A new reparation method for incomplete data in the context of supervised learning *

Matteo Magnani
Department of Computer Science,
University of Bologna
Mura Anteo Zamboni 7,
40127 Bologna, Italy
matteo.magnani@cs.unibo.it

Danilo Montesi
Department of Mathematics and Informatics,
University of Camerino,
Via Madonna delle Carceri 9,
62032 Camerino MC, Italy
danilo.montesi@unicam.it

Abstract

Real-world data is often incomplete. There exist many statistical methods to deal with missing items. However, they assume data distributions which are difficult to justify in the context of supervised learning. In this paper we propose a new method of repairing incomplete data. This technique is a variation of a general strategy, here called local imputation*. It repairs incomplete records, only when this is reasonable. It is able to identify wrong tuples. It is more general than other similar methods, because of a parametric similarity function. Finally, it also works with noisy data sets.*

1. Introduction

Traditional classification methods usually cannot deal directly with real-world data, because of missing or wrong items. This paper concerns techniques which take an incomplete data set as input, and produce a complete one by filling in values where they are missing. Some methods are very easy to apply, but they have too many drawbacks, which limit their use to uninteresting problems [1, 21]. "Intensive" statistical methods have been mainly developed to manage survey data [3, 18]. They have proven to be very effective in many situations. Unfortunately, they need strong model assumptions, which are usually difficult to justify in Knowledge Discovery in Databases (KDD). There is also a class of techniques, here called *local imputation procedures* (LIPs), that do not assume particular data distributions. Therefore, it is appealing to see if they can be used as preprocessing tasks in KDD [20, 11, 7, 2, 5].

* Partially funded by projects GERONACCESS and "Implementing and background for innovation technology".

We propose a new local imputation method. Our method inherits some interesting properties from other techniques. It repairs incomplete records only when we have sufficient information [5]. It is also able to identify wrong tuples, which are otherwise difficult to discover [11]. Additionally, it is more general than other similar methods, because of a parametric similarity function. This allows the user to specify different approaches when comparing records with missing values. It also works with slightly noisy data sets.

The next section presents existing works on missing data management. Section 3 explains the method and compares the newly proposed technique with five similar algorithms, pointing out advantages and possible drawbacks. Finally, we draw some conclusions and sketch further work.

2. Related works

The problem of missing data is one aspect of a more general topic: data uncertainty [14, 15]. Usually, missing values in databases are represented using one or more special values, i.e., *null*. To analyze data with *null* values, we have to modify traditional analysis techniques, or we must repair the data set.

The effectiveness of a MDT depends on the missing mechanism. When data are *missing completely at random (MCAR)*, the fact that a value is missing is unrelated to its value or to the value of other variables. If the probability that a value is missing depends only on the value of other variables, we say that it is *missing at random (MAR)*. If missingness depends on missing values, data are *not missing at random (NMAR)*, and this is a problem for many statistical MDTs.

When we have to analyze an incomplete data set, we have two possibilities. Either we build a complete one, then we analyze it using traditional techniques. Or we use analysis tools which can internally manage missing data. C4.5 is one of the most known algorithms of the second type [17].

Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'04)
0-7695-2108-8/04 $ 20.00 © 2004 IEEE

This algorithm has proved to be very effective if compared with other methods [6]. For a short overview of tree-based classification procedures see [16] and [12]. The efficiency of C4.5 can be increased by the use of committee learning techniques [24]. If we are interested in guessing statistics, i.e., summary values of the sample, we may also use the EM algorithm [3]. Even if C4.5 and EM have been widely used, we are more interested in complete data methods, for the following reasons. Data collectors may have knowledge of missing data mechanisms, which can be used while building the complete data set. With a complete data set, every existing learning technique can be used. A common starting point allows comparisons of different learning algorithms.

The easiest way to obtain a complete data set from an incomplete one is to erase missing items. These *conventional* methods are used because of their simplicity. They are usually the default option in statistical packages. *Listwise deletion* is the easiest way to obtain a complete data set. However, if we apply it to real situations we may lose a lot of data [10]. Moreover, data must be MCAR. *Pairwise deletion* is a variant of listwise deletion, which uses an incomplete record during analysis only when the needed variable is not missing. If we do not want to lose data and perhaps information, we may try to guess missing items. This process is generally called *imputation*. If we look at recorded values of a variable with missing items, we may choose a measure of central tendency, such as mean, median, mode, trimean or trimmed mean, and use it to fill the holes. In particular, *mean imputation* and *most common attribute value (mode) imputation* have been compared with other MDTs [22, 6, 8]. *Non-deterministic versions* of these methods are also possible, which add a random disturbance to the mean. Another variation of this approach consists in *assigning all possible values* to a missing item, i.e., all the values that the missing variable takes in other records. However, this is computationally unfeasible [6]. If there are correlations between missing and non-missing variables, we may learn these relationships and use them to predict missing values. One such strategy is *imputation by regression*. A more sophisticated imputation technique is *multiple imputation* [18, 19]. *Hot Deck imputation* is a procedure where imputations come from records in the same data [4, 20]. These are chosen on the basis of the incomplete record. Several variations of Hot Deck imputation have been developed, in particular [11, 7, 2, 5]. We call these techniques *local imputation procedures (LIPs)*.

3. Local imputation method

A local imputation algorithm groups objects in homogeneous classes. Then, for every missing item, it extracts a value from records in the same class. In particular, our method is listed in Table 1. In this code, n refers to the number of records and m to the number of attributes, while "?" represents a missing value. Exit code -1 indicates that we have no sufficient information to guess an unidentified concept. Exit code -2 means that we have not identified a concept, or that the imputed attribute is not significant. The method is based on the following assumptions and definitions. A data set represents objects. Objects belong to concepts. Concepts are hard (they have a characteristic function), and they are organized in a hierarchy of fuzzy classes (membership functions). A concept has a unique description based on its relevant attributes. Some errors in the data set do not change the concept structure in a significant way. All these considerations form the basis on which our method should work.

```
INPUT: incomplete data set (x_1,...,x_n),
       weights w, tolerance t
OUTPUT: a partially repaired data set
 1) For i in [1..n-1], j in [i..n]
 2)    D[i][j] := (sum_{k=1}^{m} w_k · similarity(x_ik, x_jk)) / (sum_{k=1}^{m} w_k)
 3) CLUSTER(D, 'hard', 'partition')
 4) For every missing value x_pk
 5) BEGIN
 6)    If ( |{ i : (x_i ∈ [x_p]) ∧ (x_ik ≠ ?)}| / |[x_p]| ≤ t)
 7)       then exit(-1)
 8)    If (A_k is categorical) then
 9)       If (∃ V ∈Dom(A_k) ( |{i:x_i ∈ [x_p] ∧ x_ik = V }| / |[x_p]| )
          ≥ (1 − t)))
10)            then x_pk := V
11)       Else exit(-2)
12)    Else if (A_k is numeric) then
13)       x_pk := mean_trimmed(1 − t)
14) END
```

Table 1. Local imputation method.

While we explain its behavior, we also compare it with existing similar algorithms: traditional Hot Deck (described in [20]), Grzymala-Busse et al. [7], Lee et al. [11], Batista and Monard [2], and Gediga and Düntsch [5].

3.1. Input data set

Lines 1 and 2 of Table 1 calculate syntactic distances between objects. It is worth noting that all records are compared and clustered. Traditional Hot Deck uses records which are complete on pre-defined attributes (auxiliary variables). Other approaches only use complete records [11]. In this way, we risk losing important information. In general, only classified records are used in training sets, so it seems reasonable to only repair them. In our opinion, also unclassified records may contribute to data reparation. In

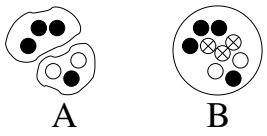

Figure 1. All records carry topological information.

fact, even if class information is absent, objects carry intrinsic topological information, which can be useful in identifying concepts. This is illustrated in Figure 1. Even when records do not carry classification information, they provide topological information. If we do not use unclassified records, we risk losing the concept structure. In this figure, part A shows two discovered concepts, while in part B we can see how they are clearly a unique concept, if we also show unclassified records.

3.2. Looking for concepts

In line 3 of Table 1, a clustering process is used. There is a huge number of clustering algorithms [9], and we do not suggest a particular one. However, the result must have a structure that can match concepts. Therefore, classes must be hard[1]. The number of concepts is not known in advance. Every class may have a different size. For example, in a first-aid station we find that some diseases are more frequent than others.

Similar algorithms use approaches which are not satisfactory. Classes are usually fuzzy [2, 5], and we have a class for every record. Sometimes classes have a fixed number of elements [2].

3.3. Similarity function

The clustering algorithm uses an internal distance (or similarity) function. We may write it as [13]:

$$d(x_i, x_j) = f(x_i, x_j, \mathcal{C}, \mathcal{E})$$

This function depends on the context $\mathcal{C}$[2], on the concepts $\mathcal{E}$ to be found, and on the syntactic distance between the records. Context and concepts are used to add background knowledge to the clustering procedure.

Hot Deck is not particularly sensitive to any kind of distance function [20]. However, our function must consider incomplete records. Different interpretations of missing data may lead to quite different results. We propose the following syntactic similarity function:

1 Every object belongs to only one concept.

2 That is, other records in the data set. For example, in [11] weights depend on the variability of the attribute values.

#	Records	G.-B.	Lee	Ged.	Wang	New
1	(a,b) (c,d)	0	0	0	0	0
2	(a,b) (?,d)	0	0.5	0	0	0.25
3	(a,?) (?,d)	0	1	1	0	0.5
4	(a,b) (a,d)	0.5	0.5	0	0	0.5
5	(a,b) (a,?)	0.5	1	1	1	0.75
6	(a,b) (a,b)	1	1	1	1	1
7	(?,?) (?,?)	0	1	1	1	0.5

Table 2. Comparison of similarity functions on significant examples. Original functions have been normalized.

$$f(x_{ik}, x_{jk}) = \frac{\sum_{k=1}^{m} w_k \cdot \mathrm{sim}(x_{ik}, x_{jk})}{\sum_{k=1}^{m} w_k} \qquad (1)$$

$$\mathrm{sim}(x_{ik}, x_{jk}) = \begin{cases} 0 & \text{(symbolic attribute)} \\ & \text{if } (x_{ik} \neq x_{jk}) \\ q_k & \\ & \text{if } (x_{ik} = \,?\,) \vee (x_{jk} = \,?\,) \\ 1 & \\ & \text{if } (x_{ik} = x_{jk}) \\ 1 - \frac{|x_{ik} - x_{jk}|}{|\max_{\in[1,n]}(x_{pk}) - \min_{\in[1,n]}(x_{pk})|} & \\ & \text{o.w.} \end{cases}$$

The parameter **q** allows us to give more or less restrictive interpretations of missing values.

In Table 2 we have compared our function with the ones used in five similar methods. It is clear that there are two approaches. The first (G.-Busse) is pessimistic, because it considers missing values as wrong. The second (Gediga) is optimistic, because it considers missing values as right. [23] and [11] are variations of these two interpretations. Both these assumptions may be used in some situations, but in general they are not satisfactory. With the pessimistic assumption, entries 1 and 3 of Table 2 have the same score, even if case 1 compares two different records, while case 3 compares two possibly equal records. Under the optimistic assumption, entry 7 has the same score as entry 6. In the first case, we know nothing about the two records. In the second, we know they are identical.

Notice that equation 1 reduces to the pessimistic function with $q_k = 0$, and to the optimistic function with $q_k = 1$. In Table 2 we have $q_k = 0.5$. In this way, entries 1 and 6 of the table have similarity 0 and 1, as with all the other functions. Records in entry 2 are more similar than those in entry 1, because we are not sure that second values are different. But they are less similar than those of entry 4, where we are sure they are equal. The parameter q can be used to impose different interpretation degrees. For example, if

we know that the domain of an attribute has cardinality $\#a$, we may use $q_a = \frac{1}{\#a}$, which is the probability that a randomly chosen value is equal to the recorded one.

3.4. Grouping variables

As we can see in function 1, all feature variables are used to discriminate between concepts ($\sum_{k=1}^{m}$).

In traditional LIPs, i.e., Hot Deck, only a complete subset of the attributes is used to compute classes. These *auxiliary variables* should be correlated or linked by some dependencies (edits) with missing fields, or they should be stratification variables. The number of auxiliary variables is never too high. This is done to obtain classes of a reasonable size [20]. This is not a good approach. We would like to obtain classes with few elements, when they are outliers or objects for which an imputation is not possible. Assume there is a record which is far from all the others. This can be due to a wrong value. With a clustering algorithm which identifies classes of different sizes, we may obtain a concept with only that object. This can be presented to the user to evaluate its correctness. It is worth noting that the same result may not be obtained by looking for outliers inside variables. Here, also objects with reasonable values, but strange combinations of them, can be discovered [11].

As a final consideration, sometimes highly predictive variables may be incomplete, and in this case traditional procedures would not use them. Additionally, it is not always possible to find variables which are complete over the whole data set. In this case, we may lose part of the records.

3.5. Dealing with noisy data sets

To cope with noisy data we suggest the introduction of a tolerance parameter t. This is used in lines 6 and 9 of Table 1. When we impute a value, this means that at most a percentage of data less than the tolerance does not support that imputation. This is useful to avoid overfitting.

3.6. Imputation strategy

When we have to choose values to be imputed, we must consider the model assumptions (Section 3, first paragraph). Mode is a good choice, but we must pay attention: concepts have unique decriptions. This means that if we find that not all records (given the tolerance) have the same value in that group, either this is not a concept or that attribute is not one of the characteristic attributes of the concept (Table 1, line 9). In this case we cannot impute.

When missing attributes are numeric, and if we are inside a concept, this means that any value inside the concept range is good. Therefore, in this context the mean seems to be a good choice. However, as with categorical attributes, there may be wrong values. Therefore, we prefer a measure of central tendency which is less sensible to outliers. We may use a mean trimmed $100(1-t)\%$, where t is the tolerance previously introduced. For instance, if we have a tolerance $t = 0.03$, lower and higher 1.5% of scores are discarded, then the arithmetic mean is evaluated.

3.7. Reuse of imputations

With traditional Hot Deck procedures it is not always possible to do imputation in a single pass. This happens because the Hot Deck is the set of complete records. If this set is small, we may lose a lot of information. In this case, reiterations of the algorithm are used to not waste data.

The case is different if the algorithm is repeated to use previous imputations as new information [5]. When a value has been imputed, this can be used to repair other records. It is not easy to say if this is a clever idea. Using an imputed value as if it were recorded increases the uncertainty of the final data set. Further studies should investigate benefits and drawbacks of this approach.

When the problem is the imputation of missing class attributes, the situation is slightly different. Usually, when feature attributes are missing, we try to impute them, while when decision attributes are missing, we classify the records. In reality, this distinction is not so hard. In both cases we are using information from the data set to add a value. The main difference is that feature attributes do have a precise value, while decision attributes may be intrinsically uncertain, due to multiple contributions to the data set. To identify the nature of missing decision attributes, we may look at the behavior of other records in the same concept. Again, if we do not want to lose the concept structure, we must use all available information at the same time.

For instance, in Figure 1(B), we may say that the behavior of the complete part of the concept is (black:$\frac{4}{6}$, white: $\frac{2}{6}$). If data are MAR (MCAR inside the concept), we may assign to the unclassified records these class membership functions. If we do not consider the global information given by the concept structure, and we classify objects one at a time, we may obtain unsatisfactory results, as in Figure 2. Unclassified records are classified (using a nearest-neighbor approach), and then added to the classifier. Levels A and B show how a different order in the choice of the records may lead to completely different scenarios.

4. Final remarks

In this paper we have presented a new data reparation method for classification data sets. This method inherits important capabilities from other techniques. It is able to identify a class of wrong records which is otherwise difficult to

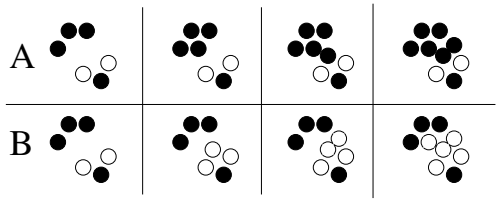

Figure 2. Two possible nearest-neighbor classification outcomes.

discover. It is able to decide if imputation is justified by the available information, and, if not, it leaves the record unchanged.

Moreover, our technique has new interesting features. It uses all available information, including unclassified records, to discover the data structure. It works with a parametric similarity function, which is more general then existing ones, and may be adjusted to different users and situations. It still works on noisy data, managing spread records and wrong values.

We have purposely decided to perform a theoretical analysis, because a simulation study would be useless without a clear understanding of the application context. In fact, it may happen that the tested algorithm works well because of contingent data sets. Now that we have identified a procedure which is theoretically well-founded, further research is needed to verify the effectiveness of our results on real data sets. We will also investigate the use of hierarchical clustering, to distinguish between classes and concepts.

References

[1] P. D. Allison. *Missing data*. Sage Publications, Inc, 2001.

[2] G. E. A. P. A. Batista and M. C. Monard. An analysis of four missing data treatment methods for supervised learning. *Applied Artificial Intelligence*, 17(5-6):519–533, 2003.

[3] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum-likelihood from incomplete data via the EM algorithm. *Journal of The Royal Statistical Society Series B*, 39(1):1–38, 1977.

[4] B. L. Ford. An overview of hot-deck procedures. In *Incomplete data in sample surveys*, pages 185–207. Academic Press, Inc., 1983.

[5] G. Gediga and I. Düntsch. Maximum consistency of incomplete data via non-invasive imputation. *Artificial intelligence Review*, 19:93–107, 2003.

[6] J. Grzymala-Busse and M. Hu. A comparison of several approaches to missing values in data mining. In W. Ziarko and Y. Y. Yao, editors, *Rough Sets and Current Trends in Computing*, volume 2005 of *Lecture Notes in Computer Science*. Springer, 2001.

[7] J. W. Grzymala-Busse, W. J. Grzymala-Busse, and L. K. Goodwin. A comparison of three closest fit approaches to missing attribute values in preterm burth data. *International journal of intelligent systems*, 17:125–134, 2002.

[8] M. Hu, S. M. Salvucci, and M. P. Cohen. Evaluation of some popular imputation algorithms. In *Section on Survey Research Methods*, pages 308–313. American Statistical Association, 2000.

[9] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323, 1999.

[10] K. Lakshminarayan, S. A. Harp, and T. Samad. Imputation of missing data in industrial databases. *Applied Intelligence*, 11:259–275, November 1999.

[11] R. C. T. Lee, J. R. Slagle, and C. T. Mong. Towards automatic auditing of records. *IEEE Transactions on Software Engineering*, 4(5):441–448, 1978.

[12] W. Z. Liu, A. P. White, S. G. Thompson, and M. A. Bramer. Techniques for dealing with missing values in classification. *Lecture Notes in Computer Science*, 1280:527–536, 1997.

[13] R. S. Michalski and R. Stepp. Learnin from observation: conceptual clustering. In R. S. Michalski, J. Carbonell, and T. Mitchell, editors, *Machine Learning: An Artificial Intelligence Approach*, pages 331–363. Tioga publishing Co., Palo Alto, 1983.

[14] A. Motro. Management of uncertainty in database systems. In W. Kim, editor, *Modern database systems: the object model, iteroperability, and beyond*, chapter 22, pages 457–476. ACM Press, 1995.

[15] S. Parsons. Current approaches to handling imperfect information in data and knowledge bases. *IEEE Transactions on Knowledge and Data Engineering*, 8(3):353–372, 1996.

[16] J. R. Quinlan. Unknown attribute values in induction. In B. Spatz, editor, *Proceedings of the Sixth International Workshop on Machine Learning*, pages 164–168. Morgan Kaufmann, 1989.

[17] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[18] D. B. Rubin. Multiple imputation for nonresponse in surveys. John Wiley and Sons, 1987.

[19] D. B. Rubin. An overview of multiple imputation. In *Survey Research Section*, pages 79–84. American Statistical Association, 1988.

[20] I. G. Sande. Hot-deck imputation procedures. In *Incomplete Data in Sample Surveys*, volume 3, pages 339–349, 1983.

[21] J. Schafer. *Analysis of incomplete multivariate data*. Chapman Hall, 1997.

[22] J. Scheffer. Dealing with missing data. *Res. Lett. Inf. Math. Sci.*, 3:153–160, 2002.

[23] G. Wang. Extension of rough set under incomplete information systems, 2002. available at: citeseer.nj.nec.com/526828.html.

[24] Z. Zheng and B. T. Low. Classifying unseen cases with many missing values. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 370–374, 1999. Tech Report (TR C99/02) (available at http://www3.cm.deakin.edu.au/∼zijian/Papers/comm-missing-trC99-02.ps.gz).