Preprocesing Missing Attribute Values to Machine Learning [⋆]

Inmaculada Fortes[1], Gonzalo Ramos-Jimenez[2], and Manuel Baena[3]

[1] E.T.S.I. Informatica, Dept. Matematica Aplicada, Univ. Malaga, Campus Teatinos, 29071 Malaga (Spain), ifortes@ctima.uma.es

[2] E.T.S.I. Informatica, Dept. Lenguajes y Ciencias de la Computacion, Univ. Malaga, Campus Teatinos, 29071 Malaga (Spain), ramos@lcc.uma.es

[3] E.T.S.I. Informatica, Studient, Univ. Malaga, Campus Teatinos, 29071 Malaga (Spain), manuelbaena@airtel.net

Abstract. In this paper a new approach for working with missing attribute values in inductive learning algorithms is introduced. The goal is the "a priori" assignation of values to missing attributes values. We present a method that makes a preprocess of the data-set for allow the learning algorithms use, that don't stand working with missing values, on data-set with missing values. We define an attribute order selection criterion and two methods to making the preprocessing. Experimental results are shown.

Key words: missing data, data preprocessing, machine learning, data mining

1 Introduction

Inductive learning is made from a set of training cases. The success of this learning depends on the quality of the training cases. The basic learning algorithm assume that the attribute values for all training cases of the training set are known. If the training set contains missing attribute values, we must decide how to work with them. The normally decision could be to discard the training cases with missing attribute values (ID3); other decision is to adapt the algorithms to include them (C45, PostP)[9, 2]; and other ones can be preprocess the data-set to fill this missing values.

If the training cases with missing attribute values are discarded, we would decrease the ability to find patterns, and we would obtain worse predictions because fewer training cases are taken into consideration.

The algorithms adapted to work with missing attribute take a value for fill the missing values in the process of classification.

Our decision, and the main goal of this paper, is making a filter to preprocess the data to fill missing values and permit applying learning algorithm, that don't

[⋆] This work has been partially supported by FRESCO project number PB98-0937-C04 of the CICYT-Spain.

support working with missing attribute values, over any data-set. Our approach consist on fill missing values using maximum possible information. Before an analysis to determinate the order of attribute to fill their missing values.

We now review the treatment of missing values: from the inductive learning point of view, and from the statistical data analysis point of view.

From the first point of view, there are several approaches to the problem of assigning values and these are empirically compared in [8]. The basic conclusion is that no one approach is consistently superior to the other. There are two common approaches: filling in missing attribute values with a value or filling it in using a probability distribution of the attribute values. In the first approach the value is obtained from the training set in the current node, e.g., the most probable value [5]. The second approach is adopted in C4.5 and C5 [9] by following ASSISTANT86 [3]. Both approaches only consider the values of the attribute (they do not consider the classes).

From the statistical data analysis point of view, missing data have been mainly studied by Little and Rubin [7]. In Little and Rubin's approach, a previous study of the data distribution is carried out to estimate the parameters of the distribution. Then the missing values are replaced by estimated values. Subsequently, the parameters are estimated again and so on, and the iteration carries on until convergence. The same approach considered in [7] is considered by using Bayesian networks [6]. In this work a simpler instance is studied: unrestricted multinomial distributions. The analysis of distribution data, in both cases, is mixed with the estimation of missing values.

Other approach is based on considering the values of the attributes and the class too. The analysis of data distribution is a previous step to the estimation of missing values. It's Adopted in PostP [2]

2 Our Approach

In this section we develop the ideas contributed in this work: the definition of a general attribute order selection criterion and the assignation of values to missing attribute values taking into account more information (all know or all possible).

We assume that the unknown values (in the training cases or in the observations) have been lost in a random manner. For selecting an attribute in the filling process a general order criterion will be defined. First, the missing value is taken as a new value of the attribute. Second, the order criterion takes into account this new value. Finally, a reduced weight is assigned to the attribute with less known information.

Let us suppose that an attribute has been selected. Then, a classifier (decision tree for example) for this attribute is generated before a pre-election of the data to train the classifier.

The pre-election of the training cases could be: full experiences only, attribute with all known values, all attributes (including attributes with missing values).

The classifier generated is used to predict the value of all missing values in the attribute.

The rest of the paper is organized as follows. The next section introduces the notation used and preliminary concepts. The fourth section is devoted to explain how to make the preprocessor to work with missing attribute values. Finally, experimental results are presented along with conclusions and future work.

3 Notation and Previous Concepts

In this section we provide basic concepts.

Definition 1. *An attribute X_i is a set of values that cover a domain (continuous or discrete). It represents a character of an element.*

Definition 2. *The domain D of an attribute is the set of values that the attribute can take. The value "?" represents a missing value. We name D_i to the domain of the attribute X_i.*

Definition 3. *An experience is represented by the tuple $e = (X_1(e), X_2(e), \ldots, X_f(e))$ where $X_i(e)$ is the value of the attribute X_i evaluated in e (We don't consider a class).*

Definition 4. *It's defined experience universe U_E as the cartesian product $U_E = D_1 \times \ldots \times D_n$ of all the attribute domains. It represent every possible experiences (object) of the problem universe.*

Definition 5. *A training set T is a set of experiences of the same universe.*

Definition 6. *A test set TS is a set of experiences of the same universe. An attribute X_i is unknown for all experiences in TS.*

4 Making Preprocess of Missing Attribute Values

In this section, the core of the paper is developed; i.e., a new approach to work with missing attribute values to machine learning is presented.

The process of filling missing values consists basically in generating a classifier for each attribute with missing values in the data-set. First, we define the attribute order selection criterion. Second, we explain how to assign values to the training cases with missing attribute values.

4.1 Attribute order selection criterion

The process of fill missing values is made incrementally. In other words, each missing value filled in an attribute influences in value that the next attribute will take, locally (in the same experience) and globally (in measure in which the classifier does it). By this way, the order in which we select the attribute for fill their missing values influences in the quality of the resulted data-set.

As it is known, when more information we have of a problem more efficiently we can construct its model. By this way, the attribute order selection criterion is basically catching the first attribute which has the most possible information known. This isn't ever catching attribute which has less missing values.

We define a function to measure the index of unknowledge of an attribute. We select the order to fill missing values using this function.

Definition 7. *Let an experience set T, an attribute X_i we define $M_i = |\{e \in T / X_i(e) = ?\}|$.*

Definition 8. *Let an experience set T with attributes $\{X_1, X_2, \ldots, X_n\}$ we define $M_{i,j} = |\{e \in T / X_i(e) = ? \wedge X_j(e) \neq ? \wedge i \neq j\}|$.*

Definition 9. *We defined index of unknowledge of an attribute X_j respect of an experience set T as $IU_j = M_j \times max(\{M_{k,j} / k = 1, \ldots, n \wedge k \neq j\})$.*

Attribute order selection criterion is used over attribute with missing values. It consists on catching in each iteration the attribute X_k to be filled with minimum index of unknowledge: $min(\{I_k / k \in \{1, \ldots, n\}\})$. If two attributes have the same index of unknowledge we catch the one with less missing values.

The distribution of missing values has less importance in the experiences set as more efficiently distribute over attribute are these missing values. This criterion is a measurement of the distribution over attributes of missing values in an experience set.

Example 1. This simple example taken from [9] will be used throughout the paper. Some climatological characteristics are observed, then we decide to play or not to play. Let X_1 be the outlook, X_2 the temperature, X_3 the humidity, X_4 be windy, X_5 the election of playing or don't.

In the table 1 we show the training set T.

$IU_2 = 0; IU_3 = 4$. By the attribute order selection criterion we select first the attribute X_2 to fill their missing values and second the attribute X_3.

4.2 Preprocessing data

We implement two systems. The first one to work with classifiers that don't support to work with missing values. And another one to work with classifiers that support to work with missing values attributes.

Method 1 About this method it's detached that it can be implemented using learning algorithms that don't support working with missing values (ID3).

Let an experience set T with attributes $(X_1, X_2, \ldots, X_n)$.

In first place, we apply the attribute order selection criterion over the set of attributes with missing values to obtain the correctly order to fill missing values.

Once it's made, we do for each attribute X_k in the established order:

Table 1. The training set T

	$X_1(e)$	$X_2(e)$	$X_3(e)$	$X_4(e)$	$X_5(e)$
e_1	sunny	75	70	yes	Play
e_2	sunny	?	?	yes	Don't Play
e_3	sunny	85	85	no	Don't Play
e_4	sunny	?	?	no	Don't Play
e_5	sunny	69	70	no	Play
e_6	overcast	?	90	yes	Play
e_7	overcast	83	78	no	Play
e_8	overcast	?	65	yes	Play
e_9	overcast	81	75	no	Play
e_{10}	rain	71	80	yes	Don't Play
e_{11}	rain	?	70	yes	Don't Play
e_{12}	rain	75	80	no	Play
e_{13}	rain	68	80	no	Play
e_{14}	rain	?	?	no	Play

T is copied in T' and all attributes (X_k is excluded) with missing values is eliminated in T' (It's supposed T has at last one attribute without missing values).

T' is divided in two subsets: a training set TRS with the experiences that have attribute X_k defined, and a test set TS with the experiences that have attribute X_k with missing value.

We construct a classifier using a learning algorithm over the set TRS. And finally, we use this classifier to predict the value of the attribute X_k for each experience in TS. These values are updated in the initial set T.

Method 2 In this method we can use only learning algorithms that support making classifiers over data-set with missing values (C45, PostP).

Let an experience set T with attributes $(X_1, X_2, \ldots, X_n)$.

In first place, we apply the attribute order selection criterion over the set of attributes with missing values to obtain the correctly order to fill missing values.

Once it's made, we do for each attribute X_k in the established order:

T is divided in two subsets: a training set TRS with the experiences that have attribute X_k defined, and a test set TS with the experiences that have attribute X_k with missing value.

We construct a classifier using a learning algorithm over the set TRS. And finally, we use this classifier to predict the value of the attribute X_k for each experience in TS. These values are updated in the initial set T.

5 Experimental Results

We have implemented our algorithm and used standard data-sets to obtain experimental results. We have collected the usual and known data-sets from Holte

[4]. This set of problems aims at representing all real-life problems. Holte claimed that the experimental results obtained for these problems by learning systems could shed light on the performance of each learning system in a real situation. All the selected problems comply to two conditions: representing a real-life problem that has not been constructed artificially; and the examples are described by means of attributes used naturally in real life. The only data-set which does comply with the second condition is CH. This data-set represents the endgames in chess. According to Holte this data-set is designed to fit well in Quinlan's ID3.

5.1 Description of Datasets

There are several versions of Holte's problems. Basically, there are changes in the number of training cases used or in the number of attributes used. Here we have used the version stored in the UCI Repository of Automated Learning Repository in California University, Irvine [1].

In the table 2 we present a brief summary of the characteristics of these data-sets: the first column shows the code of the data-set, the second column shows the number of training cases, the third column includes the number of missing values, the fourth column shows the percentage of occurrence of the most frequent value of the class, the fifth and sixth columns show the number of continuous and discrete attributes, respectively, and the last column shows the number of different values for the class. For the experiment, two algorithms

Table 2. DataSets

Prob	Size	Missing	Ex.maj.class	Contin	Nom	N.Class
BC	286	9	70.3	0	9	2
CH	3196	0	52.2	0	36	2
GL	214	0	35.3	9	0	7
G2	163	0	53.4	9	0	2
HD	303	7	54.5	5	8	2
HE	155	167	79.4	6	13	2
HO	368	1927	63.0	7	15	2
HY	3163	5329	95.2	7	18	2
IR	150	0	33.3	4	0	3
LA	57	326	64.9	8	8	2
LY	148	0	56.7	2	16	4
MU	8124	2480	51.8	0	22	2
SE	3163	5329	90.7	7	18	2
SO	47	0	36.2	0	35	4
VO	435	381	61.4	0	16	2
V1	435	392	61.4	0	16	2

were implemented to build decision trees. In the first algorithm, the training cases with missing values are distributed following the approach of C4.5; that is, the probability distribution function of the attribute under consideration is used to distribute the cases with missing values in this attribute. In the second algorithm, the cases with missing values are distributed taking into account the probability distribution obtained using the attribute and the class.

Before using the data-sets we apply a filter to make discrete continuous attributes. During the first stage, the original data-sets were used; subsequently, we randomly introduced a percentage of missing values in the original training cases (20, 30, and 40 per cent, respectively). We make a cross-validation to perform the test. Therefore, 40 experiments were done for each data-set. For the 40 estimated decision trees, the correction of classification has been calculated. For each group of decision trees belonging to the same percentage of missing values, mean hit and standard deviations were calculated.

5.2 Results

The experiment were done with the learning algorithm C4.5. The results obtained are shown in the table 3. The first column shows the data-set code. In columns 2, 3, and 4, the results obtained for the data-set with 20, 30, and 40 % of missing values, respectively, are shown. For each percentage, it's shown the result of apply directly C4.5, of apply Method 1 with C4.5 and of apply Method 2 with C4.5. For each data-set, the correct percent of the experiments is shown in the first row, and the standard deviation of this correct percent appears in the second row. Although the results are similar in experiments, we may pay attention that in the first case (applying C4.5 directly) the algorithm use a probability distribution to determine missing values which is more efficient that assigning a value. We must have in consideration that our intention is making a preprocessing of data-set to let us use learning algorithms that don't support working with missing values.

6 Conclusions and Future Work

A new approach to working with missing attribute values has been presented. An attribute order selection criterion has been defined to work with missing attribute values. We have developed a new approach to tackle the problem of assigning values to unknown values of an attribute from a preprocessing point of view. This approach takes into account more information (as more as possible) than C4.5 and other algorithms that work with missing values.

The advantage of preprocessing is that one time we have done it we can apply any learning algorithm, include those which aren't prepared for working with missing attribute values. Also, it can be interesting to dispose of data-set without missing values to another utilities.

We want to detach the importance in our methods about the attributes selection order to complete missing values. It can be experimentally proved that this order decisively influences in the filled goodness missing values.

Table 3. Results

Code	Initial C4.5-M1-M2	20% Missing C4.5-M1-M2	30% Missing C4.5-M1-M2	40% Missing C4.5-M1-M2
BC	75.18-75.53-75.53	69.86-70.3-72.01	70.32-72.48-70.7	70.3-73.51-71.38
	6.37-4.6-4.6	7.18-8.51-7.37	4.1-8.29-6.66	1.43-7.08-5.61
CH	98.81-98.81-98.81	87.3-87.39-88.05	84.6-84.7-84.79	78.31-78.41-78.16
	0.53-0.53-0.53	1.39-1.64-1.5	1.45-1.23-1.61	2.52-1.99-2.27
G2	75.99-75.99-75.99	69.85-71.03-69.85	68.01-71.65-69.82	69.82-69.82-69.19
	12.39-12.39-12.39	9.73-10.92-10.17	7.95-9.64-10.63	8.85-10.22-10.06
GL	63.46-63.46-63.46	49.85-54.11-56.93	50.06-53.7-53.66	46.67-46.54-51.28
	9.12-9.12-9.12	10.32-6.44-6.67	7.01-11.07-6.87	9.75-13.29-7.52
HD	82.15-81.85-81.85	74.58-74.91-76.26	71.56-73.56-75.88	71.6-75.59-75.57
	6.53-6.06-6.06	6.62-9.05-8.02	8.3-8.82-8.92	7.28-5.83-6.9
HE	83.29-82.62-81.96	80.08-79.38-80.08	76.17-80.54-81.29	78.71-80.04-78.67
	7.45-8.06-7.31	9.04-7.17-9.11	5.1-7.23-7.8	3.03-9.42-8.33
HO	85.88-83.98-84.52	85.88-83.98-84.52	83.99-83.18-82.63	80.98-79.63-78.53
	6.45-6.52-6.6	6.45-6.52-6.6	6.26-8.77-7.63	4.41-4.04-6.18
HY	97.5-97.5-97.5	96.74-96.96-97	97-96.46-96.4	95.26-96.3-96.4
	0.31-0.31-0.31	0.6-0.56-0.58	0.54-0.47-0.48	0.01-0.79-0.58
IR	94.67-94.67-94.67	92-83.33-83.33	83.33-74-78	83.33-70.67-72.67
	6.89-6.89-6.89	5.26-6.48-6.48	5.67-8.58-10.45	9.03-13.77-14.21
LA	84-82.67-84.33	84-82.67-84.33	84-82.67-84.33	82-83.67-80.67
	5.84-13.68-14.74	5.84-13.68-14.74	5.84-13.68-14.74	9.58-17.39-15.46
LY	78.95-78.95-78.95	79.81-80.43-74.33	72.38-73.67-74.33	62.14-68.14-68.9
	8.42-8.42-8.42	8.15-5.77-12.58	10.5-5.74-9.45	8.06-11.26-9.19
MU	100-100-100	98.78-94.08-92.92	97-89.67-89.62	95.52-86.57-87.14
	0-0-0	0.38-2.07-1.75	0.71-3.06-2.23	0.92-2.05-3.67
SE	96.05-96.02-96.02	94.97-95.1-95.04	93.71-94.63-94.56	90.74-93.52-93.39
	1.24-1.29-1.19	0.92-0.73-1.06	0.77-1.1-0.85	0.14-1.22-0.91
SO	98-98-98	89-87.5-87.5	78-81-81	74.5-66-62
	6.32-6.32-6.32	11.74-14.39-14.39	16.87-15.06-15.06	21.66-19.41-20.3
V1	91.5-90.8-89.2	90.57-88.74-89.21	87.6-84.38-81.81	83.21-80.92-82.54
	4.45-5.2-4.48	3.95-3.95-4.65	5.5-5.49-5.83	5.83-4.93-5.8
VO	96.78-97-97	93.78-90.1-90.55	90.14-86.21-86.21	85.04-80.9-80.66
	3.29-2.89-2.89	4.75-5.39-5.42	5.46-5.12-4.89	4.61-7.05-7.41

The results obtained show us that its a valid method. Normally, it gives better results as more missing values we have.

We'll have carried out some experiments with other learning algorithm such the IADEM [10] and CIDIM method [11].

Future research based on the inductive learning approach will take these aspects. To integrate the attribute order selection criterion into a classifier and we will carry out more experiments.

References

1. Blake, C., Keogh, E., Merz., C.J.: UCI repository of machine learning databases. University of California, Irvine, Dept. of Information and Computer Sciences, 1998. http://www.ics.uci.edu/mlearn/MLRepository.html.
2. Fortes, I., Morales-Bueno, R., Mora, Ll., Triguero, F.: A Decision Theory Approach to work with Missing Attribute Values in Inductive Learning Algorithms. *Proc. of COMPSTAT2000 (14TH Conference of the International Association for Statistical Computing)*, Utrecht. 2000.
3. Cestnik B. and I. Kononenko.: ASSISTANT 86: A knowledge-elicitation tool for sophisticated users. *In I. Bratko and N. Lavrac (eds.), Progress in Machine Learning.* Wilmslow, UK: Sigma Press. 1987
4. Holte R. C.: Very simple classification rules perform well on most commonly used datasets. *Machine Learning* 11, 63–91. 1993.
5. Friedman J. H.: A recursive partitioning decision rule for non-parametric classification. *IEEE Transactions on Computers.* 404–408. 1977.
6. Heckerman D.: Bayesian Networks for Data Mining. *Data Mining and Knowledge Discovery.* Vol. 1 Iss.1. 79–120. 1997.
7. Little R. J. A. and Rubin D.B.: *Statistical Analysis with Missing Data.* John Wiley & Sons Inc. 1987.
8. Quinlan J.: Unknown attribute values in induction. *Proceedings of the Sixth International Machine Learning Workshop* 164–168. San Mateo, CA: Morgan Kaufmann. 1989.
9. Quinlan J.: *C4.5: Programs for Machine Learning.* Los Gatos, CA: Morgan Kaufmann. 1992.
10. Ramos-Jimenez, G., Morales-Bueno, R.: Formalizacion de los Algoritmos TDIDT y CIDIM. Techn. Report LCC-ITI 99/01. Dept. Computer Science. Malaga University. 1999.
11. Ramos-Jimenez, G., Morales-Bueno, R., Villalba-Soria, A.: CIDIM. Control of Induction of Sample Division Method. Proc. of IC-AI'2000, International Conference on Artificial Intelligence. Las vegas, Nevada (USA) CSREA Press, 1083-1087. 2000.
12. Ramos-Jimenez, G.: Nuevos Desarrollos en Aprendizaje Inductivo. Tesis Doctoral. Departamento de Lenguajes y Ciencias de la Computacion. Universidad de Malaga. 2001.
13. Ramos-Jimenez, G., Morales-Bueno, R.: A new method for induction decision trees by sampling. *Neurocolt Workshop on Applications of Learning Theory* Bellaterra, Barcelona. 2000.