=n* with a support of 13% and a confidence of 92% states

that 13% of animals have fins, tail and no hairs and that 92% of animals with fins and tail have no hairs. The problem of mining rules is to find all rules that satisfy a user-specified minimum support and minimum confidence.
One of the reason of the association rules success is that there are fast algorithms [2] [16] which allow to explore quickly very large databases with a minimum number of pass on data: supports are calculated using an efficient construction of pertinent itemsets of size k from pertinent itemsets of size k-1. Confidences are calculated using valid supports previously found, without new passes on the datasets.

Association rules have been firstly developed to mine transaction databases (basket data analysis). In these databases the missing values problem do not exist in practically, which explain that this problem has not been considered of interest by the fast algorithms. The RAR algorithm corrects this fault, to discover association rules in databases containing multiple missing values.

3.2 The RAR algorithm

In order to avoid the collapse of fast association rules algorithms when a database has missing values, a key point of RAR is to discover rules in valid databases rather than in the whole database. A valid database (vdb) is defined for an itemset X as the largest sub database without missing values for this itemset.

Id	X1	X2	X3	X4
1	?	a	a	c
2	a	a	b	?
3	a	b	c	c
4	a	b	d	c
5	?	b	e	d
6	b	b	f	?
7	b	c	g	d
8	b	c	h	d

VDB 1

Id	X1	X2	X3	X4
1	Disabled			
2	a	a	b	?
3	a	b	c	c
4	a	b	d	c
5	Disabled			
6	b	b	f	?
7	b	c	g	d
8	b	c	h	d

VDB 2

Id	X1	X2	X3	X4
1	?	a	a	c
2	Disabled			
3	a	b	c	c
4	a	b	d	c
5	?	b	e	d
6	Disabled			
7	b	c	g	d
8	b	c	h	d

VDB 3

Figure 1: RAR approach for dealing with missing values

For example in the Fig. 1, if the first left database is the whole database it is also the vdb of the itemsets {X2}, {X3} and {X2, X3} because there is no missing value of X2 and X3. The second one is the vdb of itemsets {X1}, {X1,X2}, {X1,X3}, {X1,X2,X3} and the third one is a vdb for {X4}, {X2,X4}, {X3,X4}, {X2,X3,X4}.

To be fully compatible, without increasing the complexity, with the core of the fast association rules algorithms[2], some modifications for support and confidence calculations have been to be done (see [15] for more information).

If ignoring missing values often leads to a huge waste of data, it is not the case for the RAR algorithm: the association rules approach is an exhaustive research so ignoring missing values does not lead to definitively reject data contrary to methods using heuristics, e.g decision trees. For example, in the Fig. 1, data 1 is not used to evaluate support of $\{X1, X3\}$ but it is used to evaluate support of $\{X2, X4\}$. In a decision tree, if X1 is the root attribute, data 1 would have been rejected, and as it is impossible to construct all the trees (NP-complete problem [7]), this data would be definitively rejected from the analysis task. We have shown in [15] that this approach, which avoid to use a treatment to complete, gives satisfying results for association rules.

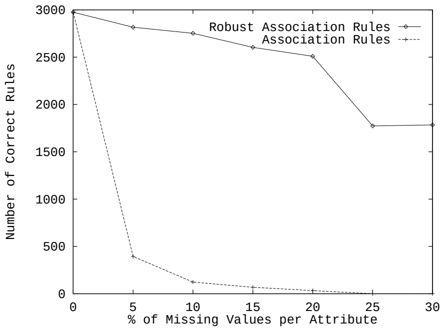

Figure 2: Robust Association Rules Performances

Figure 2 depicts the results of an experiment comparing performances between the RAR algorithm and the main fast algorithm[2]: a reference ruleset is constructed from a database, 2000 data and 8 attributes, with no missing values. Then missing values are randomly introduced with a rate of 5% at first and 30% to finish with an increment of 5% on each attribute of this database. The number of correct rules (include in the reference ruleset) retrieved with the two different approaches are shown by the curves in Fig 2. We see that the number of retrieved rules is clearly larger with RAR, especially when the number of missing values increases.

4 Interest of RAR for Missing Values

4.1 Presentation of MVC

The efficiency of RAR, to find all the association rules quickly in a database with missing values, allows us to propose the MVC method. This method works as in the Fig 3, where it first extracts rules and then uses them to fill missing values, with a possible interaction with the user.

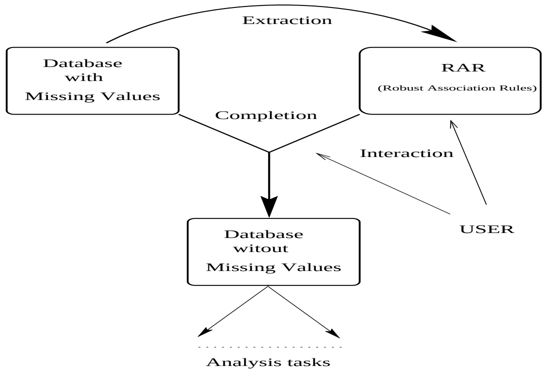

Figure 3: The MVC Process

For the completion, the rules matching a data and concluding on a missing attribute are used to decide its value. Two situations are possible:
1- All the matching rules indicate the same conclusion, then it is used.
2- Several matching rules conclude on different values. In this case the confidence measure and the number of rules concluding on a same value are used to solve the conflict automatically. Of course, the user can intervene. This approach is simple but, in our experiments, there is always a clear preference, in number of rules, for one value rather than the others.

We show below an example of completion for a data from the Zoo database[3].
data: **aquatic=y, tail=y, hairs=?, legs=0, fins=?**
with use of *Rule 1: aquatic=y, and, legs=0 $\longrightarrow$ fins=y (sup[4]: 17%, conf[5]: 97%)*
becomes: **aquatic=y, tail=y, hairs=?, legs=0, fins=y**
with use of *Rule 2: fins=y and tail=y $\longrightarrow$ hairs=n (sup: 13%, conf: 92%)*
becomes: **aquatic=y, tail=y, hairs=n, legs=0, fins=y**

In this example, we can see that recursion is permitted with MVC. However, to avoid series of wrong completions, only rules with a high confidence value (more than 95%) are used for the recursion. With this intuitive treatment the user can easily intervene in the completion process: he can visualize the ruleset, remove, change or add new rules. Furthermore the confidence of a rule can be a good indication of the correctness of the treatment: the rule with a confidence of 97%, in the previous example, indicates that 3% of noise may

[3]UCI Machine Learning Repository, http://www.ics.uci.edu/ mlearn/MLRepository.html
[4]support
[5]confidence

be introduced. It is always possible to find matched rules, to complete, if we allow the use of rules with a low support and a low confidence. This point can appear as a weakness of our approach because a low confidence may implicate noise in data. As a matter of fact, this critic can apply to any method which completes the missing values, like the ones presented in Sect. 2. But, as they cannot give a value like confidence to measure the noise, it does not point out. We will consider this point in our conclusion.

As MVC is a preprocessing method, it comes within the data cleaning step in the KDD process. Another important point is that MVC corrects some weaknesses of usual missing values treatments.

The first one is, as we have seen in Sect. 2, that multiple missing values and computational time prevent usual methods from finding relevant associations to fill missing values. In RAR, the use of the association rules concept, which is designed to explore quickly large databases, allows to construct a ruleset especially for the missing values problem. Thus when a missing value occurs, all the available information of the ruleset is used by MVC to fill it.

Another problem is due to internal treatments, e.g for c4.5 or CART. In this case when a missing value is not correctly filled, the resulting noise will perturbate the treatment of the remaining missing values: if an example E, is sent in a wrong subset because of an incorrect determination of a missing value, all the following subsets will be falsified by the presence of E. Consequently missing values treatments, using information in these subsets, will be falsified too. In MVC such a situation does not appear because rules are discovered in a first step, before any completion. Furthermore during the completion step, the recursion can be controlled.

In the next section we will show that such properties allow MVC to be a declarative and independent treatment but also more efficient.

5 Results

In this section, we try to evaluate the contribution of MVC for classification tasks and compare the ability of treatments (MVC and the internal method of a decision tree program c4.5) to treat missing values. c4.5[13] is one of the most used decision tree program and it can handle missing values by sending them to each subset, with a weight proportional to the known values in the subset. As the missing values treatment of c4.5 is automatic, for scientific exactness, MVC has been used without interaction with the user.

5.1 Introduction of multiple missing values

For this experiment we take a database with no or few missing values, to have a reference database, and we randomly introduce missing values for each attribute (rate 5% to 20% by increment of 5%). Two studies are then realized. The first one is made running MVC and comparing the returned complete database with the reference one. We will see then the capacity of MVC to preprocess data. A second study is made running c4.5 twice on these databases: firstly, using its own treatment to deal with missing values and secondly using MVC to preprocess the missing values. We will compare results in classification using MVC or the missing values treatment of c4.5. Two databases, coming from the UCI Machine Learning Repository has been used for these experiments. The Vote database which has 435 data and 17 attributes and the Credit Card database which have 690 data and 15 attributes[6].

Table 1: Power of Completion with MVC

MV[a]	Vote Database		Credit Card Database	
	Percentage of Completion	Correct Completion	Percentage of Completion	Correct Completion
5%	100%	99.00%	100%	98.56%
10%	100%	98.00%	100%	96.70%
15%	100%	96.70%	100%	95.68%
20%	100%	95.50%	100%	94.13%

[a] Percentage of Missing Values

Tab. 1 gives the results of the first study to show the power of completion of MVC. We can see that MVC has succeed to complete all the missing values, introducing only a low noise rate: for example, in the vote database with 15% of missing values on each attribute, the noise introduced by the completion step is 3.3%.Let us remember that it is impossible to make a such experiment with the c4.5 approach because the completion of a missing value is not unique (a missing values is filled by a distribution of values).

Tab. 2 gives the results of classification with c4.5 using either his own missing values treatment or either MVC. We can see that classification error rate using MVC is lower: it can be divided up to two. Another result is that the more we introduce missing values in the credit card database the better are the results. Such a result may be explained by the fact that some unexpected values for data (e.g noise) may be set to unknown with this experiment and then are filled by MVC with typical ones.

5.2 Real world applications

In this experiment we use 2 databases, from real world, with missing values at origin:

1. OERTC database: This database is used, for classification tasks, in collaboration with the lymphome group of the O.E.R.T.C (European Organization of Research and Treatment of Cancer). We have used the

[6] some with missing values

Table 2: Average Results over ten trials with multiple missing values

MV[a]	Use of MVC	Unpruned Tree Size Tree	Err.[b] (train)	Err. (test)	Pruned Tree Size Tree	Err. (train)	Err. (test)	Err. (predic.)
Vote Database								
(Ref. Result[c])		29.2	1.9%	4.8%	16	2.6%	4.4%	5.7%
5%	no	42.4	2.5%	7.8%	13.9	4.2%	6.7%	7.6%
10%	no	57.7	3.2%	9.2%	10.6	7.1%	8.3%	9.8%
15%	no	51.1	4.1%	8.0%	8.8	9.1%	10.1%	11.8%
20%	no	64.6	4.6%	9.7%	11.2	10.8%	12.9%	14.4%
5%	yes	35.5	2.5%	8.0%	6.1	4.6%	5.5%	6.5%
10%	yes	30.7	2.6%	6.2%	6.1	4.3%	5.3%	6.2%
15%	yes	37.6	2.6%	7.6%	8.8	4.5%	6.0%	6.7%
20%	yes	46.6	2.6%	8.5%	10.0	4.9%	6.7%	7.4%
Credit Card Database								
(Ref. Result)		162.4	7.4%	16.5%	21.9	11.2%	13.2%	14.4%
5%	no	200.8	7.8%	16.8%	13.2	12.4%	13.5%	16.1%
10%	no	190.4	8.8%	17.9%	14.9	13.2%	13.8%	17.4%
15%	no	182.5	9.1%	18.0%	11.4	14.3%	15.1%	19.0%
20%	no	206.8	8.9%	16.8%	15.2	15.9%	17.8%	21.6%
5%	yes	141.5	7.4%	16.5%	23.3	11.3%	14.6%	14.9%
10	yes	134.1	6.5%	13.9%	39.9	9.6%	13.8%	14.6%
15	yes	155.1	6.3%	15.8%	22.1	11.0%	13.9%	14.3%
20	yes	145.6	5.6%	13.9%	15.4	10.0%	12.0%	12.4%

[a] Percentage of Missing Values
[b] Percentage of Errors.
[c] Reference Result

H7 protocol which has 832 cases (1988-1993) and 27 attributes. Eleven of them have missing values with the following proportions: 52%, 43%, 88%, 84%, 39%, 8%, 6%, 7%, 2%, 6% and 6.5%.

2. Auto insurance database: it comes from the UCI Machine Learning Repository and has 205 cases with 25 attributes[7]. Six of them have missing values with the following proportions: 20%, 0.97%, 1.95%, 1.95%, 0.97%, 0.97%, 1.95%.

Tab. 3 gives the results. In the OERTC database results are significatively increased using MVC. An interesting point is that MVC has made to emerge an attribute not used otherwise. With it, qualities of the tree seem to be better (a predicted error of 6.8%). In the Auto database, results are the same but, the low rates of missing values and the few cases, cannot really decide if MVC could be useful. However MVC has completed missing values with an unique value, contrary to c4.5, which is better for the understanding.

[7] numeric attributes have been discretized for this experiment

Table 3: Average Results over ten trials on real world applications with c4.5

Use of MVC	Unpruned Tree Size Tree	Err.[a] (train)	Err. (test)	Pruned Tree Size Tree	Err. (train)	Err. (test)	Err. (predic.)
	OERTC Database						
no	91.4	4.4%	8.5%	41.3	6.2%	7.3%	10.0%
yes	89.7	2.2%	5.6%	40.4	3.5%	6.5%	6.8%
	Autos Database						
no	152.1	4.0%	16.6%	88.3	8.3%	19.5%	30.0%
yes	149.2	4.3%	16.0%	83.4	8.3%	18.5%	29.9%

[a] Percentage of Errors.

5.3 Conclusion on Results

With MVC, results seem to be noticeably improved in classification when there are many missing values. Otherwise, results are at least as good with the preprocessing. Unlike c4.5 approach, another advantage is that the completion is made with only one value which allows the user to understand and react about it.

6 Conclusion

Contrary to previous approaches reviewed in Sect. 2, the ability of RAR to discover associations in missing values databases has permitted to propose an efficient method to fill missing values. The latter, MVC, uses RAR to guess the missing values in databases. Such an approach leads to a significant gain of performances: the error rate in classification, with c4.5[13], can be divided up to two, if MVC is used rather than the classical missing values treatment of c4.5. But the main advantage of MVC is to be a preprocessing method, independent of any analysis tasks, which offers a more understandable treatment of the missing values and a possible interaction with the user. Such qualities, rarely available as far as we know for the missing values problem, may enable MVC to become a tool for the data cleaning step of the KDD process [6].

At the present time, we are working on several points to improve the use of rules in MVC. One of them, is a definition of a new rule score, specially designed for the completion problem: it will combine several criteria rather than only the use of the confidence measure. A second point is to try to determinate an optimum completion threshold: we think that there is a threshold for which the noise, introduced by the missing values completion, could become a more important problem than the remaining missing values. As MVC can evaluate the completion of missing values (using the confidence measure) a such study is possible and should be interesting.

References

[1] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In Proc. of the ACM SIGMOD Conference on Management of Data,p 207-216, Washington, USA, 1993.

[2] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen and A. I. Verkamo. Fast Discovery of Association Rules. In Advances in Knowledge Discovery and Data Mining, p 307-328, MIT Press, 1996.

[3] L. Breiman, J.H Friedman, R.A Olshen, C.J Stone. Classification and Regression Trees, The Wadsworth Statistics/Probability Series, 1984.

[4] G. Celeux. Le traitement des donnes manquantes dans le logiciel SICLA. Technical reports number 102, INRIA, France, December 1988.

[5] P. Cheeseman, J. Kelly, M. Self, J. Stutz, W. Taylor and D. Freeman. Bayesian Classification. In Proc. of American Association of Artificial Intelligence (AAAI), p. 607-611, San Mateo, USA, 1988.

[6] U.M Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From data mining to knowledge discovery: An overview. In Advances in Knowledge Discovery and Data Mining, p. 1-36, MIT Press, 1996.

[7] L. Hyafile and R. Rivest. Constructing optimal binary decision trees is np-complete. Information Processing Letters number 5, p. 15-17, 1976.

[8] K. Lakshminarayan, S.A Harp, R. Goldman and T. Samad. Imputation of missing data using machine learning techniques. In Proc. of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), MIT Press, 1996.

[9] R.J.A Little, D.B Rubin. Statistical Analysis with Missing Data, Wiley series in probability and mathematical statistics, John Wiley and Sons, USA, 1987.

[10] W.Z Liu, A.P White, S.G Thompson and M.A Bramer. Techniques for Dealing with Missing Values in Classification. In Second Int'l Symposium on Intelligent Data Analysis, London, 1997.

[11] J.R Quinlan. Induction of decision trees. Machine learning, Vol 1, p. 81-106, 1986.

[12] J.R Quinlan. Unknown Attribute Values in Induction, in Segre A.M.(ed.), In Proc. of the Sixth Int'l Workshop on Machine Learning, p. 164-168, Morgan Kaufmann, Los Altos, USA, 1989.

[13] J.R Quinlan. C4.5: Programs for Machine Learning, Morgan Kaufmann, San Mateo, USA, 1993.

[14] A. Ragel: Traitement des valeurs manquantes dans les arbres de dcision. Technical reports, Les cahiers du GREYC, number 2, University of Caen, France, 1997.

[15] A. Ragel and B. Crmilleux. Treatment of Missing Values for Association Rules. In Proc. of The Second Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD-98), volume 1394 of Lecture Notes in Artificial Intelligence, p. 258-270, Melbourne, Australia, 1998.

[16] H. Toivonen. Sampling large databases for association rules. In Proc. of the 22nd Int'l Conference on Very Large Databases (VLDB'96), p. 134-145, Morgan Kaufmann, India, 1996. [available on the web from http://www.informatik.uni-trier.de/ ley/db/conf/vldb/Toivonen96.html]