Estimating Missing Attribute Values Using Dynamically-Ordered Attribute Trees

Jing Wang
Computer Science Department, The University of Iowa
jing-wang-1@uiowa.edu

W. Nick Street
Management Sciences Department, The University of Iowa
nick-street@uiowa.edu

Abstract

Classification performance can degrade if data contain missing attribute values. Many methods deal with missing information in a simple way, such as replacing missing values with the global or class-conditional mean/mode. We propose a new iterative algorithm to effectively estimate missing attribute values in both training data and test data. The attributes are selected one by one to be completed. For each attribute, the unknown values are predicted using a decision tree built using the other attributes from cases with known values of the attribute. The training set filled in this way is used to classify a tuning set whose prediction error rate decides which attribute is selected to be filled in the current iteration. Prediction error rate of the tuning set is recorded at each iteration to determine an optimal stopping point, as filling all missing values may lead to overfitting. The experiments show that the method generally outperforms several reasonable baseline methods and the ordered attribute trees method proposed by Lobo and Numao.

1 Introduction

In classification, the goal is to construct a classifier from a training set, and improve the prediction accuracy on the test set. However, many real world data contain missing attribute values, making it difficult to generate useful knowledge from training data, and to provide accurate prediction of test data as well. Therefore, many strategies to deal with incomplete data have been developed.

The simplest approach is to simply ignore instances with any missing values. This reduces the amount of information available, and therefore is not feasible in the case of high percentages of missing values. Another common approach is to replace missing values with the global or class-conditional mean/mode, but it does not make use of any dependencies of the missing attribute with other attributes. Quinlan [5] found that dividing cases with missing values among subsets when partitioning the training set in a decision tree, and combining all possible outcomes on a test case with missing values during classification, gives better classification accuracy than the other variations in decision tree learning. C4.5 [6] uses such a probabilistic approach to handle missing values in both training and test data. To further utilize the relations between attributes, Shapiro's method [8] constructs a decision tree for each attribute by using the subset of training data consisting of those instances whose values of the attribute are known. Problems arise when multiple missing values appear in an example. The *ordered attribute trees* (OAT) method [3] estimates missing values by building a decision tree for each attribute according to a specific order, determined by *mutual information* [4] between the attributes and the class. The method orders the attributes from low to high mutual information. The decision tree of an attribute is built using all the instances in the training set that have known values of the attribute, and using only the previously filled attributes.

A few methods have been designed to deal with missing values in test data specifically, such as *dynamic path generation* [9]. A decision tree is built for a test case using only those attributes whose values are present in the case. When a missing attribute value is found in a new case, the attribute is not used for branching when classifying the case. Since a whole classifier cannot be constructed beforehand, and instead a path is dynamically generated for each different test case, the method is computationally expensive. Saar-Tsechansky and Provost [7] address the problem of computation and storage expenses using hybrid reduced-model approaches. [2] proposes a *probabilistic attribute trees* approach, in which each attribute is predicted with a probability distribution using all of its dependent attributes. It solves the major problem of OAT, which is not taking into account all the dependencies between attributes, because they are built in an ordered manner.

The method proposed here handles missing values in both the induction phase and the prediction phase effectively. We aim at using the dependencies between attributes to predict missing data. Since decision trees are designed to select the most important attributes when determining the value of a target attribute, in this paper, we apply decision trees to predicting missing values of a target attribute using all the other attributes. An ordering for filling attributes is determined based on the predictive accuracy on a tuning set for a new tree constructed using the newly-filled attribute. By using tuning accuracy, we directly evaluate the predictive value of filling the attribute, which combines its importance in predicting the class (as in OAT) with our ability to correctly estimate its missing values. As opposed to OAT, in which the ordering is determined in advance, our method dynamically decides the ordering once one attribute is filled. We compare our method, termed Dynamically-Ordered Attribute Trees (DOAT), to OAT and several reasonable baseline methods. The experiments are conducted on different types of datasets with only nominal attributes.

2 Estimating Missing Values in Training Data

In our approach, missing attribute values are estimated iteration by iteration. At each iteration, one attribute is chosen, and all missing values of that attribute are filled. For each unselected attribute, an attribute tree AT_i is constructed by using i as the target and *all* other attributes as predictors. We choose not to include the (original) class attribute as a predictor, so that we can use the same model when filling values in the test data. Therefore, the information gained during the model construction matches the information available in testing, and the evaluation of test performance is more reliable.

A tuning set is set aside to estimate the classification performance. A classifier C_i having the original class attribute as the target is induced from the training data that has attribute i filled by its attribute tree AT_i. The attribute k whose corresponding classifier C_k produces the lowest prediction error rate on the tuning data is selected at this iteration. Accordingly, the missing values in the selected attribute k are filled by the corresponding attribute tree AT_k. In this manner, the ordering is determined dynamically after one attribute is selected to be estimated, making our method more flexible than the static ordering in OAT. Previously-induced attribute values are used for the new predictors, and also updated as new attributes are filled. Our experiments indicated that converging the previously-filled attributes gives better performance than with no convergence.

However, it is not always the case that filling all the missing values leads to the best performance, as can be shown from a number of experiments. To address this problem, we track the tuning error of the selected classifier C_k at each iteration. After all attributes are filled, we select the classifier that produces the lowest tuning error rate as the final classifier.

Unlike OAT, which fills the next attribute using only the already-filled attributes, our method uses all the other attributes when estimating missing values in an attribute. Doubtlessly, this may add noise due to the existence of missing values in those attributes that have not been filled. On the other hand, we are utilizing the maximum amount of information derivable from the data, thanks to the known values in the other attributes, which are useful despite those missing values. In addition, to reduce the interference of missing values in the other attributes during the construction of decision tree for the target attribute, missing values are replaced by modes. Likewise, the final chosen classifier is induced from the training data with some attributes filled and the others replaced by modes.

3 Estimating Missing Values in Test Data

It often happens in real data that both training data and test data contain missing values. Our method can work with both cases. Missing values in test data will also cause the class prediction to degrade even if training data are filled. Hence, before classifying test data, we estimate missing values in a similar manner as in training data. The selected attribute at the current iteration is regarded as the target attribute, and the attribute tree used for training data is again applied on test data. The ordering of filling attributes is still maintained when handling test data. If multiple missing values exist in a test case, we replace all of them (except for the current target) with their modes before the prediction of the target.

Furthermore, due to the fact that the final classifier in the training phase may be induced from the middle of the process where the training data are not completely filled, we do not use predictive trees for those attributes that have not been filled in the training phase. Attribute trees are built to estimate missing values in test data only for those attributes that have been filled in training data. As in the last phase, the missing values in the unfilled attributes are replaced by modes.

4 Experiments and Results

We compared four missing data imputation methods on nine datasets. Different percentages of missing values were artificially introduced into the datasets. To fairly compare the four methods, all the chosen datasets have only nominal attributes, because of the limitation of decision tree to only nominal values. Methods' performances are evaluated based on the error rate of class prediction on different datasets. Both attribute and class trees are built using the Weka [10] implementation of C4.5.

4.1 Experimental Setup

The nine datasets were selected from the UCI Machine Learning repository [1] and are summarized in Table 1.

Table 1: Summary of Datasets

Datasets	Instances	Attributes	Classes
Breast Cancer	699	9	2
Car Evaluation	1728	6	4
Chess	28056	6	18
Lymphography	148	19	4
Mushroom	8124	22	2
Soybean	307	35	19
SPECT	267	22	2
Tic-Tac-Toe	958	10	2
Voting	435	17	2

The distribution of missing data in our experiments was missing completely at random (MCAR). Different quantities, i.e. 10%, 20%, 30%, 40%, 50% and 60% of data were randomly turned into missing values, in order to observe the methods' performances under low missing rate and high missing rate. Each attribute, except for the class attribute which contains no missing, has the same amount of missing data.

The four imputation methods are listed below. All the methods are able to deal with missing data in both training data and test data.

- Mode: replace all missing values in training data and test data with the mode of the attribute in the training data, and then apply the decision tree method to the filled datasets.

- Attribute Trees (AT): construct a decision tree for each attribute to estimate missing data.

- Ordered Attribute Trees (OAT): Lobo's method.

- Dynamically-Ordered Attribute Trees (DOAT): our method.

4.2 Experimental Results

For each of the methods, the dataset was tested through five runs of 10-fold cross-validation. From the training data of every fold, 10% were extracted to be tuning data. We computed the average error rate of classification prediction over the five runs. The comparisons of methods are shown from the four representative datasets in Figures 1 to 4.

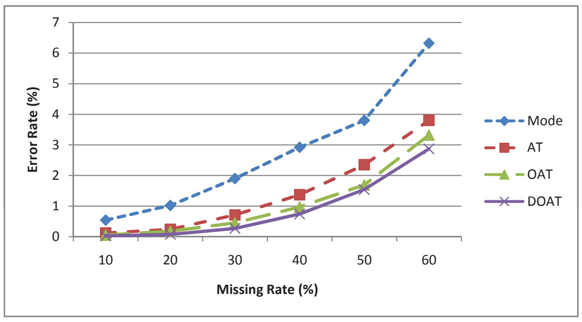

Figure 1: Mushroom

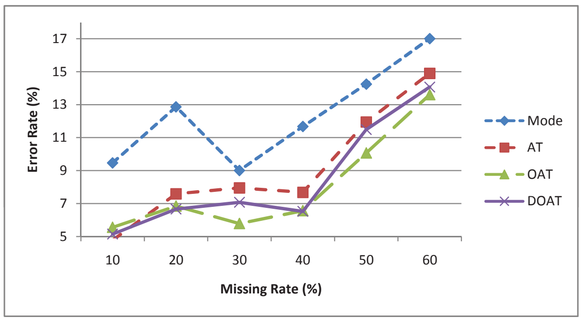

Figure 2: Voting

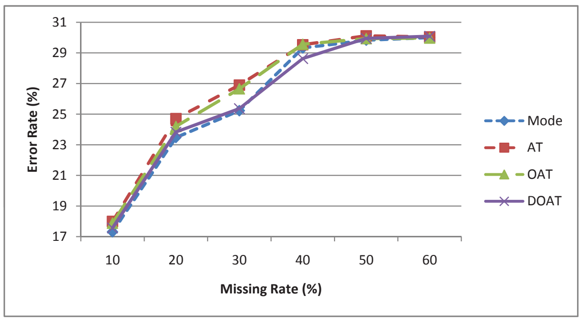

Figure 3: Car Evaluation

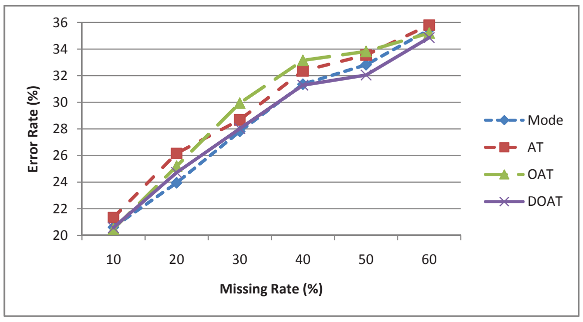

Figure 4: Tic-Tac-Toe

Figures 1 and 2 show two cases where all the attribute tree methods are significantly better than the mode method. DOAT seems to perform as well as OAT, but neither is consistently better. Moreover, DOAT outperforms AT. In contrast, Figures 3 and 4 show cases where the mode method is almost as good as DOAT, while AT and OAT are significantly worse. In this type of situation, DOAT beats the mode method in only a few tests.

From the results in the first cases, we conclude that the superior performance of the attribute trees methods is due to the relatively high correlation of the attributes to one another, allowing the attribute trees to perform well and thereby improve generalization. In addition, we notice that the number of attributes are relatively larger in these datasets. This reinforces the effect of taking advantage of relationships between attributes. On the contrary, in the second group, the correlation between attributes in those datasets is lower. Therefore, utilizing the dependencies between attributes to estimate the missing values does not have much effect. In fact, this feature inversely affects the performance in AT and OAT, as they are even worse than the mode method. Further, the number of attributes is smaller, leaving little space for the attribute trees methods to improve performance.

We compared DOAT with each of the other methods over all the nine datasets under all the missing rates. Results are shown in Table 2. Bold entries in the table are significantly worse than DOAT, and underlined entries are significantly better (2-sided pairwise t-test at 5%). The results can be summarized as follows:

- Mode: DOAT has 25 wins, 28 ties, and 1 loss.
- AT: DOAT has 23 wins, 28 ties, and 3 losses.
- OAT: DOAT has 19 wins, 30 ties, and 5 losses.

In general, DOAT outperforms all the three methods. Also, by looking at the comparisons of AT with OAT and DOAT, we may conclude that the attribute trees method with order is more effective than that without order, i.e., the one of filling all attributes at a single iteration.

DOAT works consistently well in different domains, although it has only a small advantage over the winners under the two different contexts. The consistent effectiveness may be because DOAT not only utilizes the correlation between attributes for estimating missing values, but also intends to select the best attribute at each iteration by evaluating the performance of a tuning set which simulates the result of test data. The ordering is more reliable in that the approach takes every aspect into account.

Table 2: Summary of Results

Dataset		Mode	AT	OAT	DOAT
Breast	10%	6.9	6.67	6.52	6.29
	20%	**7.9**	**7.1**	6.58	6.41
	30%	**10.5**	8.61	7.9	8.61
	40%	9.7	7.87	8.04	9.36
	50%	**12.39**	9.81	9.61	10.39
	60%	11.07	9.61	9.36	10.07
Car	10%	17.31	17.99	17.88	17.56
	20%	23.48	**24.71**	24.2	23.83
	30%	25.23	**26.9**	**26.66**	25.36
	40%	29.33	**29.54**	**29.57**	28.63
	50%	29.84	30.12	29.94	29.95
	60%	30.01	30.05	29.98	30.09
Chess	10%	**57.58**	**58.81**	**58.54**	57.37
	20%	65.18	**66.78**	**66.81**	65.39
	30%	70.07	**71.99**	**71.46**	69.92
	40%	73.95	**75.4**	**75.02**	73.99
	50%	76.6	**77.94**	**77.68**	76.63
	60%	78.25	**78.78**	**78.87**	78.22
Lymphography	10%	23.65	23.38	22.3	23.78
	20%	22.16	25.14	24.59	23.51
	30%	26.89	25.14	24.73	25.27
	40%	**41.62**	32.97	30.41	31.49
	50%	**35**	32.43	30.14	32.43
	60%	37.16	37.7	39.73	37.57
Mushroom	10%	**0.54**	**0.12**	**0.052**	0.027
	20%	**1.02**	**0.24**	**0.18**	0.074
	30%	**1.9**	**0.71**	**0.45**	0.27
	40%	**2.92**	**1.37**	**0.97**	0.74
	50%	**3.8**	**2.35**	**1.69**	1.54
	60%	**6.32**	**3.8**	**3.32**	2.87
Soybean	10%	**17.39**	13.03	14.07	13.88
	20%	**27.04**	16.42	16.55	18.7
	30%	**38.11**	**24.76**	23	23.06
	40%	**39.8**	31.47	31.6	29.9
	50%	**50.49**	39.67	37.92	40.26
	60%	**54.2**	49.84	47.56	47.88
SPECT	10%	20.22	**22.17**	21.95	20.3
	20%	20.6	21.2	22.1	21.5
	30%	17.38	18.88	**20.15**	17.9
	40%	30.79	**31.46**	**34.01**	29.96
	50%	22.02	20.9	22.62	22.1
	60%	37.53	36.7	36.48	36.1
Tic-Tac-Toe	10%	20.61	21.34	20.48	20.58
	20%	23.95	26.16	25.2	24.72
	30%	27.81	28.68	**29.94**	28.02
	40%	31.36	**32.36**	**33.15**	31.29
	50%	**32.82**	**33.55**	**33.82**	32.05
	60%	35.45	35.8	35.2	34.86
Voting	10%	**9.47**	4.78	5.56	5.15
	20%	**12.87**	7.59	6.85	6.67
	30%	**9.01**	7.95	5.79	7.08
	40%	**11.68**	7.68	6.57	6.53
	50%	**14.25**	11.95	10.07	11.49
	60%	**17.01**	**14.9**	13.61	14.07

5 Conclusions and Future Work

An approach dealing with missing attribute values in both training data and test data has been proposed. Experiments have been done by comparing it with the basic method that uses mode to fill missing values, an attribute trees method that builds a decision tree for each attribute, and a previously proposed *ordered attribute trees* method. We come up with the observation that our method performs consistently better in different domains of datasets.

The decision tree is the only learning scheme that is applied in the presentation of the method and in the tests, because of its nature to capture the relations between attributes. Further experimentation on data which contain both nominal and numeric attributes using other learners is expected. Our approach is mainly focusing on the estimation of missing values in training data, and the missing values in test data are simply predicted by the induced training data. We are interested in combining our approach in induction phase and other approaches in prediction phase, such as *dynamic path generation*, so as to effectively handle missing values in training data and test data respectively.

Finally, we intend to incorporate this approach into the design of active learning method for choosing missing values to be manually filled. If attribute values have an associated cost, then an important part of estimating the utility of acquiring a missing value would be an estimate of how well the value can be filled automatically.

References

[1] A. Asuncion and D. Newman. UCI machine learning repository, 2007. http://www.ics.uci.edu/∼mlearn/MLRepository.html.

[2] L. Hawarah, A. Simonet, and M. Simonet. *Mining Complex Data*, chapter Dealing with Missing Values in a Probabilistic Decision Tree during Classification, pages 55–74. Number 165 in Studies in Computational Intelligence. Springer-Verlag Berlin Heidelberg, 2009.

[3] O. O. Lobo and M. Numao. Ordered estimation of missing values. *PAKDD 1999: Proceedings of the Third Pacific Asia Conference on Methodologies for Knowledge Discovery and Data Mining*, pages 499–503, 1999.

[4] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986.

[5] J. R. Quinlan. Unknown attribute values in induction. *Proc. of 6th International Workshop on Machine Learning*, pages 164–168, June 1989.

[6] J. R. Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kaufmann Publishers, San Mateo, CA, 1993.

[7] M. Saar-Tsechansky and F. Provost. Handling missing values when applying classification models. *Journal of Machine Learning Research*, 8:1625–1657, 2007.

[8] A. Shapiro. *The role of structured induction in expert systems.* PhD thesis, University of Edinburgh, 1983.

[9] A. P. White. Probabilisitic induction by dynamic path generation in virtual trees. *Research and Development in Expert Systems*, III:35–46, 1987.

[10] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques.* Morgan Kaufmann, San Francisco, CA, 2nd edition, 2005.