The Lsquare System for Mining Logic Data

Giovanni Felici
Istituto di Analisi dei Sistemi ed Informatica (IASI-CNR), Italy

Klaus Truemper
University of Texas at Dallas, USA

INTRODUCTION

The method described in this chapter is designed for data mining and learning on *logic data*. This type of data is composed of records that can be described by the presence or absence of a finite number of properties. Formally, such records can be described by variables that may assume only the values *true* or *false*, usually referred to as *logic* (or *Boolean*) *variables*. In real applications, it may also happen that the presence or absence of some property cannot be verified for some record; in such a case we consider that variable to be *unknown* (the capability to treat formally data with missing values is a feature of logic-based methods). For example, to describe patient records in medical diagnosis applications, one may use the logic variables *healthy*, *old*, *has_high_temperature*, among many others. A very common data mining task is to find, based on *training data*, the rules that separate two subsets of the available records, or explains the belonging of the data to one subset or the other. For example, one may desire to find a rule that, based one the many variables observed in patient records, is able to distinguish healthy patients from sick ones. Such a rule, if sufficiently precise, may then be used to classify new data and/or to gain information from the available data. This task is often referred to as machine learning or pattern recognition and accounts for a significant portion of the research conducted in the data mining community. When the data considered is in logic form or can be transformed into it by some reasonable process, it is of great interest to determine explanatory rules in the form of the combination of logic variables, or *logic formulas*. In the example above, a rule derived from data could be:

if (*has_high_temperature* is *true*) and (*running_nose* is *true*) then (*the patient is not healthy*).

Clearly such rules convey a lot of information and can be easily understood and interpreted by domain experts. Despite the apparent simplicity of this setting, the problem of determining, if possible, a logic formula that holds true for all the records in one set while it is false for all records in another set, can become extremely difficult when the dimension involved is not trivial, and many different techniques and approaches have been proposed in the literature. In this article we describe one of them, the Lsquare System, developed in collaboration between IASI-CNR and UTD and described in detail in Felici & Truemper (2002), Felici, Sun, & Truemper (2004), and Truemper (2004). The system is freely distributed for research and study purposes at *www.leibnizsystem.com.*

Data mining in logic domains is becoming a very interesting topic for both research and applications. The motivations for the study of such models are frequently found in real life situations where one wants to extract usable information from data expressed in logic form. Besides medical applications, these types of problems often arise in marketing, production, banking, finance, and credit rating. A quick scan of the updated Irvine Repository (see Murphy & Aha, 1994) is sufficient to show the relevance of logic-based models in data mining and learning application. The literature describes several methods that address learning in logic domains, for example the very popular decision trees (Breiman et al., 1984), the highly combinatorial approach of Boros et al. (1996), the interior point method of Kamath et al. (1992), or the partial enumeration scheme proposed by Triantaphyllou & Soyster (1996). While the problem formulation adopted by Lsquare is somewhat related the work in Kamath et al. (1992) and Triantaphyllou et al. (1994), substantial differences are found in the solution method adopted. Most of the methods considered in this area are of intrinsic deterministic nature, being based on the formal description of a problem in mathematical form and in its solution by a specific algorithm. Nevertheless, some real life situations present uncertainty and errors in the data that are often successfully dealt with by the use of fuzzy set and fuzzy membership theory. In such cases the proposed system may embed the uncertainty and the fuzziness of the data in a pre-processing step, providing fuzzy functions that determine the value of the Boolean variables.

Copyright © 2006, Idea Group Inc., distributing in print or electronic forms without written permission of IGI is prohibited.

BACKGROUND

This section provides the needed definitions and concepts.

Propositional Logic, SAT and MINSAT

A *Boolean variable* may take on the values *true* or *false*. One or more Boolean variables can be assembled in a *Boolean formula* using the operators $\neg$, $\wedge$, and $\vee$. A Boolean formula where Boolean variables, possibly negated, are joined by the operator $\wedge$ (resp. $\vee$) is called a *conjunction* (resp. *disjunction*). A *conjunctive normal form system* (CNF) is a Boolean formula where some disjunctions are joined with the operator $\wedge$. A *disjunctive normal form system* (DNF) is a Boolean formula where some conjunctions are joined with the operator $\vee$. Each disjunction (resp. conjunction) of a CNF (resp. DNF) system is called a *clause*. A Boolean formula is *satisfiable* if there exists an assignment of *true/false* values to its Boolean variables so that the value of the formula is *true*. The problem of deciding whether a CNF formula is satisfiable is known as the *satisfiability problem* (SAT). In the affirmative case, one also must find an assignment of *true/false* values for the Boolean variables of the CNF that make the formula *true*. A variation of SAT is the *minimum cost satisfiability problem* (MINSAT), where rational costs are associated with the *true* value of each logic variable and the solution to be determined has minimum sum of costs for the variables with value *true*.

Logic Data and Logic Separation

We consider {0, +/-1} vectors of given length *n,* each of which has an associated outcome with value *true* or *false*. We call these vectors *records* of logic data and view them as an encoding of logic information. A 1 in a record means that a certain Boolean variable has value *true*, and a –1 that the variable has value *false*. The value 0 is used for *unknown*. The outcome is considered to be the value of a Boolean variable *t* that we want to explain or predict. We collect the records for which the property *t* is present in a set A, and those for which *t* is not present in set B. For ease of recognition, we usually denote a member of A by *a*, and of B by *b*. The Lsquare system deduces {0, +/-1} separating vectors that effectively represent logic formulas and may be used to compute for each record the associated outcome, that is, to separate the records in A from the records in B. A separating set is a collection of separating vectors. The separation of A and B makes sense only when both A and B are non-empty, and when each record of A or B contains at least one {+/-1} entry. Consider two records of logic data; say *g* and *f*. We say that *f is nested in g* if for any entry f_i of *f* equal to +1 or –1, the corresponding entry g_i of *g* satisfies $g_i = f_i$. It is easy to show that if A and B are sets of {0, +/-1} records of the same length, a separating set S exists if and only if no record $a \in$ A is nested in any record b $\in$ B. A clear characterization of separating vectors can thus be given: a {0, +/-1} vector *s* separates a record $a \in$ A from B if:

s is not nested in any $b \in$ B (1)

s is nested in *a* (2)

Accordingly, we say that a separating set S separates B from A if, for each $a \in$ A, there exists a separating vector $s \in$ S that separates *a* from B.

MAIN THRUST

The problem of finding a separating set for A and B is decomposed into a sequence of subproblems, each of which identifies a vector *s* that separates a nonempty subset of A from B solving two minimization problems. To formulate these problems we introduce Boolean variables linked with the elements s_i of the vector *s* to be found. More precisely, we introduce Boolean variables p_i and q_i and state that $s_i = 1$ if p_i = True and q_i = False, $s_i = -1$ if p_i = False and q_i = True, and $s_i = 0$ if $p_i = q_i$ = False. The case $p_i = q_i$ = True is ruled out by enforcing the following conditions:

$$\neg p_i \vee \neg q_i \text{ , i } = 1, 2, ..., \text{n} \tag{3}$$

We can express the separation conditions (1) and (2) with the Boolean variables p_i and q_i. For (1) we have that *s* must not be nested in any $b \in$ B. Defining b^+ as the set of indices *i* for which b_i of b is equal to 1, that is, $b^+ = \{i | b_i = 1\}$ and $b^- = \{i | b_i = -1\}$, $b^0 = \{i | b_i = 0\}$, we summarize condition (1) writing that

$$(\vee_{i \in (b+ \cup b0)} q_i) \vee (\vee_{i \in (b- \cup b0)} p_i); \; \forall \; b \in \text{B} \tag{4}$$

For condition (2) we have to enforce that, if *s* separates *a* from B, then *s* is nested in *a*. In order to do so we introduce a new Boolean variable d_a that determines whether *s* must separate *a* from B. That is, d_a = *true* means that *s* need not separate *a* from B, while d_a = *false* requires that separation. For $a \in$ A, the separation condition is:

$$\neg q_i \vee d_a; \forall \text{ i } \in (a^+ \cup a^0)$$
$$\neg p_i \vee d_a; \forall \text{ i } \in (a^+ \cup a^0) \tag{5}$$

3 more pages are available in the full version of this document, which may be purchased using the "Add to Cart" button on the publisher's webpage: www.igi-global.com/chapter/lsquare-system-mining-logic-data/10686

Related Content

Identifying Single Clusters in Large Data Sets
Frank Klawonnand Olga Georgieva (2005). *Encyclopedia of Data Warehousing and Mining (pp. 582-585).*
www.irma-international.org/chapter/identifying-single-clusters-large-data/10664

Event/Stream Processing for Advanced Applications
Qingchun Jiang, Raman Adaikkalavanand Sharma Chakravarthy (2010). *Intelligent Techniques for Warehousing and Mining Sensor Network Data (pp. 305-325).*
www.irma-international.org/chapter/event-stream-processing-advanced-applications/39551

DEA Evaluation of Performance of E-Business Initiatives
Yao Chen, Luvai Motiwallaand M. Riaz Khan (2005). *Encyclopedia of Data Warehousing and Mining (pp. 349-352).*
www.irma-international.org/chapter/dea-evaluation-performance-business-initiatives/10621

Mining E-Mail Data
Steffen Bickeland Tobias Scheffer (2005). *Encyclopedia of Data Warehousing and Mining (pp. 768-772).*
www.irma-international.org/chapter/mining-mail-data/10700

A Machine Learning Approach to Data Cleaning in Databases and Data Warehouses
Hamid Haidarian Shahri (2009). *Progressive Methods in Data Warehousing and Business Intelligence: Concepts and Competitive Analytics (pp. 43-58).*
www.irma-international.org/chapter/machine-learning-approach-data-cleaning/28161