Application of Data Decomposition to Incomplete Information Systems

Rafał Latkowski

Institute of Computer Science, Warsaw University
ul. Banacha 2, 02–097 Warsaw, Poland
rlatkows@mimuw.edu.pl

Abstract. Many developed classification methods and knowledge discovery software, that were research subjects for years, suffer from the lack of possibility to handle data with missing attribute values. To adapt existing classification methods to incomplete information systems, we propose a decomposition method that allows more appropriate missing value attributes handling. The decomposition method consists of two phases. In the first step data from original decision table are partitioned into subsets. In the second step, knowledge from those subsets, that in our case is classification hypothesis, is combined to achieve a final classification based on a whole original decision table. There were carried out some experiments in order to evaluate the decomposition method.

Keywords

Missing Attribute Values, Inductive Reasoning

1 Introduction

In recent years many researchers have faced the problem of missing attribute values [2,5,6,9,19]. Nowadays data acquisition and warehousing capabilities of computer systems are sufficient for wide application of computer aided knowledge discovery. Inductive learning is employed for various domains such as images, medical data, bank transactions and others. Due to various factors, those data suffer from impreciseness and incompleteness. The hard task of dealing with data imperfection in inductive learning methods was addressed in the area of data impreciseness by Pawlak in early 80's [14]. He proposed a Rough Set approach that made possible to precisely express facts about imprecise data in a formal way. The main concept of Rough Sets, indiscernibility relation, proved to be very useful for analysis of decision problems concerning objects described in a data table by a set of conditional attributes and a decision attribute [15,18]. In practical applications, however, the data table is often not only imprecise but also not complete, because some data are missing. Missing attribute values are frequently distributed not uniformly, but generated by underlying mechanism of investigated real world domain. Many developed classification methods and knowledge discovery software, that were research subjects for years, suffer from the lack of possibility to handle data with missing attribute values. To adapt existing classification methods to incomplete information systems, we propose a decomposition method that allows more appropriate missing value attributes handling.

Missing values can reduce soundness of inductive inference and result in decrease of classification quality. We expect the proposed method to reveal the pattern that governs the appearance of missing values in dataset. Such a decomposition avoids reasoning about missing values and makes possible the application of already developed methods and software, initially incapable of incomplete information systems processing. This method was experimentally compared with Quinaln's C4.5 method [17]. C4.5 is an example of inductive learning method that tries to deal with missing attribute values problem [16].

2 Method Description

The decomposition method was developed to meet certain assumptions. The primary aim of search for another method that could deal with missing attribute values was to find a possibility of adaptation for many of existing, well-known classification methods, that are not able to handle incomplete data. A development of solution, which makes possible to analyze incomplete information systems by already known and implemented classification methods, will reduce an effort necessary to construct a new software and framework for such data analysis from the beginning. The secondary aim was to cope with the problem of incomplete information systems without making an assumption of independent random distribution of missing values and without data imputation [4,7]. Many real world applications have showed that appearance of missing values is governed by very complicated dependencies, similar to the ones that we used to searching between decision and conditional attributes. This problem potentially could be solved by application of classification methods to predict value of missing data. However, the application of arbitrary method for data imputation produce a cycle of inductive reasoning steps and the feedback from such a cycle can drastically increase error rate of the classifier.

To meet those assumptions we created a decomposition method that allows for processing of incomplete information systems with use of methods that originally cannot handle missing attribute values. Roughly, the decomposition method consists of two phases. In the first phase incomplete information system, describing entire real-world interest domain, is decomposed into a number of complete information systems, consisting of objects and attributes that are taken from original information system. By accomplishing certain *filling patterns* we expect those complete information subsystems to describe some subproblems of investigated real-world problem, in which the mechanism of missing values appearance is similar. In the second phase some knowledge fusion is necessary in order to merge hypotheses about approximated real-world concept previously decomposed into subproblems. Application of conflict resolving methods is required to merge knowledge from a number of classifiers. As an example of conflict resolving method one may take voting mechanism or similar classifier to the one used in classification of decision subtables obtained from the decomposition phase.

3 Algorithm Description

The decomposition method consists of two phases. In the first one data from original decision table are partitioned into subsets. In the second step, knowledge from those subsets, that in our case is classification hypothesis, is combined to achieve a final classification based on a whole original decision table.

The aim of the first step is to decompose data according to regularities in missing value distribution in data table. The result of decomposition is a number of subtables that are free from missing values and contain as much meaningful data as possible. To provide a mechanism for extraction of data regularities we must apply some kind of pattern extraction that is customized for this task.

Definition 1. Filling pattern.

Let $a_i \neq *$ be a *filling descriptor*. An object satisfies filling descriptor $a_i \neq *$, if value of the attribute a_i for this object is not missing, otherwise object does not satisfy filling descriptor. *Filling pattern* is a conjunction of filling descriptors. An object satisfies filling pattern $a_{k_1} \neq * \wedge \ldots \wedge a_{k_n} \neq *$ if values of attributes $a_{k_1} \ldots a_{k_n}$ for this object are not missing.

Filling patterns are used to discover regular areas in data that contain no missing values. Once we have a filling pattern, we can identify it with a subtable of original data table. Such a subtable consists of attributes that are elements of filling pattern and contains all objects that satisfy this pattern. With such a unique assignment of filling patterns and subtables of original data we can think of result of decomposition step as of set of filling patterns. The decomposition itself becomes a problem of covering data table with patterns, as investigated in [12,13].

Standard approach to generation of covering pattern set is based on greedy strategy. The set of patterns is iteratively extended until all objects from data table are *covered*. The object is covered when exists at least one pattern in generated set that is satisfied by this object. We can outline this algorithm as follows:

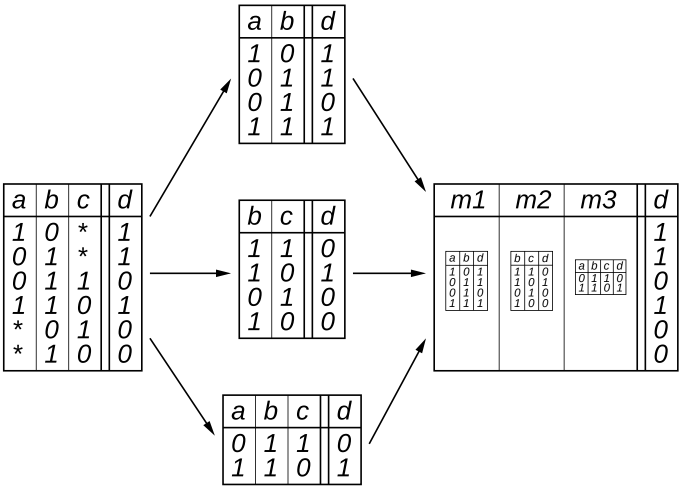
1. Extract best filling pattern according to some criteria,
2. Remove objects that satisfy extracted pattern from data table,
3. Repeat 1–2 until all objects are covered by some pattern.

Extraction of best pattern is subordinate to some criteria. In experiments we used a number of criteria that are described in the next section. Also, we carried out some experiments with modified algorithm, where object removal was replaced by various object's weighting techniques. Such a modifications, however, do not improve results so in the final experiments only the above algorithm was used. Pattern extraction was performed with the help of a genetic algorithm that was customized for this task [11]. Deterministic methods [12] were not applicable. Criteria used for selection of best filling patterns are based not only on width and height of pattern, as described in the next section. Customization of genetic algorithm, some data compression and partial result caching permitted results comparable to results of deterministic pattern extraction methods.

Decomposition provides a set of subtables uniquely determined by filling patterns. Those subtables consist of a subset of attributes from original decision table and a subset of objects. Sets of objects and attributes in such subtables are different, but usually not disjoint. Data subtables are free from missing attribute values, so in such tables a classifier that is not able to handle missing values can be applied. We expect that discovered regularities in missing values' distribution are meaningful for inductive learning and partition of interest domain into subproblems in accordance to its hidden nature. The classifier induced over such subset of data should be able to appropriately classify new objects satisfying its filling pattern. When new object is classified, there may occur a situation that more than one, previously selected in decomposition phase, filling pattern is satisfied. Thus, there exists more then one classifier that is capable of classifying such object. In the second phase of decomposition a knowledge integration is required in order to obtain a classifier valid for a whole domain of objects. The problem is how to combine knowledge from different classifiers to obtain one decision. The first approach to this problem is to apply a voting mechanism. However, due to reduction of positive region (see [8]) in decision subtables voting mechanism is not enough. This fact became apparent during the early stage experiments. The more precise method is to apply a classifier induction algorithm similar to the one that is used for subtables. Application of another classifier at the top of subtable-based classifiers allows to fine tune system's answer in a non-linear way. We can imagine a situation that this top classifier can completely change assignment of object's decision obtained from subtable based classifiers. With such a proceeding confidence level of a particular subclassifier depends on filling patterns that are satisfied.

We can briefly summarize the algorithm of classifier induction as follows:

1. Decomposition: Greedy generate filling patterns that meet certain properties,
2. Split data into subtables according to filling patterns,
3. Induce classifier from subtables,
4. Integration: Induce classifier from answers of classifiers based on subtables.



a	b	c	d
1	0	*	1
0	1	*	1
0	1	1	0
1	1	0	1
*	0	1	0
*	1	0	0

a	b	d
1	0	1
0	1	1
0	1	0
1	1	1

b	c	d
1	1	0
1	0	1
0	1	0
1	0	0

a	b	c	d
0	1	1	0
1	1	0	1

m1	m2	m3	d
			1
			1
			0
			1
			0
			0

Fig. 1. Illustration of the decomposition method idea.

The classification of new objects is a two step process that requires both layers of previously induced classifiers. Objects that are classified could have missing attribute values, so we have to apply processing scheme similar to the previous one. Classification of such an object proceeds as follows:

1. Check, which filling patterns are satisfied,
2. Compute answers from subclassifiers (i.e. classifiers based on subtables),
3. Compute final decision from top classifier that bounds answers from subclassifiers.

The classification employs methods that are originally unable by itself to process data with missing attribute values. However, the described decomposition method allows to work around this problem. The presented solution does not completely eliminate problem of missing values, because missing values return as missing answers coming from classifiers based on subtables. This problem could be eliminated by using another method of resolving conflict between subtable based classifiers. It is our strong believe that discovered blocks of complete data are relevant and helpful for inductive learning of dependencies between conditional attributes and decision attribute. Missing answers contain information not exactly about missing attribute values, but rather about patterns in missing data distribution. In such patterns missing attribute values should be more comparable, than in whole data. Empirical results showed that this assumption can by correct, but we should cautiously choose proper methods of filling pattern generation.

4 Decomposition Criteria

Subsets of original decision table must meet some requirements in order to achieve good quality of inductive reasoning as well as to be applicable in case of methods that cannot deal with missing attribute values. Original decision table is partitioned into smaller decision tables. Those smaller tables are not necessarily disjoint, but we expect them to exhaustively cover input table. They should contain no missing attribute values. It is obvious, that the quality of inductive reasoning depends on a particular partition and some partitions are better than others. From one point of view quality of learning depends on the number of examples. It is proven that inductive construction of concept hypothesis is only feasible, when we can provide enough number of concept examples. A strict approach to this problem can be found in [20] where Vapnik-Chervonenkis dimension is presented as a tool for evaluation of examples number requirements. From the second point of view inductive learning tries to discover a relationship between decision attribute and conditional attributes. A precise description of concepts in terms of conditional attributes values is required to achieve good quality of classification. Without an attribute, which value is important to concept description it is impossible to accurately approximate a concept. This yields an assumption that subtables derived from partition should also have sufficient number of attributes for inductive learning. It clearly suggests that we should not blindly select partition, especially because the partition has strong influence on reasoning quality.

We should have some measurements in order to compare each partition to others and select the best one. By measuring some properties of decision subtables we can evaluate correspondence between such tables and the decomposition of real world problem into subproblems. Several approaches were applied to estimate the filling pattern correspondence. Among them are:

- Traditional size based evaluation,
- Predictive quality — a real reasoning quality factor that can be evaluated by applying classification algorithm to a subtable.

Traditional size based evaluation is related to width and height of generated subtable. By width we understand a number of attributes in such a table and by height a number of objects. These values correspond to width and height of generated filling pattern. We can combine these two values into fitness function that can be used in genetic algorithm. Standard approach is to evaluate fitness of pattern as a product of width and height ($q = w \cdot h$, where w stands for width of pattern and stands h for height). We can also apply modified fitness function from family $q = w^{\alpha} \cdot h$. However empirical evaluation showed, that there are no big differences between such functions. The observation was, that in searched space of filling patterns there were several patterns with similar size based fitness value, but with drastically different results in quality of classification. This implies the second approach.

	all	exact	$w \cdot h$	$w \cdot h \cdot p$	$w \cdot h \cdot p^2$	$w \cdot h \cdot p^4$	$w \cdot h \cdot p^8$	p
att	17.15	3.90	3.94	4.00	4.10	4.15	4.09	5.35
ban	56.80		5.33	8.09	8.28	9.01	10.08	22.14
cmc2	6.96	2.00	2.00	2.41	2.59	2.91	3.51	3.92
dna2	7.80		1.06	2.54	2.63	2.61	3.55	7.08
hab2	5.00	3.83	3.69	3.20	3.01	2.78	2.50	1.84
hco	164.65	5.03	5.46	5.80	6.16	6.89	9.70	67.54
hep	18.48	3.84	4.03	4.12	4.30	4.70	5.30	8.27
hin	25.97	4.11	3.83	4.91	5.74	7.21	8.77	13.22
hyp	17.96	2.00	2.01	2.01	2.02	2.01	2.01	4.55
pid2	6.77	2.97	2.98	2.99	3.11	3.41	3.48	3.89
smo2	4.00	2.00	2.00	2.42	1.80	1.33	1.39	2.14
tumor	6.40	1.99	2.17	2.53	3.03	3.58	3.84	4.37

Table 1. Average number of filling patterns.

The size based fitness function is only an estimation of relevance for classifier induction. Instead of size based fitness function we can estimate relevance by measuring predictive quality — quality of classifier induced from subset of data determined by this filling pattern. We can easily incorporate such a factor into fitness function of filling pattern. To measure the influence of predictive quality on decomposition relevancy we can test some functions from family $q = w \cdot h \cdot p^{\alpha}$, where p stands for predictive quality and α is the tuning coefficient. We can also completely eliminate a size based factor and put $q = p$ fitness function. Empirical evaluation shows that this last function behaves best.

5 Empirical Evaluation

There were carried out some experiments in order to evaluate decomposition method and its components, such as genetic algorithm for binary pattern extraction. Results were obtained from the average of classification quality from 100 times repeated five-fold Cross-Validation (CV5) evaluation. This testing method was introduced to assure preciseness in measuring the number of generated patterns. The *C4.5* method was used as a classifier and tests were performed with different decomposition approaches as well as without using decomposition method at all. The *WEKA* software system [3], which contains re-implementation of Quinlan's C4.5 Release 8 algorithm in Java, was utilized in experiments. Data sets from UCI [1] machine learning repository were used for evaluation of the decomposition method Selected data sets contain missing values in range from 14.1% to 89.4% of all values in data.

First group of results, presented in Table 1, shows efficiency of filling pattern generation. The numbers in the table are averages over 100 CV5 steps

	C4.5	$w \cdot h$	$w \cdot h \cdot p$	$w \cdot h \cdot p^2$	$w \cdot h \cdot p^4$	$w \cdot h \cdot p^8$	p
att	52.55	+2.39	+3.22	+5.23	+7.79	+9.39	+10.78
ban	62.14	+3.68	+6.37	+8.29	+10.55	+12.77	+14.16
cmc2	45.72	-0.80	+1.56	+2.89	+4.37	+5.61	+5.69
dna2	86.84	-6.11	-0.64	+0.11	+0.32	+1.55	+2.23
hab2	71.54	-3.47	-2.40	-0.83	+1.36	+3.13	+4.44
hco	81.68	-2.18	+0.28	+1.92	+3.34	+4.31	+4.32
hep	80.12	-4.24	-0.59	+1.36	+3.58	+5.17	+6.41
hin	70.47	-0.51	-0.31	+0.02	+0.51	+0.63	+0.06
hyp	95.82	+0.90	+0.94	+0.97	+0.98	+0.99	+1.27
pid2	60.81	+1.17	+1.38	+3.03	+5.43	+6.30	+7.48
smo2	60.75	-4.61	-2.83	+4.73	+7.72	+8.20	+8.91
tumor	38.89	-2.61	+1.28	+3.31	+4.16	+5.00	+4.41

Table 2. Comparison of experiments' results.

of the numbers of patterns generated. The column entitled *exact* corresponds to a method, which is an implementation of exhaustive exact method that checks all 2^n possible patterns and selects the best one. Such a exhaustive search is very time-consuming and therefore was evaluated only for $q = w \cdot h$ fitness function and data that contain no more than 25 attributes. The results presented in following columns were obtained with the help of genetic algorithm using various fitness functions. The corresponding fitness function is described in the header of each column. To make comparison easier we also provide the number of *all* filling patterns that were present in the data. As we can see, the results of exhaustive search are similar to those of genetic algorithm with the same fitness function. Results for other fitness function show that — at least for some data sets — the use of predictive quality does not drastically increase the number of selected patterns.

The second group of results presents a classification qualities in comparison with the C4.5 method. The decomposition method, which uses the predictive quality as a fitness function overcomes results of the C4.5 method. As the importance of predictive quality in filling pattern's evaluation function increases, we can observe a continuous increase of the classification quality. For some data sets other fitness functions achieve good results too. We should consider that evaluation of predictive quality is very time-consuming, in spite of partial results' caching and other optimizations. The last presented column corresponds to experiments with predictive quality as a fitness function. These results nearly overcome all other approaches, including the one the is embed in the C4.5 method of missing attribute values handling.

6 Conclusions

Most of existing methods are incapable of incomplete information systems processing. The decomposition method proved that it is an efficient tool for adapting existing methods to data with missing attribute values. It can be applied to various algorithms of classifier induction to enrich them with capabilities of incomplete information systems processing. In comparison with other approaches to missing attribute values handling [10,21], this method can be applied to a broad group of inductive reasoning algorithms. This unique property of the decomposition method allows an efficient transforming of already developed software systems that could not handle missing data. The week point of this method is the computational complexity of predictive quality evaluation. The answer how to better and less time-consuming evaluate a filling pattern suitability to the decomposition process, remains an open question for a further research.

Acknowledgments

I wish to thank professor Andrzej Skowron for a great support while writing this paper. This work was partially supported by the Polish State Committee for Scientific Research grant No. 8T11C02519.

References

1. C. L. Blake and C. J. Merz. *UCI Repository of machine learning databases.* http://www.ics.uci.edu/~mlearn/MLRepository.html, Univerity of California, Departament of Information and Computer Science, Irvine, CA, 1998.
2. N. H. Bshouty and D. K. Wilson. On learning in the presence of unspecified attribute values. In *Proceedings of the Twelfth Annual Conference on Computational Learning Theory, COLT'99*, pages 81–87. ACM, 1999.
3. E. Frank, L. Trigg, and M. Hall. *Weka 3.1.9, Waikato Environment for Knowledge Analysis.* http://www.cs.waikato.ac.nz/ml/weka, The University of Waikato, Hamilton, New Zealand, 2000.
4. Y. Fujikawa and T. Ho. Scalable algorithms for dealing with missing values. 2001.
5. S. A. Goldman, S. Kwek, and S. D. Scott. Learning from examples with unspecified attribute values. In *Proceedings of the Tenth Annual Conference on Computational Learning Theory*, pages 231–242, 1997.
6. S. Greco, B. Matarazzo, and R. Słowiński. Rough sets processing of vague information using fuzzy similarity relations. In C. S. Caldue and G. Paun, editors, *Finite vs. infinite: contribution to an eternal dilemma*, pages 149–173, Berlin, 2000. Springer-Verlag.
7. J. W. Grzymała-Busse and M. Hu. A comparison of several approaches to missing attribute values in data mining. In W. Ziarko and Y. Y. Yao, *Proceedings of 2nd International Conference on Rough Sets and Current Trends in Computing, RSCTC-2000*, pages 180–187, 2000.

8. J. Komorowski, Z. Pawlak, L. Polkowski, and A. Skowron. Rough sets: A tutorial. In S. K. Pal and A. Skowron, editors, *Rough Fuzzy Hybridization. A New Trend in Decision Making*, pages 3–98. Springer-Verlag, 1998.
9. M. Kryszkiewicz. Properties of incomplete information systems in the framework of rough sets. In L. Polkowski and A. Skowron, editors, *Rough Sets in Data Mining and Knowledge Discovery*, pages 422–450. Physica-Verlag, 1998.
10. W. Z. Liu, A. P. White, S. G. Thompson, and M. A. Bramer. Techniques for dealing with missing values in classification. In X. Liu, P. Cohen, and M. R. Berthold, editors, *Advances in Intelligent Data Analysis*, pages 527–536. Springer-Verlag, 1997.
11. Z. Michalewicz. *Algorytmy genetyczne + struktury danych = programy ewolucyjne*. WNT, 1999.
12. S. H. Nguyen. *Regularity Analysis and its Application in Data Mining*. Praca doktorska, Warsaw University, Faculty of Mathematics, Computer Science and Mechanics, 1999.
13. S. H. Nguyen, A. Skowron, and P. Synak. Discovery of data patterns with applications to decomposition and classification problems. In L. Polkowski and A. Skowron, editors, *Rough Sets in Knowledge Discovery*, volume 2, pages 55–97, Heidelberg, 1998. Physica-Verlag.
14. Z. Pawlak. *Rough sets: Theoretical aspects of reasoning about data*. Kluwer, Dordrecht, 1991.
15. L. Polkowski, A. Skowron, and J. M. Żytkow. Tolerance based rough sets. In T. Y. Lin and A. M. Wildberger, editors, *Soft Computing*, pages 55–58. San Diego Simulation Councils Inc., 1995.
16. J. R. Quinlan. Unknown attribute values in induction. In A. M. Segre, editor, *Proceedings of the Sixth International Machine Learning Workshop*, pages 31–37. Morgan Kaufmann, 1989.
17. J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufman, San Mateo, 1993.
18. A. Skowron. Boolean reasoning for decision rules generation. In J. Komorowski and Z. Ra, editors, *Proceedings of the 7th International Symposium ISMIS'93, Trondheim, Norway*, pages 295–305. Springer-Verlag, 1993.
19. A. Skowron. Extracting laws from decision tables. *Computational Intelligence*, 11 (2):371–388, 1995.
20. V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, New York, 1995.
21. S. M. Weiss and N. Indurkhya. Decision-rule solutions for data mining with missing values. IBM Research Report RC-21783, IBM T. J. Watson Research Center, 2000.