Logical Analysis of Data

Endre Boros
RUTCOR, Rutgers University, USA

Peter L. Hammer
RUTCOR, Rutgers University, USA

Toshihide Ibaraki
Kwansei Gakuin University, Japan

INTRODUCTION

The logical analysis of data (LAD) is a methodology aimed at extracting or discovering knowledge from data in logical form. The first paper in this area was published as Crama, Hammer, & Ibaraki (1988) and precedes most of the data mining papers appearing in the 1990s. Its primary target is a set of binary data belonging to two classes for which a Boolean function that classifies the data into two classes is built. In other words, the extracted knowledge is embodied as a Boolean function, which then will be used to classify unknown data. As Boolean functions that classify the given data into two classes are not unique, there are various methodologies investigated in LAD to obtain compact and meaningful functions. As will be mentioned later, numerical and categorical data also can be handled, and more than two classes can be represented by combining more than one Boolean function.

Many of the concepts used in LAD have much in common with those studied in other areas, such as data mining, learning theory, pattern recognition, possibility theory, switching theory, and so forth, where different terminologies are used in each area. A more detailed description of LAD can be found in some of the references listed in the following and, in particular, in the forthcoming book (Crama & Hammer, 2004).

BACKGROUND

Let us consider a binary dataset $(T, F$, where $T \subseteq \{0,1\}^n$ (resp., $F \subseteq \{0,1\}^n$) is the set of positive (resp., negative) data (i.e., those belonging to positive (resp., negative) class). The pair (T, F) is called a *partially defined Boolean function* (or, in short, a *pdBf*), which is the principal mathematical notion behind the theory of LAD. A Boolean function f: $\{0,1\}^n \rightarrow \{0,1\}$ is called an extension of the pdBf (T, F) if $f(x)=1$ for all vectors x in T, and $f(y)=0$ for all vectors y in F. The construction of extensions that carry the essential information of the given dataset is the main theme of LAD. In principle, any Boolean function that agrees with the given data is a potential extension and is considered in LAD. However, as in general learning theory, simplicity and generalization power of the chosen extensions are the main objectives. To achieve these goals, LAD breaks up the problem of finding a most promising extension into a series of optimization problems, each with their own objectives, first finding a smallest subset of the variables needed to distinguish the vectors in T from those in F (finding a so called *support set*), next finding all the monomials which have the highest agreement with the given data (finding the strongest *patterns*), finally finding a best combination of the generated patterns (finding a *theory*). In what follows, we shall explain briefly each of these steps. More details about the theory of partially defined Boolean functions can be found in Boros, et al. (1998, 1999).

An example of a binary dataset (or pdBf) is shown in Table 1; Table 2 gives three extensions of it, among many others. Extension f_1 may be considered as a most compact one, as it contains only two variables, while extension f_3

Table 1. An example for pdBf (T, F)

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
	0	1	0	1	0	1	1	0
T	1	1	0	1	1	0	0	1
	0	1	1	0	1	0	0	1
	1	0	1	0	1	0	1	0
F	0	0	0	1	1	1	0	0
	1	1	0	1	0	1	0	1
	0	0	1	0	1	0	1	0

Table 2. Some extensions of the pdBf (T, F) given in Table 1

$$f_1 = x_5 x_8 \vee \overline{x}_5 \overline{x}_8$$
$$f_2 = \overline{x}_1 x_5 \vee x_3 \overline{x}_7 \vee x_1 x_5 \overline{x}_7$$
$$f_3 = x_5 x_8 \vee x_6 x_7$$

also may be interesting, as it reveals a monotone dependence on the involved variables.

MAIN THRUST

In many applications, the available data is not binary, which necessitates the generation of relevant binary features, for which the three-staged LAD analysis then can be applied. Following are some details about all the main stages of LAD, including the generation of binary features, finding support sets, generating patterns, and constructing theories.

Binarization of Numerical and Categorical Data

Many of the existing datasets contain numerical and/or categorical variables. Such data also can be handled by LAD after converting such data into binary data (i.e., binarization). A typical method for this is to introduce a cut point α_i for a numerical variable x_i; if $x_i \leq \alpha_i$ holds; then, it is converted to 1, and 0 otherwise. It is possible to use more than one cut point for a variable to create one or more number of regions of 1. Similar ideas also can be used for a categorical variable, defining a region converted into 1 by a subset of values of the variable. Binarization aims at finding the cut points for numerical attributes/finding the subsets of values for categorical attributes, such that the given positive and negative observations can be distinguished with the resulting binary variables, and their number is as small as possible. Mathematical properties of and algorithms for binarization are studied in Boros, et al. (1997). Efficient methods for the binarization of categorical data are proposed in Boros and Meñkov (2004).

Support Sets

It is desirable from the viewpoint of simplicity to build an extension by using as small a set of variables as possible. A subset S of the n original variables is called a support set, if the projections of T and F on S still has an extension. The pdBf in Table 1 has the following minimal support sets: $S_1=\{5,8\}, S_2=\{6,7\}, S_3=\{1,2,5\}, S_4=\{1,2,6\}, S_5=\{2,5,7\}, S_6=\{2,6,8\}, S_7=\{1,3,5,7\}, S_8=\{1,4,5,7\}$. For example, f_1 in Table 2 is constructed from S_1. Several methods to find small support sets are discussed and compared in Boros, et al. (2003).

Pattern Generation

As a basic tool to construct an extension f, a conjunction of a set of literals is called a pattern of (T, F), if there is at least one vector in T satisfying this conjunction, but no vector in F satisfies it, where a literal is either a variable or its complement. In the example of Tables 1 and 2, x_5x_8, $\bar{x}_5\bar{x}_8$, $\bar{x}_1x_5$,... are patterns. The notion of a pattern is closely related to the association rule, which is commonly used in data mining. Each pattern captures a certain characteristic of (T, F) and forms a part of knowledge about the data set. Several types of patterns (prime, spanned, strong) have been analyzed in the literature (Alexe et al., 2002), and efficient algorithms for enumerating large sets of patterns have been described (Alexe & Hammer, 2004; Alexe & Hammer, 2004; Eckstein et al., 2004).

Theories of a pdBf

A set of patterns that together cover T (i.e., for any $x \in T$, there is at least one pattern that x satisfies) is called a theory of (T, F). Note that a theory defines an extension of (T, F), since disjunction of the patterns yields such a Boolean function. The Boolean functions f_i in Table 2 correspond to theories constructed for the dataset of Table 1. Exchanging the roles of T and F, we can define co-patterns and co-theories, which play a similar role in LAD. Efforts in LAD have been directed to the construction of compact patterns and theories from (T, F), as such theories are expected to have better performance to classify unknown data. An implementation in this direction and its results can be found in Boros, et al. (2000). The tradeoff between comprehensive and comprehensible theories is studied in Alexe, et al. (2002).

FUTURE TRENDS

Extensions by Special Types of Boolean Functions

In many cases, it is known in advance that the given dataset has certain properties, such as monotone dependence on variables. To utilize such information, extensions by Boolean functions with the corresponding properties are important. For special classes of Boolean functions, such as monotone (or positive), Horn, k-DNF, decomposable, and threshold, the algorithms and complexity of finding extensions were investigated (Boros et al., 1995; Boros et al., 1998). If there is no extension in the specified class of Boolean functions, we may still want to find an extension in the class with the minimum number of errors. Such an extension is called the best-fit extension and studied in Boros, et al. (1998).