14

Decomposition in Data Mining - A Case Study

Objectives:

- Decomposition is a tool for managing complexity in data mining and enhancing the quality of knowledge extracted from the large databases.
- A typology of decomposition approaches applicable to data mining is presented.
- One of the decomposition approaches, the structured rule-feature matrix, is used as the backbone of a system for informed decision making.
- As a new discipline, data mining draws from other areas such as statistics, machine learning, database retrieval, pattern recognition, and high-performance computing.
- In this section, numerous decomposition approaches are defined and applied for effective knowledge discovery and decision making.
- Decomposition has been discussed in the data mining literature, however, largely in the context of distributed learning.
- There are two basic approaches to data mining
- Direct mining of data sets
- Mining of transformed data sets
- The following two forms of decomposition in space are considered in this section
- Feature set decomposition
- Object set decomposition
- One of the most meaningful applications of decomposition in data mining is hybrid modeling.

Abstract. Decomposition is a tool for managing complexity in data mining and enhancing the quality of knowledge extracted from the large databases. A typology of decomposition approaches applicable to data mining is presented. One of the decomposition approaches, the structured rule-feature matrix, is used as the backbone of a system for informed decision making. Such a system can be implemented as a

S. Sumathi and S.N. Sivanandam: *Decomposition in Data Mining - A Case Study*, Studies in Computational Intelligence (SCI) **29**, 303–313 (2006)
www.springerlink.com

decision table, a decision map, or a decision atlas. This case study is taken from A. Kusiak, Intelligent Systems Laboratory University of Iowa, 2001.

Data mining is concerned with discovery of patterns, associations, rules, and other forms of knowledge in data sets. This knowledge is automatically extracted from data rather being formulated by a user as it is done in traditional modeling approaches, e.g., statistical or optimization modeling. As a new discipline, data mining draws from other areas such as statistics, machine learning, database retrieval, pattern recognition, and high-performance computing.

In many applications, data is automatically generated and therefore the number of objects to be mined can be large. The time needed to extract knowledge from such large data sets is an issue, as it many easily run in days, weeks, and beyond. One way to reduce computational complexity of knowledge discovery with data mining algorithms and decision making based on the acquired knowledge is to reduce the volume of data to be processed at a time, which can be accomplished by decomposition. In this section, numerous decomposition approaches are defined and applied for effective knowledge discovery and decision-making. Besides simplifying competition, decomposition facilities dynamic extraction of knowledge that can be used for real-time decision making.

14.1 Decomposition in the Literature

Decomposition has been discussed in the data mining literature, however, largely in the context of distributed learning. This research emphasizes the use of decomposition to enhance decision making rather than learning. Grossman *et al.* 1999 outlined fundamental challenges for mining large-sale databases, with one of them being the need to develop distributed data mining algorithms. Guo and Sutiwaraphun 1988 described a meta-learning concept called *knowledge probing* to distributed data mining. In knowledge probing, supervised learning is organized in two stages. At the first stage, a set of base classifiers is learned in parallel from a distributed data set. At the second stage, the relationship between an attribute vector and the class from all of the base classifiers is determined. Zaki *et al.* 1999 discussed a project called SPIDER that uses shared-memory multiprocessors systems (SMPs) to accomplish parallel data mining on distributed data sets. Cluster analysis provides the basic theory and algorithms for decomposition. Some of the most efficient clustering algorithms are presented in Kusiak *et al.* 2000.

14.1.1 Machine Learning

Bazan 1998 categorized the existing learning algorithms as follows:

- Decision tree

Decision rule
Inductive logic programming and
Rough set algorithms

14.2 Typology of Decomposition in Data Mining

There are two basic approaches to data mining

Direct mining of data sets
Mining of transformed data sets

The first approach is most often applied for mining data sets that can be processed in an acceptable time by the existing data mining algorithms. Transforming data sets before mining is intended either for large data sets or data sets with special properties, e.g., hybrid data discussed next. One of the most useful forms of data transformation is decomposition, which may take place in space and time. The area of decomposition in time is extensive and has received rather broad coverage in the literature though it is beyond the scope of this section.

The following two forms of decomposition in space are considered in this section

Feature set decomposition
Object set decomposition

The feature set decomposition is further classified into:

- Content-based decomposition: The feature set is decomposed into mutually exclusive or partially overlapping subsets with the same decision D used for each subset. The feature origin, availability, and any other criterion could drive the content of each feature set.
- Intermediate-decision decomposition: In some applications feature values are generated over time. In addition the downstream features may dependent on the upstream features.
- Feature-type decomposition: some of the existing rule extraction algorithms are intended for specific types of feature, e.g., discrete value features.
- Feature relevance decomposition: Features may show various degree of relevance to the outcome, measured with statistical metrics (e.g., correlation) and context relationship, which is more tacit and difficult to measure (e.g., the impact of outside temperature on computer energy consumption).

The object set decomposition is further classified as:

- Object content decomposition: Objects are grouped according to time interval, origin, applicability, and so on.
- Decision value decomposition: the set of objects is split into subsets according to the decision value.
- Feature value decomposition: the objects (and possibly features) are partitioned into subsets based on the value of selected features.

14.3 Hybrid Models

One of the most meaningful applications of decomposition in data mining is hybrid modeling. A hybrid model is a collection of models of different types, for example, models developed on the first principles and models constructed from the knowledge extracted by machine learning algorithms. Hybrid models are often built because of different degree of understanding of the modeled process, availability of data, and other application specific limitations.

The need for hybrid has been motivated by numerous engineering and medical applications. A typical process, e.g., a semiconductor manufacturing process or disease management process, involves stages that are well understood due to available models and stages that are only loosely known. This lack of process knowledge is likely behind unwanted events in industrial processes (e.g., products below expected quality level) and in medicine (e.g., premature patient's death).

This section meets the need for hybrid models involving both well-defined models and knowledge models derived from the data collected while observing an actual process. During the process operations, both desirable and adverse events occur, and therefore, the data collected is transformed into knowledge that can be used to minimize or even prevent adverse events from happening.

A domain of interest captured by the hybrid decision making model in Fig. 14.1 is decomposed into three stages. The model allows for bi-directional reasoning, which is important for in-depth understanding of the model process.

The rule-structuring algorithm in Kusiak 2000 organizes the knowledge extracted by different learning algorithms into decision tables that will be integrated with analytical models as well as models of other types are illustrated in Fig. 14.2. The structured knowledge is "packaged" in the form of decision tables that can be combined in other constructs such as decision maps, and those in turn in decision atlases. These constructs will increase transparency and effectiveness of the decision-making process.

A decision table provides decision basis (e.g., decision rules, rule support, rule coverage, etc.) and justification for the decision (e.g., historical cases supported by the decision rule) as symbolically illustrated in Fig. 14.2.

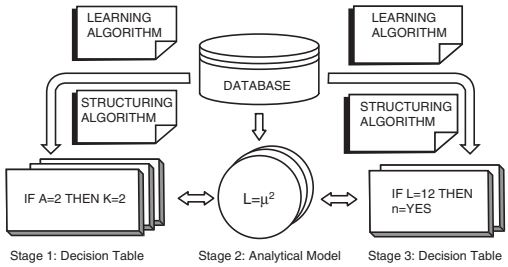

Fig. 14.1. Hybrid decision-making process

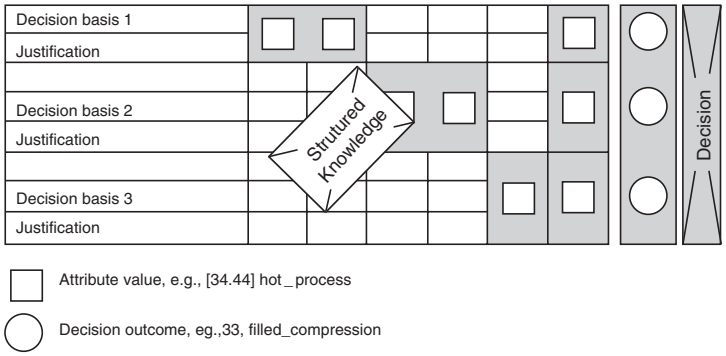

Fig. 14.2. Example of a decision table

The entries of the decision table in Fig. 14.2 are attribute values generated by a learning algorithm. Each entry may represent a singular numerical or symbolic value, a bounded range of values, an unbounded value range (inequality), and so on. Each decision basis contains feature values, while justification characterizes the decision basis, for example, it may contain the decision confidence and risks associated with a decision. The simplest way of making decisions with a decision table is to match the values of an object with unknown outcome to an appropriate row of the decision table. Other ways of making robust decisions with orthogonal algorithms are discussed in Kusiak *et al.* 2000.

An Example for a typical hybrid process:

Consider a five-stage process in which the models at stages 1, 3, and 5 are unknown, while stages 2 and 4 are modeled with known functions, F_1 and F_2 (see the functions below and Figure). During a three-month period, for process stages, 1, 3, and 5 three data sets containing numerous observations (objects) on selected features $f_{i,}I = 1, \ldots, 9$ and the decision D have been collected. A learning algorithm has extracted three sets 1, 2, and 3 of decision rules shown next.

Rule set 1
IF $f_1 = 2$ AND $f_2 =$ Low THEN $f_3 = 4$
IF f_2 in [2.1,4] THEN $f_3 = 5$
Function F1
$F_4 = 3.1 + (f_3 - 3.1)^3$
Rule set 2
IF $f_4 < 8.5$ AND $f_5 =$ High THEN $f_6 = 8.4$

IF f_5= Low THEN $f_6 = 12.4$
Function F2
$f_7 = \text{In } (f_6 + 2.9)^{1/2}$
Rule Set 3
IF $f_7 < 1.3$ THEN D = Good
IF $f_8 >= 3.3$ and $f_{9=}$ Positive THEN D = Bad

The content of each of the three rule sets may change frequently while the two functions remain the same. In fact, the two functions, F_1 and F_2 could interface with alternative rules generated from the corresponding data sets.

When the number of rules is large, their collective interactions are difficult to understand. For better usability of decision rules they will be organized by the rule-structuring algorithm in Kusiak 2000 and represented as decision tables, decision maps, or decision atlases.

The decomposition approach discussed in this section offers the following advantages.

- Ease of model construction and understanding: the data set is partitioned into independent subsets (data sets 1, 3, and 5 in Fig. 14.3) and therefore the rule induction is simplified.
- Support of the evolutionary computation concept: the basic the premise of the discussed approach is that the models are separable and they change with different frequency. Evolutionary computation algorithms can be involved in individual models as well as controlling the evolution of the overall model;
- Increased model structure stability: Only one sub-model (a rule set or a function in Fig. 14.3) at a time is usually modified.
- Ease of data acquisition and model maintenance: As the scope of data at each stage is limited it is easier to acquire the data and maintain the component models;
- Reuse of known models and dependencies: Rather than building a new model from scratch, the model is built around existing component models, e.g., functions and neural network models;
- Representation of alternative solutions with positive and negative rules: The role of negative rules in decision making is discussed in Tsumoto *et al.* 2000.

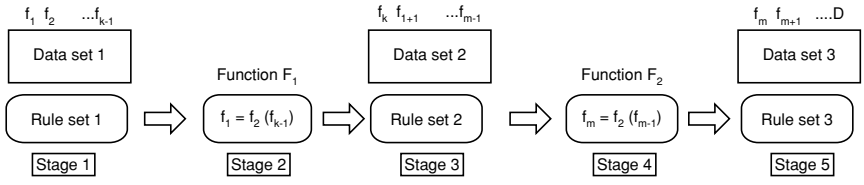

Fig. 14.3. Example process with three rule sets linked by two functions

14.4 Knowledge Structuring

One of the main reasons for extracting knowledge from data sets is to use it for decision making, which has not received sufficient attention in the literature. Most of decision-making algorithms are rather simplistic and usually based on partial or full-matching schemes. Many users have difficulty accepting decision rules that are nonintuitive and algorithms making decisions based on nontransparent matching. This section addresses this important gap in the presentation of knowledge for effective decision making.

The knowledge extracted by a learning algorithm is usually in the form of decision rules that make predictions at some level of accuracy, typically far from perfect. Decision rules might be numerous, the relationships discovered may be flawed, and their meaning might be difficult to understand, and so on. In other words, users have certain expectations for the knowledge discovered that are outside of the scope of learning algorithms. The rule-structuring models and algorithms to be developed in this research will meet these expectations. They will be used both for supporting the user view as well as autonomous decision making.

An example for the rule-structuring concept

Three different learning algorithms A1–A3 were used to extract eight decision rules R1–R8 from a data set. These rules R1–R8 are represented as the rule-feature matrix in fig.14.4. To simplify our considerations the information pertinent to each rule such as support, classification quality and so on has not been included. Though the rule set in the Fig. 14.4 is small, its analysis is not simple. Transforming the rule-feature matrix into the structured matrix significantly improves interpretation of the rule set. The rule-structuring algorithm Kusiak 2000 generated the matrix in fig. 14.4A from the one in fig. 14.4 by removing two rules R7 and R8 due to their dissimilarity with the rules R1 through R6 and changing the sequence of the remaining rows and columns.

The content of the matrix in fig. 14.4A is structured and it allows drawing numerous conclusions, for example:

- The decisions D = High, Medium, and Low are totally separated by features;
- The rules R3 and R6 generated by algorithms A2 and A3 are equivalent;
- The decision D = Low can be reached in two alternative ways, using the features values f1 = {B, C, D} and f2=a, or f1= {E, F} and f2= b.

The example illustrates only a few of numerous user's requirements that can be incorporated in the rule-structuring algorithm, such as:

- Matrix structure: different structures of the rule-feature matrix may be considered, e.g., block-diagonal (see fig. 14.4A), block-diagonal matrix with overlapping features, block-diagonal matrix with overlapping

f_1	f_2	f_3	f_4	f_5	D	Rule	Algorithm
{B, C, D}	a				Low	R1	A1
			>9		Medium	R2	A2
			<2	(2, 5]	Low	R7	A3
		(2, 6]		=<8	High	R3	A2
{E, F}	b				Low	R4	A3
{C, F}		<4			Medium	R8	A1
			>9		Medium	R5	A1
		>=2		[1, 3]	High	R6	A3

Figure 4. Rule-feature matrix.

f_3	f_5	f_2	f_1	f_4	D	Rule	Algorithm
(2, 6]	=<8				High	R3	A2
>=2	[1, 3]				High	R6	A3
		a	{B, C, D}		Low	R1	A1
		b	{C, F}		Low	R4	A3
				>9	Medium	R2	A2
				>9	Medium	R5	A1

Structured rule-feature matrix

Fig. 14.4. &A: Rule-Feature matrix and Structured rule-feature matrix

rules, triangular (for dependency analysis among rules), L-shape matrix, T-shape, etc.

- Differentiation of decision on features: Each decision value is associated with an independent subset of features;
- Differentiation of decision on features values: Any two decision values are discernable on unique subset of feature values;
- Inclusion of user-selected features: A user may have her/his preferences in terms of the features to be included in the selected rules, exclusion of some features, presence of the minimum set of features, and so on;
- Contrasting positive rules against negative ones is valuable in decision making in some applications.

14.5 Rule-Structuring Model

The rule-structuring problem can be represented as m-partite graph with each node representing a set of rules called *candidate rule set*. The rules contained in each candidate rule set share some common property, e.g., same decision value, common features set, which depends on the objective of the rule-structuring problem. The candidate rule sets could be defined for the rules extracted a

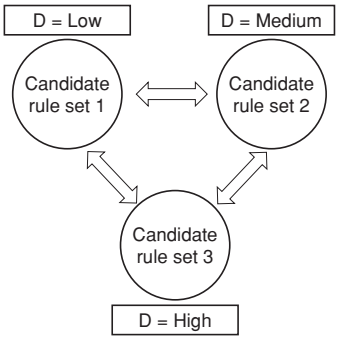

Fig. 14.5. Graph representation of the rule-structuring problem for the rules in Fig. 14.4.

specific data set, e.g., data set 1 in fig. 14.3 or across different data sets, e.g., data set 1 and 3 in fig. 14.3. Candidate rules sets may include rules extracted by different learning algorithms. An arc of the$m$-partite graph represents relationships between the corresponding nodes, e.g., a distance.

Transforming the unstructured matrix in fig. 14.4 into the structured matrix in fig. 14.4A calls for development of a rule-structuring model that could be solved with the standard or a specialized algorithm. Both modeling and solution alternatives are explored in this section. The rule-structuring model is illustrated in the next example.

Example: Represent the eight rules in fig. 14.5 with the 3-partite graph in fig. 14.5.

Each of the three candidates rule sets in fig. 14.5 contains rules with the same outcomes. For example, the candidate rule set 1 includes the rules R1, R4, and R7 from fig. 14.4. The arrows in fig. 14.4 symbolize relationships (e.g., distances) among the rules belonging to different candidate rules sets. In fact these relationships can be of different types thus leading to a hypergraph [Berge & Shi 1992].

Based on the m-partite graph (or hypergraph) representation of the rule-structuring problem different models and algorithms can be developed. The rule-structuring problem could be loosely formulated as follows:

Minimize the total distance between any two rules belonging to different nodes of the m-partite graph subject to constraints, for example, limiting the number of rules selected from one node, limiting the number of clusters.

14.6 Decision Tables, Maps, and Atlases

A decision table is a collection of knowledge needed to make decision in a particular domain. It generalizes the structured matrix introduced in fig. 14.4A by:

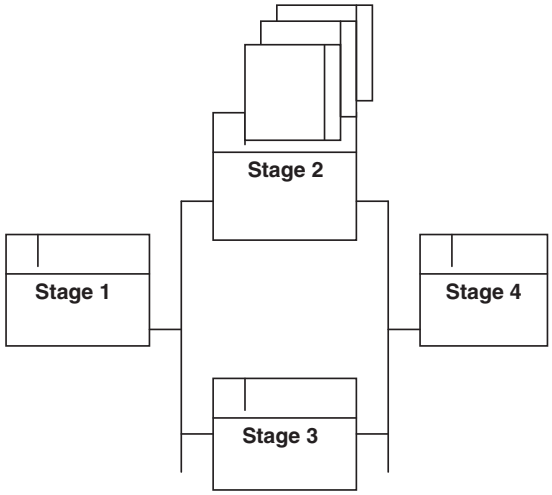

Fig. 14.6. Simple decision map

- Including decision rules generated by different learning algorithms;
- Transforming rules that may combine different decision rules, and so on;
- Content organization determined by the rule-structuring algorithm.

The actual decision can be made using one or more decision tables at any stage of the decision map (see fig. 14.6). Note that some decision tables may reduce to functions, neural network models, etc. The collection of decision tables distributed over all decision stages constitutes a decision map. Maps in turn can be combined in an atlas and multiple atlases make up a library, etc. There are two primary reasons for alternative decision tables. One is that a decision may follow one or more paths contained in one of those tables. The notion of independence has a profound impact on decision accuracy and user's confidence in the algorithmically generated result.

14.7 Summary

The decomposition has been discussed in the data mining literature, however, largely in the context of distributed learning. A typology of decomposition approaches applicable to data mining has been presented. One of the decomposition approaches, the structured rule-feature matrix, is used as the backbone of a system for informed decision making. Such a system can be implemented as a decision table, a decision map, or a decision atlas.

14.8 Review Questions

1. What is the typology of decomposition in data mining?
2. With an example explain a typical hybrid process and rule-structuring concept.
3. Form a simple decision map with its stages of operation.