Missing Value Imputation Using a Semi-supervised Rank Aggregation Approach

Edson T. Matsubara, Ronaldo C. Prati,
Gustavo E.A.P.A. Batista, and Maria C. Monard

Institute of Mathematics and Computer Science at University of São Paulo
P. O. Box 668, ZIP Code 13560-970, São Carlos, SP, Brazil
{edsontm,prati,gbatista,mcmonard}@icmc.usp.br

Abstract. One relevant problem in data quality is the presence of missing data. In cases where missing data are abundant, effective ways to deal with these absences could improve the performance of machine learning algorithms. Missing data can be treated using imputation. Imputation methods replace the missing data by values estimated from the available data. This paper presents CORAI, an imputation algorithm which is an adaption of CO-TRAINING, a multi-view semi-supervised learning algorithm. The comparison of CORAI with other imputation methods found in the literature in three data sets from UCI with different levels of missingness inserted into up to three attributes, shows that CORAI tends to perform well in data sets at greater percentages of missingness and number of attributes with missing values.

1 Introduction

Machine learning (ML) algorithms usually take a set of cases as input (also known as examples or instances) to generate a model. These cases are generally represented by a vector, where each vector position represents the value of an attribute (feature) of a given case. However, in many applications of ML algorithms in real world data sets, some attribute values might be missing. For example, patient data may contain unknown information due to tests which were not taken, patients' refusal to answer certain questions, and so on.

In cases where missing data are abundant, effective ways to deal with these absences could improve the performance of ML algorithms. One of the most common approaches of dealing with missing values is imputation [1]. The main idea of imputation methods is that, based on the values present in the data set, missing values can be guessed and replaced by some plausible values. One advantage of this approach is that the missing data treatment is independent of the learning algorithm used, enabling the user to select the most suitable imputation method for each situation before the application of the learning algorithm.

A closely related research topic that has emerged as exciting research in ML over the last years is Semi-Supervised Learning (SSL) [2]. To generate models, SSL aims to use both labeled (*i.e.*, cases where the values of the class attribute, that is a special attribute we are interested in predicting based on the others

G. Zaverucha and A. Loureiro da Costa (Eds.): SBIA 2008, LNAI 5249, pp. 217–226, 2008.

attributes, are known in advance when generating the model) and unlabeled data (*i.e.*, cases where the values of the class attribute are not known when generating the model). To accomplish this task, some SSL algorithms attempt to infer the label of the unlabeled cases based on few labeled cases. Therefore, in a broad sense, these SSL algorithms might be seen as "imputation methods for the class attribute", which is the view considered in this work.

In this work we propose an algorithm named CORAI (CO-TRAINING Ranking Aggregation Imputation), which is an adaptation of the SSL algorithm CO-TRAINING, that can be used to deal with missing data. The comparison of CORAI with other imputation methods found in the literature in three data sets from UCI with different levels of missingness inserted into up to three attributes, assessed in terms of imputation error rate, shows that CORAI tends to perform well in data sets at greater percentages of missingness and number of attributes with missing values.

The outline of this paper is as follows: Section 2 presents related work. Section 3 describes CORAI. Section 4 presents the experimental results and Section 5 concludes this paper.

2 Related Work

According to the dependencies among the values of the attributes and the missingness, missing values can be divided into three groups [1]: (1) missing completely at random (**MCAR**) is the highest level of randomness and occurs where missingness of attribute values is independent of the values (observed or not); (2) missing at random (**MAR**) occurs when the probability of a case having a missing value may depend on the known values, but not on the value of the missing data itself; (3) not missing at random (**NMAR**) occurs when the probability of a case having a missing value for an attribute could depend on the value of that attribute.

A straightforward way to deal with missing values is to completely discard the cases and/or the attributes where missing values occur. Removing the cases is the most standard approach although, in case the missing values are concentrated into a few attributes, it may be interesting to remove them instead of removing the cases. Case and attribute removal with missing data should be applied only if missing data are MCAR, as not MCAR missing data have non-random elements, which can make the results biased.

Another approach is to fill in the missing data by guessing their values [3]. This method, known as imputation, can be carried out in a rather arbitrary way by imputing the same value to all missing attribute values. Imputation can also be done based on the data distribution inferred from known values, such as the "cold-deck/hot-deck" approach [4], or by constructing a predictive model based on the other attributes. An important argument in favor of this latter approach is that attributes usually have correlations among themselves. Therefore, these correlations could be used to create a predictive model for attributes with missing data. An important drawback of this approach is that the model estimated values

are usually more well-behaved than the true values would be. In other words, since the missing values are predicted from a set of attributes, the predicted values are likely to be more consistent with this set of attributes than the true (not known) values are. A second drawback is the requirement for correlation among the attributes. If there are no relationships among other attributes in the data set and the attribute with missing data, then the model will not be appropriate for estimating missing values.

As already mentioned, some SSL algorithms can be viewed as a way of "guessing the class" of a set of unlabeled cases. SSL algorithms have recently attracted considerable attention from the ML community, and numerous SSL approaches have been proposed (see [5] for an up-to-date review on the subject). In this paper, we are interested in investigating whether SSL approaches might be used to deal with the missing data problem. To the best of our knowledge, the only algorithm that is used to handle both missing data and SSL problems is Expectation Maximization [6]. In this paper, however, we are interested in a special family of SSL that can take advantage of alternative predictive patterns in the training data, such as multi-view SSL algorithms [7,8,9]. Our research hypothesis is that, by exploiting these alternative predictive patterns, missing data can be imputed in a better way than other methods.

3 Proposed Method

Let $X = A_1 \times ... \times A_M$ be the instance space over the set of attributes $\{A_1, ..., A_M\}$, and let $y \in Y = \{y_1, ..., y_Z\}$ be the class attribute. Assume that instances $(\mathbf{x}, y)$, where $\mathbf{x} \in X$ and $y \in Y$, are drawn from an unknown underlying distribution D. The supervised learning problem is to find $h : X \rightarrow Y$ from a training set of labeled examples $L = \{(\mathbf{x}_l, y_l) : l = 1, ..., n\}$ that are drawn from D. In semi-supervised learning, we also have unlabeled data $U = \{\mathbf{x}_u : u = n + 1, ..., N\}$ in the training set that are drawn from D without their corresponding class y_u. In our problem, some examples may have attributes with missing values and those attributes are denoted as A_i^*, where $dom(A_i^*) = dom(A_i) \cup \{\text{"?"}\}$ and "?" denotes a missing value. An instance space which contains A_i^* is represented as X^*. The imputation method to deal with missing values is to find $h^* : (X^*, Y) \rightarrow X$ which can be used to map all A_i^* back to A_i.

Numerous approaches can be used to construct h^*. Among them are predictive models, which can be used to induce relationships between the available attribute values and the missing ones. In this paper, we propose to adapt SSL algorithms to deal with missing data by imputation by considering each attribute A_i^* that has missing values as the class attribute into a SSL algorithm[1]. Therefore, examples which do not have missing values in A_i^* are treated as "labeled" attributes and examples with missing values are treated as "unlabelled."

Numerous SSL algorithms have been proposed in recent years. In this work, we have selected Co-training [7], a well known SSL algorithm, which was the first

[1] As we are dealing with missing value imputation as a semi-supervised classification problem, in this work we restrict the domain of A_i^* to be qualitative.

Algorithm 1. CORAI

```
Input: L, U
Output: L
Build U' ;
U = U − U';
while stop criteria do not met do
    Induce h1 from L;
    Induce h2 from L;
    R'1 = h1(U') ;
    R'2 = h2(U') ;
    R = bestExamples(R'1, R'2);
    L = L ∪ R;
    if U1 = ∅ then return(L) else
        Randomly select examples from U to replenish U';
    end
end
return(L1);
```

to introduce the notion of multi-view learning in this area. Multi-view learning is applied in domains which can naturally be described using different views. For instance, in web-page classification, one view might be the text appearing on the page itself and a second view might be the words appearing on hyperlinks pointing to this page from other pages on the web. CO-TRAINING assumes compatible views (examples in each view have the same class label) and each different view has to be in itself sufficient for classification. Although it is not always possible to find different views on data sets which meet these assumptions, we can only use one view and different learning algorithms to compose the views. This is an idea proposed in [8] which extends CO-TRAINING for problems restricted to one view data sets.

The main differences between CORAI and CO-TRAINING are the use of a different strategy to select the best examples to be labeled on each iteration and the use of two learning algorithms rather than two views. Our method can be described as follows: initially, a small pool of examples U' withdrawn from U are created, and the main loop of Algorithm 1 starts. First, the set of labeled examples L are used to induce two classifiers h_1 and h_2 using two different learning algorithms (in our case NAÏVE BAYES and C4.5). Next, the subset of unlabeled examples U' is labeled using h_1 and inserted into R'_1, and U' is used again but now it is labeled using h_2 and inserted into R'_2. Both sets of labeled examples are given to the function *bestExamples* which is responsible for ranking good examples according to some criterion and inserting them into L. After that, the process is repeated until a stop criteria is met.

We also modify the *bestExamples* function as proposed in the CO-TRAINING method. Originally, this function first filters examples which disagrees with their classification, *i.e*, $h_1(\mathbf{x}) \neq h_2(\mathbf{x})$. However, attributes with missing values may assume many different values, and when examples are filtered, $h_1(x) \neq h_2(x)$ for almost all examples. This occurs because it is less likely that classifiers agree

with their classification in multi-class problems rather than binary problems. To deal with this problem, we proposed the use of ranking aggregation to select the best examples.

From now on, assume that the missing value problem has been mapped to a semi-supervised learning problem by swapping the class attribute with an attribute A_i^*. Thus, the reader should be aware that Y is actually referring to A_i^*.

First, we need to define classification in terms of *scoring classifiers*. Scoring classifiers maps $s : X \rightarrow \mathbb{R}^{|Y|}$, assigning a numerical score $s(\mathbf{x})$ to each instance $\mathbf{x} \in X$ for each class. NAÏVE BAYES actually computes a sort of scoring classifier where the classification is given by the class with the largest score. Decision trees can also be adapted to output scores by counting the distribution of examples for each possible classification in their leaves.

A rank aggregation combines the output of scoring classifiers on the same set of examples in order to obtain a "better" ordering. In this paper, rank aggregation uses two scoring classifiers obtained from two sets of examples R'_1 and R'_2 scored by h_1 and h_2, respectively (Algorithm 1). Let y_{1z} be the scores given by h_1 and y_{2z} be the scores given by h_2 for the class y_z $(z = 1...Z)$. The method which implements best examples orders examples according to scores for each class and compute $rpos_{1z}$ which is the rank position of an instance with regards to y_z on R'_1. For instance, to compute $rpos_{11}$, initially the instances according to y_{11} are ordered and then the rank position from this ordering is stored in $rpos_{11}$. This is done for all classes $y_{11}, ..., y_{1Z}$ to obtain $rpos_{11}, ..., rpos_{1Z}$. In the same way, the method uses R'_2 to compute $rpos_{2z}$. Finally, the rank position obtained from R'_1 and R'_2 is given by $rpos_z = rpos_{1z} + rpos_{2z}$ for each class y_z, and the selected instances are the ones with low $rpos_z$. Taking the instances with low $rpos_z$ means that these examples have a good (low) rank position on average, which is similar to selecting the examples with high confidence in a ranking perspective. In our implementation, we preserve the class distribution observed in the initial labeled data by selecting the number of examples proportional to this distribution.

4 Experimental Analysis

4.1 Experimental Setup

The experiments were carried out using three data sets from UCI Machine Learning Repository [10]. Originally, all data sets had no missing values and missing data were artificially implanted into the data sets. The artificial insertion of missing data allows a more controlled experimental setup. First of all, we can control the pattern of missing data. In this work, missing data were inserted in the MCAR pattern. Secondly, as the values replaced by missing data are known, imputation errors can be measured. Finally, this experimental setup allows the missing data to be inserted using different rates and attributes.

Table 1 summarizes the data sets used in this study. It shows, for each data set, the number of examples (#Examples), number of attributes (#Attributes),

Table 1. Data sets summary description

Data set	# Examples	#Attributes (quanti., quali.)	#Classes	Majority error
CMC	1473	9 (2,7)	3	57.29%
German	1000	20 (7,13)	2	30.00%
Heart	270	13 (7,6)	2	44.44%

together with the number of quantitative and qualitative attributes, number of classes (#Classes), and the majority class error.

Ten-fold cross-validation was used to partition the data sets into training and test sets. Finally, missing values were inserted into the training sets. Missing values were inserted in 20%, 40%, 60% and 80% of the total number of examples of the data set. In addition, missing values were inserted in one, two or three attributes. In order to choose in which attributes to implant missing data, we conducted some experiments to select attributes that are relevant to predict the class attribute. Observe that it is important to insert missing values into relevant attributes, otherwise the analysis might be hindered by dealing with non-representative attributes which will not be incorporated into the classifier by a learning algorithm such as a decision tree. Since finding the most representative attributes of a data set is not a trivial task, three feature subset algorithms, available in Weka software [11] were used: Chi-squared ranking filter; Gain ratio feature evaluator and ReliefF ranking filter. All three feature selection algorithms generate a ranking of the most representative attributes. For each data set the three rankings were composed into an average ranking, and the three top ranked qualitative attributes were chosen. Table 2 shows the selected attributes for each data set, as well as the number of values (#Values) of each attribute.

As mentioned before, missing values were inserted in 20%, 40%, 60% and 80% of the total number of examples for one (the attribute selected in first place), two (the attributes selected in first and second places) and the three selected attributes (the attributes in first, second and third places). Inserting missing values into more than one attribute is performed independently for each attribute. For instance, 20% of the missing values inserted into two attributes means that, for each attribute, two independent sets with 20% of examples each were sampled. In other words, the first set's values were altered to missing for

Table 2. Selected attributes for data set

Data set	Selected Attributes (Position) Name	#Values
CMC	1st (1) wife education	4
	2nd (2) husband education	4
	3th (7) standard of living	4
German	1st (0) status	4
	2nd (2) credit history	5
	3th (5) savings account	5
Heart	1st (12) thal	3
	2nd (2) chest pain	4
	3th (8) angina	2

the first attribute, similarly, the second set's values were altered to missing for the second attribute. As the two sets are independently sampled, some examples may have missing values in one, two or none of the selected attributes. A similar procedure was performed when missing data was inserted into three attributes.

Our experimental analysis involves the following methods to deal with missing data: CORAI, the proposed method; 9NNI [3], an imputation method based on k-nearest neighbor; and mode imputation, an imputation method that substitutes all the missing data by the attributes' most frequent value. In order to deal with missing values in multiple attributes, CORAI is executed independently for each attribute. In each execution, one different attribute with missing data is considered as class, and all other attributes are left in the data set to build the classification model.

4.2 Experimental Results

The main purpose of our experimental analysis is to evaluate the imputation error of the proposed method compared with other methods in the literature. As stated before, missing data were artificially implanted and this procedure allows us to compare the imputed values with the true values. Imputation error rate was calculated for each possible attribute value.

In order to analyze whether there is a difference among the methods, we ran the Friedman test[2]. Due to lack of space, only results of these tests are reported here[3]. Friedman test was ran with four different null-hypotheses: (1) that the performance of all methods are comparable considering all results; (2) that the performance of all methods are comparable for each percentage of missing data; (3) that the performance of all methods are comparable for each amount of attributes with missing data; (4) that the performance of all methods are comparable for each percentage of missing data and amount of attributes with missing data. When the null-hypothesis is rejected by the Friedman test, at 95% of confidence level, we can proceed with a post-hoc test to detect which differences among the methods are significant. We ran the Bonferroni-Dunn multiple comparison with a control test, using CORAI as a control. Therefore, the Bonferroni-Dunn test points-out whether there is a significant difference among CORAI and the other methods involved in the experimental evaluation.

Figure 1 shows the results of the Bonferroni-Dunn test with our first null-hypothesis: that the performance of all methods are comparable considering all results. This test does not make any distinction among percentage of missing data or amount of attributes with missing data. As seen in Figure 1, CORAI performs best, followed by 9NNI and MODE. The Bonferroni-Dunn test points out that CORAI outperforms MODE, but there is no significant difference between CORAI and 9NNI.

[2] The Friedman test is the nonparametric equivalent of the repeated-measures ANOVA. See [12] for a thorough discussion regarding statistical tests in machine learning research.

[3] All tabulated results can be found in `http://www.icmc.usp.br/~gbatista/corai/`

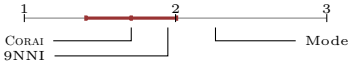
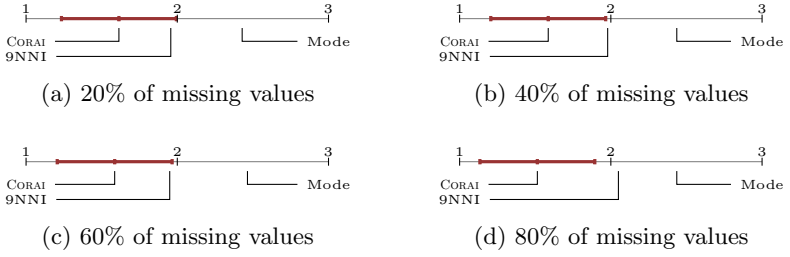


Fig. 1. Results of the Bonferroni-Dunn test on all imputation errors. The thick line marks the interval of one critical difference, at 95% confidence level.

The second null-hypothesis is that all methods perform comparably for each percentage of missing data. The objective is to analyze whether some methods perform better than others when the percentage of missing values varies, or in a critical situation, when the percentage of missing values is high. Figure 2 shows the results of the Bonferroni-Dunn test with our second null-hypothesis. Corai frequently outperforms the other methods.



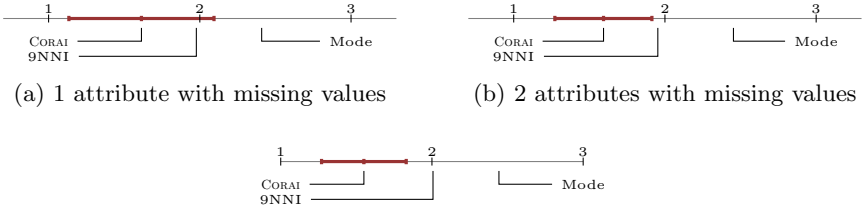
(a) 20% of missing values (b) 40% of missing values

(c) 60% of missing values (d) 80% of missing values

Fig. 2. Results of the Bonferroni-Dunn test considering the percentage of missing values. The thick line marks the interval of one critical difference, at 95% confidence level.

Our third null-hypothesis is that all methods perform comparably for different amounts of attributes with missing data. The objective is to analyze whether some methods perform better than others when the number of attributes with missing values increases. Figure 3 shows the results of the Bonferroni-Dunn test. Corai obtained the lowest imputation error followed by 9NNI and Mode. Furthermore, Corai outperformed all other methods, at 95% confidence level, when missing values were present in two or three attributes. When missing values were implanted in only one attribute Corai performs better than Mode, but do not outperform 9NNI.

Table 3 shows the results of the Friedman and Bonferroni-Dunn tests for the fourth hypothesis. With this hypothesis we analyze whether all methods perform comparably given different combinations of percentage of missing data and number of attributes with missing data. In this table, column "%Missing" represents the percentage of missing data inserted into the attributes; "#Attributes" stands for the number of attributes with missing values; columns "Corai", "9NNI" and "Mode" show the results of the Friedman test for the respective method; and finally, column "CD" presents the critical different produced by the Bonferroni-Dunn test. In addition, results obtained by Corai that outperform 9NNI and Mode at 95% confidence level are colored with dark gray, and results of Corai



(a) 1 attribute with missing values

(b) 2 attributes with missing values

(c) 3 attributes with missing values

Fig. 3. Results of the Bonferroni-Dunn test considering the number of attributes with missing values. The thick line marks the interval of one critical difference, at 95% confidence level.

Table 3. Results of the Friedman and Bonferroni-Dunn tests considering the amount of missing data and number of attributes with missing values. The thick line marks the interval of one critical difference.

%Missing	#Attributes	Imputation Methods			CD
		Corai	9NNI	Mode	
m20	#At=1	1.727	1.864	2.409	0.96
	#At=2	1.625	1.958	2.417	0.65
	#At=3	1.571	1.986	2.443	0.54
m40	#At=1	1.545	2.045	2.409	0.96
	#At=2	1.625	1.917	2.458	0.65
	#At=3	1.571	2.000	2.429	0.54
m60	#At=1	1.636	1.955	2.409	0.96
	#At=2	1.583	1.938	2.479	0.65
	#At=3	1.571	1.957	2.471	0.54
m80	#At=1	1.545	2.045	2.409	0.96
	#At=2	1.542	2.000	2.458	0.65
	#At=3	1.486	2.086	2.429	0.54

that outperform Mode only are colored with light gray. As can be observed, Corai always perform better than the other imputation methods.

As a final analysis we ran the C4.5 and Naïve Bayes learning algorithms on the imputed data sets and measured the misclassification error on the test sets. Due to the lack of space, these results are not reported here. We also ran the Friedman test in order to verify whether there is a significant difference among the classifiers. Following the Friedman test there was no significant difference at 95% confidence level.

5 Conclusion

This paper presented Corai, an algorithm for missing values imputation inspired in the multi-view SSL algorithm Co-training. Imputation using Corai was compared with Mode and 9NNI, two imputation methods found in the literature, with four percentage of missingness (20%, 40%, 60% and 80%) artificially introduced in one, two and three attributes in three data sets from UCI

machine learning repository. Results in these three data set show that CORAI tends to perform better at greater percentages of missingness and number of attributes with missing values.

One limitation of CORAI is that it only handles qualitative attributes. We plan to extend CORAI to quantitative attribute as a future work. Another important research direction is to evaluate how CORAI performs in patterns of missingness other than MCAR.

Acknowledgements. We wish to thank the anonymous reviewers for their helpful comments. This research was partially supported by the Brazilian Research Councils CAPES, FAPESP, CNPq and FPTI-BR.

References

1. Little, R.J.A., Rubin, D.B.: Statistical analysis with missing data. John Wiley & Sons, Inc., New York (1986)
2. Chapelle, O., Schölkopf, B., Zien, A. (eds.): Semi-Supervised Learning. MIT Press, Cambridge (2006)
3. Batista, G.E.A.P.A., Monard, M.C.: An analysis of four missing data treatment methods for supervised learning. Applied Art. Intell. 17(5-6), 519–533 (2003)
4. Levy, P.: Missing data estimation, 'hot deck' and 'cold deck'. In: Encyclopedia of Biostatistics. Wiley, Chichester (1998)
5. Zhu, X.: Semi-supervised learning literature survey. Computer Sciences TR 1530, University of Wisconsin Madison (2007), `http://pages.cs.wisc.edu/~jerryzhu/research/ssl/semireview.html`
6. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the em algorithm. Journal of Royal Stat. Soc. B39, 1–38 (1977)
7. Blum, A., Mitchell, T.M.: Combining labeled and unlabeled sata with co-training. In: Conference on Learning Theory, pp. 92–100 (1998)
8. Goldman, S., Zhou, Y.: Enhancing supervised learning with unlabeled data. In: International Conference on Machine Learning, pp. 327–334 (2000)
9. Zhou, Z.H., Li, M.: Tri-training: Exploiting unlabeled data using three classifiers. IEEE Transactions on Knowledge and Data Engineering 17(11), 1529–1541 (2005)
10. Asuncion, A., Newman, D.: UCI machine learning repository (2007)
11. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations, 2nd edn. Morgan Kaufmann, San Francisco (2005)
12. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research 7, 1–30 (2006)