A Nearest-Neighbor Method as a Data Preparation Tool for a Clustering Genetic Algorithm.

Conference Paper · January 2003

Source: DBLP

CITATIONS
3

READS
62

3 authors:

Eduardo R Hruschka
University of São Paulo
109 PUBLICATIONS **2,018** CITATIONS
SEE PROFILE

Estevam Rafael Hruschka
Universidade Federal de São Carlos
105 PUBLICATIONS **1,529** CITATIONS
SEE PROFILE

Nelson Ebecken
Federal University of Rio de Janeiro
300 PUBLICATIONS **1,516** CITATIONS
SEE PROFILE

A Nearest-Neighbor Method as a Data Preparation Tool for a Clustering Genetic Algorithm

Eduardo R. Hruschka[1], Estevam R. Hruschka Jr.[2], Nelson F. F. Ebecken[3]
[(1)]Universidade Católica de Santos (UniSantos).
[(2,3)]COPPE / Universidade Federal do Rio de Janeiro.

Abstract

This paper presents a Nearest-Neighbor Method to substitute missing values in continuous datasets and show that it can be useful for a Clustering Genetic Algorithm. The proposed method is evaluated by means of simulations performed in the Wisconsin Breast Cancer Dataset, which is a benchmark for data mining methods. In this sense, we verify the efficacy of the proposed method in the context of a Clustering Genetic Algorithm, comparing the average classification rates obtained in the original dataset with those obtained in a dataset formed by the substituted values. The simulation results show that the proposed method is promising.

1. Introduction

Knowledge discovery in databases (KDD) is the non-trivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data [1]. In this context, data mining is a step in this process that centers on the automated discovery of new facts and relationships in data and it consists of three basic steps: data preparation, information discovery and analysis of the mining algorithm output [2].

The data preparation step has a major importance in the whole process and it is used as a tool to adjust the databases to the information discovery step. Thus, when it is performed in a suitable way higher quality data are produced, and the KDD outcomes can be improved. In spite of its importance, the data preparation process became an effervescent research area only in the last few years.

The substitution of missing values is an important subtask in the data preparation step. The absence of values in a dataset is a common fact in real-world applications and, further that, it may generate bias in the data, affecting the quality of the KDD process. One of the most used methods to deal with the missing values problem is the mean or mode imputation [3], but this method can bring bias in the data and it is not adequate in all situations. This work describes a missing value substitution method based on the Instance-based Learning Method [4]. More specifically, we propose a Nearest-Neighbor method, which is employed to substitute missing values in datasets to be grouped by the Clustering Genetic Algorithm [29]. It is important to emphasize that, although the substitution method is a prediction task, the prediction results are not the only important issue to be analyzed. In this sense, one of the most important aspects is that the substitution method must generate values that least distort the original characteristics of the original sample [28]. In other words, the quality of the substitution process must be verified in the clustering context. To do so, we compare the average classification rates obtained in the original dataset with those obtained in a substituted dataset.

The next section presents related works concerning the missing values problem, whereas Section 3 presents our proposed method to substitute them. Section 4 briefly describes the employed Clustering Genetic Algorithm [29]. The proposed method is evaluated in the Wisconsin Breast Cancer Dataset, which is a benchmark for data mining methods, and the obtained results are described in Section 5. Finally, Section 6 describes the conclusions and points out some future work.

2. Related Work

The missing values problem is an important issue in data mining. Thereby there are many approaches to deal with it[5]: i) Ignore objects containing missing values; ii) Fill the gaps manually; iii) Substitute the missing values by a constant; iv) Use the mean of the objects in the same class as a substitution value; and v) Get the most probable value to fill the missing ones.

When working with decision trees, some practical results with missing values substitution methods can be found in [6], which just ignore the objects with missing values. In [7] the authors define the *majority method*, which replaces the missing values by the most frequent value in the objects from the same class. The *probability method* [8] constructs a decision tree to determine the missing values of each attribute - by using the information contained in other attributes and ignoring the class. The *dynamic path generation* [6] and the *Lazy decision tree approach* [9] do not generate the whole tree, but only the most promising path instead.

For a missing value classification task using committee learning approach see *Boosting* [10], *Bagging* [11], *Sasc* (Stochastic attribute selection committee) [12] and *SascMB* (Stochastic attribute selection committee with Multiple Boosting) [13].

When using a Bayesian method and having a MAR (Missing At Random) missing data mechanism [16], a way to find the posterior distribution of the joint probabilities of the variables and the marginal probability of the variable (having missing values) is to treat the missing values as unknown parameters and apply a MCMC (Monte Carlo Markov Chain) method [17]. If the missing data mechanism is NI (Not Ignorable) [15] an *imputation-based* analysis can be used [16]. However, these methods have some disadvantages [14]. First, they need information about the missing values mechanism. Second, sampling variability and non-response variability are mixed and finally, they have a high computational cost. Other Bayesian approaches can be found in [18,19].

When doing multivariate analysis, some works apply the Multiple Imputation (MI) [20] method to handle missing data. The MI method can give good estimation of the sample standard errors and any kind of analysis can be applied. Unfortunately the data must be missing at random (MAR) to generate a general-purpose imputation.

The EM (Expectation-Maximization) algorithm [21] can be applied when the model belongs to an exponential family, but it has a slow convergence rate. The MS-EM (Model Selection – Expectation Maximization) [22], which plays relatively few iterations to find the best network with incomplete data, implements a version of the EM and uses a metric to choose the best Bayesian model. Other methods applying the EM algorithm can be seen in [15, 23].

Instance-based (IB) learning methods [4] are part of another class of algorithms that can be applied to the missing values substitution process. There are some classical IB learning algorithm classes as the *Nearest Neighbor* (k-NN) [3], the *locally weighted linear regression* [24], and the *Case Based Reasoning* (CBR) [25]. One of the most important characteristics of these methods is that they do not generate a model to describe the data. In other words, they do not have the training step as the other learning methods do. Thus, instead of consulting a generated

model to estimate the best value to substitute the missing one (for each substitution), these algorithms search the whole dataset to find the best instance to be used. This characteristic produces a high computational cost when working with many attributes. On the other hand, as the learning process is specific to each query, it may be more accurate. The next section describes our proposed method to substitute missing values.

3. Nearest-Neighbor Method

Our proposed method considers that missing values can be substituted by the corresponding attribute value of the most similar complete instance (or object) in the dataset. In other words, we employ a K-nearest-neighbor method [4], using K=1 and the Euclidean distance function.

More specifically, let us consider two objects *i* and *j*, both described by a set of N continuous attributes $\{x_1,x_2,...,x_N\}$. The distance between object *i* and object *j* will be here called d(i,j). Besides, let us suppose that the *k*-th attribute value ($1 \leq k \leq N$) of the object *m* is missing. Thus, the Nearest-Neighbor Method (NNM) will compute the distances d(m,i), for all $i \neq m$, according to the Euclidean distance:

$$d(m,i)_E = \sqrt{(x_1^m - x_1^i)^2 + ... + (x_{k-1}^m - x_{k-1}^i)^2 + (x_{k+1}^m - x_{k+1}^i)^2 + ... + (x_N^m - x_N^i)^2} \quad . \qquad \textbf{(1)}$$

One observes that we are not taking into account the attribute x_k, because it is missing. After computing all the distances, we choose the smallest one, which refers to the most similar object in respect to *m*. This object is here called *s*, which is the nearest neighbor. In this way, d(m,s)=min d(m,i) for all $i \neq m$, and x_k^m is substituted by x_k^s. Besides, if there is a set of objects whose distances d(m,i) are equal, the substituted value comes from the first object of this set.

The proposed method can be easily adapted to datasets formed by discrete attributes. To do so, one can just change the Euclidean distance function by the Simple Matching Approach [26]. Thus, the NNM can be employed in several data mining tasks. In this paper we investigate a clustering application. However, the NNM can also be used in classification problems. In this sense, the substitution process can be performed in the examples of each class separately. Besides, one can employ a NNM based on the Simple Matching Approach [26] in problems involving the extraction of association rules.

4. Clustering Genetic Algorithm

This section briefly describes the Clustering Genetic Algorithm (CGA), whose basic steps are depicted in Figure 1. Further details can be found in [29]. Basically, clustering is a task where one seeks to identify a finite set of categories or clusters to describe the data. This work considers that clustering involves the partitioning of a set *X* of objects into a collection of mutually disjoint subsets C_i of *X*. Formally, let us consider a set of *N* objects $X=\{X_1,X_2,...,X_N\}$ to be clustered, where each $X_i \in \Re^\rho$ is an attribute vector consisting of ρ real measurements. The objects must be clustered into non-overlapping groups $C=\{C_1,C_2,...,C_k\}$ where *k* is the number of clusters, such that:

$$C_1 \cup C_2 \cup ... \cup C_k = X, \;\; C_i \neq \varnothing, \;\; \text{and} \;\; C_i \cap C_j = \varnothing \;\; \text{for} \;\; i \neq j. \qquad \textbf{(2)}$$

Considering that there are N objects to be clustered, the CGA employs genotypes represented as one dimensional integer arrays with (N+1) elements. As each data unit can be num-

bered from 1 to N, the i-th element of a genotype represents the i-th data unit, whereas the last gene represents the number of clusters. Therefore, each gene of a chromosome has a value over the alphabet {1,2,3,…,k}, where *k* is the maximum number of clusters. For example, considering a dataset with 20 objects one can get the following clustering: 22345123453321454552 5. This means that 5 objects {1,2,7,13,20} form the cluster whose label is 2, and the cluster whose label is 1 has 2 objects {6,14}. The last gene corresponds to the number of clusters encoded by the solution.

The CGA is supposed to optimize not only the clusterings for a given number of clusters but also the number of clusters. Therefore, the CGA employs an objective function based on the Average Silhouette Width [26]. Let us consider an object *i* belonging to cluster A. So the average dissimilarity of *i* to all other objects of A is denoted by a(i). Now consider a different cluster C and let us calculate the average dissimilarity of *i* to all objects of C, which will be here denoted by d(i,C). After computing the d(i,C) for all clusters C≠A we select the smallest of those, b(i)=min d(i,C) for C≠A .This number represents the dissimilarity of *i* to its neighbor cluster. Now one defines the silhouette s(i) like follows:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \tag{3}$$

When cluster A contains only one object we consider that s(i)=0, which is the most neutral choice [26]. In addition, it is easy to see that -1≤s(i)≤1. The objective function is the average of s(i) for i=1,2,...,N. It implies that the best value of *k* happens when the objective function value is as high as possible.

Considering the selection process, the genotypes that make part of each generation are selected according to the roulette wheel selection strategy. As this strategy does not allow negative objective function values, a constant number equals to one is summed up to each objective function value before the selection process takes place. Besides, the best genotype is always copied into the succeeding generation.

The CGA employs two basic operators for crossover and mutation. The crossover operator combines pieces of information coming from different genotypes and it works in the following way: First, two genotypes (A and B) are selected; Second, considering that A represents k1 clusters, the CGA chooses randomly $n \in [1,k1]$ clusters to copy in B. The unchanged groups of B are maintained and the changed ones have their objects allocated to the cluster that has the nearest centroid. In this way the child C is obtained. This same process is employed to get child D, but now considering that the changed clusters of B are copied in A.

There are two operators for mutation. The operator 1, which only works in clusterings formed by more than two groups, eliminates a randomly chosen group and places all their objects to the remaining cluster that has the nearest centroid. The operator 2 divides a randomly selected group into two new ones. The first group is formed by the objects closer to the centroid, whereas the other group is formed by those objects nearer to the farthest object to the centroid. Besides, 50% of the genotypes are crossed-over, 25% are mutated by operator 1 and 25% are mutated by operator 2.

We have employed the methodology developed in [26] to set up the initial population. The initial clustering process is based on the selection of representative objects. The first selected object is the most centrally located in the set of objects. Subsequently, other objects are selected. Basically, the chance of selecting an object increases when it is far from the previously selected ones and when there are many objects next to it. After selecting the representative

objects, the initial population is formed considering that the non-selected objects must be clustered according to their proximity to the representative ones. Considering *k* representative objects, the first genotype represents two clusters, the second genotype represents three clusters,…, and the last one represents *k* clusters. Thus, we have employed initial populations formed by (k-1) genotypes, where each genotype represents a different clustering.

1) Initialize a population of genotypes;
2) Evaluate each genotype in the population;
3) Apply a linear normalization;
4) Select genotypes by proportional selection;
5) Apply crossover and mutation;
6) Replace the old genotypes by the ones formed by 5);
7) If the convergence is attained, stop; if not, go to step 2).

Figure 1. Clustering Genetic Algorithm (CGA).

5. Simulation Results

We performed simulations using the Wisconsin Breast Cancer Dataset [27], which is a benchmark for data mining methods. In this database, each object has 9 attributes and an associated class label (benign or malignant). The attributes are: clump thickness (A1), uniformity of cell size (A2), uniformity of cell shape (A3), marginal adhesion (A4), single epithelial cell size (A5), bare nuclei (A6), bland chromatin (A7), normal nucleoli (A8) and mitoses (A9). All attribute values belong to the set {1,2,…,9}. Therefore, it is not necessary to normalize them. The classes are known to be linearly inseparable and the total number of objects is 699 (458 benign and 241 malignant), of which 16 have a single missing value. We removed those 16 objects and used the remaining ones (in all simulations) to simulate substitutions of missing values.

This dataset was chosen mainly because the CGA has already shown a good performance when dealing with it [19,29]. In this sense, we are interested in verifying if the CGA good performance in this dataset is maintained when simulated missing values are substituted by means of the proposed Nearest-Neighbor Method (NNM). Besides, we also compare the NNM results with those obtained by means of a simple, but usually employed, substitution method, which consists in substituting the missing values by the mean of the attribute values.

It is important to emphasize that, although the substitution method is a prediction task, the prediction results are not the only important issue to be analyzed. In this sense, one of the most important aspects is that the substitution method must generate values that least distort the original characteristics of the original sample [28]. In other words, the quality of the substitution process must be verified in the clustering context. To do so, we compare the average classification rates obtained in the original dataset with those obtained in a *substituted dataset* - formed by all substituted values.

Our NNM simulations consider that there is just one missing value at a time, and we employed the dataset formed just by the attribute values (without the *class label*). Let us consider that one has a dataset formed by L objects i=$\left(x_1^i, x_2^i, ..., x_N^i\right)$. First, we simulate that x_1^1 is missing and it is consequently substituted. Second, x_2^1 is missing and it is consequently substituted. This process is repeated until x_N^1 is substituted. After that, we simulate that x_1^2 is missing and it is consequently substituted. In summary, this procedure is repeated for all x_k^i (i=1,...,L;

k=1,...,N). After the substitution process, one has two datasets (the original one and the substituted one) and it is possible to verify how similar the substituted values are compared to the original ones. To do so, we calculate the absolute difference between the substituted value and the original one. Thus, we get the average prediction error for each attribute and Table 1 shows the obtained results. One can observe that the substitution by the Nearest-Neighbor Method provided better results than the substitution by the mean in all attributes.

Attribute	Nearest-Neighbor Substitution	Substitution by the Mean
A1	2.09	2.30
A2	0.88	2.53
A3	1.07	2.49
A4	1.35	2.25
A5	1.35	1.70
A6	1.82	3.18
A7	1.37	1.96
A8	1.32	2.46
A9	0.95	1.00

Table1. Average distance between original and substituted values.

Clustering algorithms can be evaluated by means of *classification datasets*. In this sense, the clustering algorithm is applied in the classification dataset (without the class labels) in order to verify whether it finds the correct classes/clusters or not. The CGA was applied in the original dataset and in the dataset formed only by substituted values. We have also employed the Euclidean Distance (1) as the CGA dissimilarity measure. The CGA populations were formed by 20 genotypes that, in turn, implies in using 21 clusters at most. Besides, we simulated the clustering process 11 times (this number is convenient to perform the Wilcoxon/Mann-Whitney statistical test) for each dataset and the maximum number of generations was set to 100. Tables 2 and 3 show the obtained results, where ACR stands for Average Classification Rate and μ and σ respectively stand for the mean and the standard deviation.

Simulation	**ACR(%) Benign**	**ACR(%) Malignant**	**ACR(%) Total**
1	98.20	89.54	95.17
2	98.20	90.38	95.46
3	98.20	91.21	95.75
4	98.20	89.12	95.02
5	98.42	89.96	95.46
6	98.20	89.54	95.17
7	98.20	90.79	95.61
8	98.20	91.21	95.75
9	98.20	91.21	95.75
10	98.42	88.70	95.02
11	98.20	91.21	95.75
μ	98.24	90.26	95.45
σ	0.09	0.94	0.30

Table 2. CGA Results in the Original Dataset.

Simulation	ACR(%) Benign	ACR(%) Malignant	ACR(%) Total
1	98.20	87.45	94.44
2	97.97	90.79	95.46
3	98.20	89.54	95.17
4	97.75	91.21	95.46
5	97.30	91.63	95.31
6	97.75	90.38	95.17
7	97.30	91.63	95.31
8	97.30	91.63	95.31
9	97.52	91.63	95.46
10	97.75	91.21	95.46
11	97.52	91.63	95.46
μ	97.69	90.79	95.27
σ	0.34	1.29	0.30

Table 3. CGA Results in the Substituted Dataset.

The CGA provided very similar ACRs in both datasets. In fact, we applied the Wilcoxon statistical test [30] (also known as Mann-Whitney test), supposing that the "ACR-Total" values in the original dataset are equal to those obtained in the substituted dataset, and we concluded that the results are statistically significant at the 5% significance level. Thus, the NNM provided good results in the CGA context.

6. Conclusions and Future Work

We presented a Nearest-Neighbor Method (NNM) to substitute missing values in continuous datasets and showed that it can be useful for a Clustering Genetic Algorithm (CGA) [29]. The proposed method considers that each instance containing missing values should be compared with complete instances, using a distance metric, and the closest complete instance should be used to assign the missing attribute value.

The proposed method was evaluated by means of simulations performed in the Wisconsin Breast Cancer Dataset. To do so, we compared the obtained results – in a prediction task - with those provided by a simple and usually employed method, i.e. using the mean. The simulations showed that the NNM provided better results than the use of the mean. Besides, we evaluated the efficacy of the proposed substitution method in the CGA context, comparing the obtained results in the original dataset with those obtained in the substituted dataset. These simulations showed that the NNM is a good data preparation tool for the CGA.

Considering our future work, there are many aspects that can be further investigated**.** Substitution methods must generate values that least distort the original characteristics of the original sample [28]. In this sense, the quality of the substitution process is going to be verified in classification tasks. More specifically, we are interested in comparing the NNM results with those obtained with Bayesian Networks [18,19]. In addition, we are also going to evaluate the efficacy of the proposed method in other datasets.

References

1. Fayyad, U. M., Shapiro, G. P., Smyth, P. "From Data Mining to Knowledge Discovery: An

Overview". In: Advances in Knowledge Discovery and Data Mining, Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R., Editors, MIT Press, pp. 1-37, 1996.
2. Bigus, J. P., Data Mining with Neural Networks , First edition, USA, McGraw-Hill, 1996.
3. Batista, G. E. A. P. & Monard, M. C., An Analysis of Four Missing Data Treatment Methods for Supervised Learning, Proceedings of the First International Workshop on Data Cleaning and Preprocessing, IEEE International Conference on Data Mining, 2002.
4. Mitchell, T. M., Machine Learning, McGraw-Hill, 1997.
5. Han, J. & Kamber, M. - Data Mining: Concepts and Techniques. Morgan Kaufmann, 2001.
6. Quinlan, J., Induction of Decision Trees, Machine Learning 1, 81-106, 1986.
7. Kononenko, I., Bratko, I. & Roskar, E., Experiments in Automatic Learning of Medical Diagnostic Rules. Technical Report, Jozef Stefan Institute, Ljubjana, Yogoslavia, 1984.
8. Quinlan, J. R., Unknown Attribute Values in Induction, Proceedings of 6th International Workshop on Machine Learning, 164-168, Ithaca, NY, 1989.
9. Friedman, H. F., Kohavi, R. & Yun, Y., Lazy Decision Trees, Proceedings of the 13th National Conference on Artificial Intelligence, pp. 717-724, AAAI Press/MIT Press, 1996.
10. Schapire, R. E., Freund, Y., Barlett, P. & Lee, W. S., Boosting the Margin: A New Explanation for the Effectiveness of Voting Methods, Proceedings of the 14th International Conference on Machine Learning, Morgan Kaufmann, pp. 352-330, 1997.
11. Breiman, L. Bagging Predictors, Machine Learning, 24, 123-140, 1996.
12. Zheng, Z. & Webb, G. I., Stochastic Attribute Selection Committees, Proceedings of the 10th Australian Joint Conference of Artificial Intelligence, Springer-Verlag, 1998.
13. Zheng, Z. & Webb, G. I., Stochastic Attribute Selection Committees with Multiple Boosting: Learning more Accurate and more Stable Classifier Committees, Proceedings of the 3rd Pacific-Asia Conference on Knowledge Discovery and Data Mining, Berlin: Springer-Verlag, 1999.
14. Gilks W. R., Richardson, S. & Spiegehalter, D. J., Markov Chain Monte Carlo in Practice. Chapman and Hall, London, 1996.
15. Gelman, A., Carlin, J. B., Stern, H. S. & Rubin, D. B., Bayesian Data Analysis, Chapman and Hall, London, 1995.
16. Sebastiani, P. & Ramoni, M., Bayesian Inference with Missing Data Using Bound and Collapse, Technical Report KMI-TR-58, KMI, Open University, 1997.
17. Little, R. & Rubin, D. B., Statistical Analysis with Missing Data, John Wiley & Sons, New York, 1987.
18. Hruschka Júnior, E. R., Ebecken, N. F. F. Missing Values prediction with K2. Intelligent Data Analysis.(IDA).IOS Press, Netherlands: , v.6, n.6, 2002.
19. Hruschka Júnior, E. R., Hruschka, E. R., Ebecken, N. F. F. A Data Preparation Bayesian Approach for a Clustering Genetic Algorithm, Frontiers in Artificial Intelligence and Applications, A. Abraham et al. (Eds), Soft Computing Systems: Design, Management and Applications, pp. 453-461, ISBN 1 58603 297 6, IOS Press, Proceedings of the Second International Conference on Hybrid Intelligent Systems (HIS 02), Chile, 2002.
20. Rubin, D. B., Multiple Imputation for non Responses in Surveys, New York, John Wiley & Sons, 1987.
21. Dempster, A. P., Laird, N. M. & Rubin, D. B., Maximum Likelihood from Incomplete Data via the EM algorithm, Journal of the Royal Statistical Society B, 39, 1-39, 1977.
22. Friedman, N., Learning Belief Networks in the presence of Missing Values and Hidden Variables, Proceedings of the 14th International Conference on Machine Learning, 1997.

23. Lauritzen, S. L., The EM Algorithm for Graphical Association Models with Missing Data, Computational Statistics and Data Analysis, 19, 191-201, 1995.
24. Atkeson, C. G., Moore, A. W., Schaal, S. A., Locally Weighted Learning for Control. AI Review, 1997.
25. Aamodt, A. & Plazas, E., Case-Based Reasoning: Methodological Variations, and System Approaches, AI Communications, 7(1), 39-52, 1994.
26. Kaufman, L., Rousseeuw, P. J., Finding Groups in Data – An Introduction to Cluster Analysis, Wiley Series in Probability and Mathematical Statistics, 1990.
27. Merz, C.J., Murphy, P.M., UCI Repository of Machine Learning Databases, http://www.ics.uci.edu, Irvine, CA, University of California, Department of Information and Computer Science.
28. Pyle, D., Data Preparation for Data Mining, Academic Press, 1999.
29. Hruschka, E. R., Ebecken, N.F.F. "A genetic algorithm for cluster analysis", Intelligent Data Analysis (IDA), Netherlands, v.7, n.1, 2003.
30. Triola, M. F., Elementary Statistics, 7th Edition, Addison Wesley Longman Inc., 1999.