On the complexity of mining association rules

Extended Abstract

Fabrizio Angiulli[1], Giovambattista Ianni[2], and Luigi Palopoli[3]

[1] ISI-CNR c/o Universitá della Calabria, DEIS, Via P. Bucci 41C, Rende, Italy
angiulli@isi.cs.cnr.it
[2] Universitá della Calabria, DEIS, Via P. Bucci 41C, Rende, Italy
ianni@deis.unical.it
[3] Universitá di Reggio Calabria, DIMET, Loc. Feo di Vito, Reggio Calabria, Italy
palopoli@ing.unirc.it

Abstract. In this paper we describe our ongoing research towards establishing the complexity of mining association rules from relational databases. We consider both quantitative, categorical and boolean association rules and various forms of quality indexes, including confidence, support, gain, laplace. The presented results show that all these problems are, generally, computationally hard to solve, even if we are able to single out some interesting tractable special cases.

1 Introduction

The enormous growth of information available in database systems has pushed a significant development of techniques for knowledge discovery in database. At the heart of the knowledge discovery process there is the application of data mining algorithms that are in charge of extracting hidden relationships holding among pieces of information stored in a given database [6]. Most used data mining algorithms include classification techniques, clustering analysis and association rule induction [2]. In this paper, we focus on this latter data mining technique. Informally speaking, an association rule tells that a conjunction of conditions implies a consequence. For instance, the rule $hamburger, fries \Rightarrow soft\text{-}drink$ induced from a purchase database, tells that a customer purchasing hamburgers and fries also purchases a soft-drink.

An association rule induced from a database is interesting if it describes a relationship that is, in a sense, "valid" as far as the information stored in the database is concerned. To state such validity, *indexes* are used, i.e. functions which usually take values in $[0, 1]$, that tell to what extent an extracted association rule describe knowledge valid in the database at hand. For instance, a *confidence* value of 0.7 associated to the rule above tells that 70 percent of purchases including hamburgers and fries also include a soft-drink. In the literature, several index definitions have been provided (see e.g. [4]), that encode several interestingness criteria. Clear enough, information patterns expressed in the form of association rules and associated indexes indeed denote knowledge that can be useful in several application contexts, e.g., market basket analysis.

In some application contexts, however, *Boolean* association rules, like the one above, are not expressive enough for the purposes of the given knowledge discovery task, since, in such rules, conditions are simply attributes that can evaluate either to true or false. In order to obtain more expressive association rules, one can allow more general forms of condition to occur therein. *Quantitative association rules* [13] are ones where both the premise and the consequent use conditions

of one of the following forms: (i) $A = u$; (ii) $A \neq u$; (iii) $A' \in [l', u']$; (iv) $A' \notin [l', u']$, where A is a *categorical* attribute, i.e., an attribute that has associated a discrete, unordered domain and u is a value in this domain, and A' is a *numeric* attribute, that is, one associated with an ordered domain of numbers, and l' and u' ($l' \leq u'$) are two, not necessarily distinct, values. For instance, the quantitative rule

$$(hamburger \in [2, 4]), (ice\text{-}cream\text{-}taste = chocolate) \Rightarrow (soft\text{-}drink \in [1, 3])$$

induced from a purchase database, tells that a customer purchasing from 2 to 4 hamburgers and a chocolate ice-cream also purchases from 1 to 3 soft-drinks.

In either of their forms, inducing association rules is a quite widely used data mining technique, several systems have been developed based on them [3, 10], and several successful applications in various contexts have been described [5].

Despite the wide-spread utilization of association rule induction in practical applications, a thorough analysis of the complexity of the associated computational tasks have not been developed. Some contribution about computational complexity analysis pertaining association rules are given in [11, 12, 15] and [8]. In [11] and [12], a NP-hardness result is stated regarding the mining of association rules (or, in general, of *conditions*) having an optimal *entropy* (resp. *chi-square*); in [15], under some assumptions, the NP-completeness of mining quantitative association rules with a *confidence* and a *support*[1] greater than two given thresholds is proved along with a result stating a polynomial bound on the complexity of mining quantitative rules over databases where the number of possible items is constant. In [8], it is stated the $\#P$-hardness of counting the number of mined association rules (under support measure), and a version of our Theorem 1, for boolean association rules, is given. Furthermore, [16] gives some result about the computational complexity of mining frequent itemsets under combined constraints on the number of items and on the frequency threshold.

In this paper, we explain our contribution to the understanding of the computational complexity implied by inducing association rules using some of the mostly used rule quality indexes, namely, confidence, support, θ-gain and h-laplace.

There are effective real world applications which require the presence of a special value having a particular behavior, usually denoting the absence of information, called the *null* value (in the following denoted by ϵ).
As an example, consider a market database: null values can be used to denote the absence of a product in a particular transaction. This is quite different than specifying the value 0 instead. As a further example, consider unavailable values in medical records representing clinical cases in analysis of patient data.
We call a database allowing null values, a *database with nulls*. When we learn association rules from databases with nulls, we require that conditions on attributes assuming the null value are always unsatisfied, i.e. that it is not possible to specify conditions on null values.

A boolean association rule can be regarded as a special case of quantitative or categorical association rule mined on a database with nulls.

We shall show that, when usual databases are considered, the rule mining problem belong to different classes, depending from the chosen index. When databases with nulls are considered, independently of the chosen reference index, the rule induction task is NP-complete.

[1] Entropy, confidence and support are indexes (see below).

The plan of the paper is as follows. In the following section we give preliminary definitions. In Section 3 we state general complexity results about inducing association rules. Finally, further complexity results regarding several interesting special cases of mining boolean association rules are described in Section 4.

2 Preliminaries

We begin by defining several concepts that will be used throughout the paper, including, among others, those of association rule induction problems and indexes.

An *attribute* is an identifier with an associate domain. A *categorical attribute* (resp., *numeric attribute*) is one whose domain is an unordered set of values (resp., a set of integer or rational numbers). Both categorical and numeric attributes include in their domain the special value ϵ.

Let I be a set of attributes. A *database* T on I is a relation with duplicates having schema I. Let $A \in I$ and let t be a tuple of T. We denote by $t[A]$ the value of the attribute A in the tuple t. The *size* $|t|$ of $t \in T$ is $|\{A \in I \mid t[A] \neq \epsilon\}|$. We denote by $\mathbf{dom}(A, T)$ the set $\{t[A] \mid t \in T\} - \{\epsilon\}$.

Let I be a set of attributes, and let T be a database on I. We say that T is a *database without nulls* if, for each $t \in T$, $|t| = |I|$. Otherwise we say that T is a *database with nulls*. We say that T is a *boolean database* if it is a database with nulls and, for each $A \in I$, $\mathbf{dom}(A, T) = \{c_A\}$, where c_A is an arbitrary constant. We say that T is a *sparse database* if it is a database with nulls and, for each $t \in T$, $|t| = \mathcal{O}(\log |I|)$.

An *atomic condition* on A is an expression of the form $A = u$ or $A \neq u$, where A is a categorical attribute and u is a value in the domain of A distinct from the ϵ value, or an expression of the form $A \in [l, u]$ or $A \notin [l, u]$, where A is a numeric attribute and l and u ($l \leq u$) are two, not necessarily distinct, numeric values. When $l = u$, the notation $A = u$ ($A \neq u$ resp.) is equivalent to $A \in [l, u]$ ($A \notin [l, u]$ resp.). We denote by $\mathbf{dom}(A = u, T)$ ($\mathbf{dom}(A \neq u, T)$ resp.) the set $\mathbf{dom}(A, T) \cap \{u\}$ ($\mathbf{dom}(A, T) - \{u\}$ resp.), and by $\mathbf{dom}(A \in [l, u], T)$ ($\mathbf{dom}(A \notin [l, u], T)$ resp.) the set $\mathbf{dom}(A, T) \cap [l, u]$ ($\mathbf{dom}(A, T) - [l, u]$ resp.).

A *condition* C on a set of distinct attributes $A_1, \ldots, A_n$ is an expression of the form $C = C_1 \wedge \ldots \wedge C_n$, where each C_i is an atomic condition on A_i, for each $i = 1, \ldots, n$. We denote by $\mathbf{att}(C)$ the set $A_1, \ldots, A_n$. The *size* $|C|$ of C is n.

We are now in the condition of defining association rules and their semantics. Let I be a set of attributes. An *association rule* on I is an expression of the form $B \Rightarrow H$, where B and H, called *body* and *head* of the rule resp., are two conditions on the sets of attributes I_B and I_H resp., such that $\emptyset \subset I_B, I_H \subset I$, and $I_B \cap I_H = \emptyset$. The *size* $|B \Rightarrow H|$ of the rule is $|B| + |H|$.

Let I be a set of attributes, let T be a database on I, and let t be a tuple of T. Let $A \in I$, and let C_a be an atomic condition on A, we say that t *satisfies* C_a, written $t \vdash C_a$, iff $t[A] \in \mathbf{dom}(C_a, T)$. Let $C = C_1 \wedge \ldots \wedge C_n$ be a condition, we say that t *satisfies* C, written $t \vdash C$, iff $t \vdash C_i$, for $i = 1, \ldots, n$. Otherwise we say that t *does not satisfy* C, written $t \nvdash C$. By T_C we denote the set of tuples $\{t \in T \mid t \vdash C\}$.

Let I be a set of attributes, and let T be a database on I. Let A be an attribute, and let C_a be an atomic condition on A. We say that C_a is *trivial* if $T_{C_a} = T$. Let C be a condition. We say that C is *redundant* if it contains at least a trivial atomic condition. Let $B \Rightarrow H$ be an association rule on I. We say that $B \Rightarrow H$ is *redundant* if $B \wedge H$ is redundant. Since redundant rules with

index value of interest can be always easily built, in the following, we will focus our attention on non redundant association rules.

When inducing association rules from databases in data mining applications, one is usually interested in obtaining rules that describe knowledge "largely" valid in the given database. This concept is captured by several notions of *indexes*, which have been defined in the literature. In the following, we shall consider the most widely used of them, whose definitions are given next.

Let I be a set of attributes, let T be a database on I, and let $B \Rightarrow H$ be an association rule on I. Then: (i) the *support* of $B \Rightarrow H$ in T, written $sup(B \Rightarrow H, T)$, is $\frac{|T_{B \wedge H}|}{|T|}$; (ii) the *confidence* of $B \Rightarrow H$ in T, written $cnf(B \Rightarrow H, T)$, is $\frac{|T_{B \wedge H}|}{|T_B|}$; (iii) if θ is a rational number, $0 < \theta \leq 1$, then the *θ-gain* of $B \Rightarrow H$ in T, written $gain_\theta(B \Rightarrow H, T)$, is $\frac{|T_{B \wedge H}| - \theta \cdot |T_B|}{|T|}$; (iv) if h is an integer, $h \geq 2$, then the *h-laplace* of $B \Rightarrow H$ in T, written $laplace_h(B \Rightarrow H, T)$, is $\frac{|T_{B \wedge H}|+1}{|T_B|+h}$. We define $\mathbf{I}$ as the set $\{sup, cnf, gain_\theta, laplace_h\}$.

Now that we have defined association rules and associated indexes (that, in different forms, measure the validity of an association rule w.r.t. a database where it has been induced from), we are in the condition to formally define next the association rule induction problems. Let I be a set of attributes, let T be a database on I, let k, $1 \leq k \leq |I|$, be a natural number, and let s, $0 < s \leq 1$, be a rational number. Furthermore, let $\rho \in \{sup, cnf, laplace_h, gain_\theta\}$.
The association rule induction problem $\langle I, T, \rho, k, s \rangle$ is as follows: *Is there a non redundant association rule R such that $|R| \geq k$ and $\rho(R, T) \geq s$?*

In the literature it is usually assumed that, in answering an association rule induction problem, one looks for rules which match some bounds in terms of two or more indexes [4]. Here we preferred to split the problem as to refer to one index at a time. Indeed, this allows us to single out more precisely complexity sources, and, moreover, complexity measures for problems involving more than one index can be obtained fairly easily from problems involving only one index.

In the following, we shall assume that the reader is familiar with basic concepts regarding computational complexity and, in particular, the complexity classes P , NP and L . Some further concepts are recalled next.

MAJORITY gates are unbounded fan-in gates (with binary input and output) that output 1 if and only if more than half of their inputs are non-zero.

A family $\{C_i\}$ of boolean circuits, s.t. C_i accepts strings of size i, is uniform if there exists a Turing machine $\mathcal{T}$ which on input i produces the circuit C_i. $\{C_i\}$ is said to be *logspace uniform* if $\mathcal{T}$ carries out its work using $O(\log i)$ space. Define AC^0 (resp. TC^0) as the class of decision problems solved by uniform families of circuits of polynomial size and constant depth, with AND, OR, and NOT (resp. MAJORITY) gates of unbounded fan-in [1].

3 General complexity results

Here we investigate the complexity of evaluating $\langle I, T, \rho, k, s \rangle$ when I, T, k and s are all taken as input values.

As we are interested in non redundant conditions C such that $|T_C| > 0$, in the rest of the paper we can restrict our attention to conditions and association rules including only values from the database T of interest.

Theorem 1. *The problem $\langle I, T, sup, k, s \rangle$ is NP-complete.*

Proof. (*Sketch*) The proof is by reduction of the problem CLIQUE, which is well-known to be NP-complete [7]. Let $G = (V, E)$ be an undirected graph, with set of nodes $V = \{v_1, \ldots, v_n\}$ and set of edges $E = \{e_1 = \{v_{p_1}, v_{q_1}\}, \ldots, e_m = \{v_{p_m}, v_{q_m}\}\}$. Let k be an integer. The CLIQUE problem is: *Does there exist in G a complete subgraph (clique) of size at least k ?*

W.l.o.g. suppose the graph G is connected. We build an instance of $\langle I^{clq}, T^{clq}, sup, k, s\rangle$ as follows: let I^{clq} be the set consisting of the numeric or categorical attributes $I_1, \ldots, I_n$, so that I_j represents the node v_j of G, for each $j = 1, \ldots, n$. Let T^{clq} be the database on I^{clq} formed by the tuples t_{e_i}, for each $i = 1, \ldots, m$, such that $t_{e_i}[I_j] = 0$ (or $t_{e_i}[I_j] = \epsilon$ if we consider a database with nulls) if $v_j \in e_i$, and $t_{e_i}[I_j] = 1$ otherwise (t_{e_i} denotes the edge e_i of G). It can be proved that G has a clique of size k in G iff $\langle I^{clq}, T^{clq}, sup, n-k, \frac{k(k-1)}{2m}\rangle$ is a YES instance.

Theorem 2. *Given a database T without nulls, the problem $\langle I, T, cnf, k, s\rangle$ is in P.*

Proof. (*Sketch*) The proof is based on the following result.

Lemma 1. *Let T a database without nulls. Then there exists a non redundant association rule $B \Rightarrow H$ on I such that $cnf(B \Rightarrow H, T) \geq s$ iff there exist an attribute $J_H \in I$, a value $u_H \in \mathbf{dom}(J_H, T)$, and a tuple $t \in T$, such that the rule $\left(\bigwedge_{J \in (I - \{J_H\})} (J = t[J])\right) \Rightarrow (J_H \neq u_H)$ is non redundant and has confidence greater than or equal than s.*

Hence, the problem can be solved in time $\mathcal{O}(|I| \cdot |T|^2 \log |T|)$ by testing if there exists an association rule, of the form above described, with confidence exceeding the threshold s. Figure 1 reports the algorithm deciding the problem $\langle I, T, cnf, k, s\rangle$ on databases without nulls.

For each $i = 1, \ldots, |I|$, consider the ith attribute J_i of I
  Build the ordered database T^i sorting T w.r.t. the sequence $J_1, \ldots, J_{i-1}, J_{i+1}, \ldots, J_n, J_i$
    For each block B of adjacent tuples of T^i that are identical on the attributes $I - \{J_i\}$
      Determine the value $b = \min_{u \in \mathbf{dom}(J_i, T)} |\{t \in B \mid t[J_i] = u\}|$
      If $(|B| - b)/|B| \geq s$ then return "yes" and exit
Return "no"

Fig. 1. The algorithm deciding the confidence problem on databases without nulls

Theorem 3. *Given a database T with nulls, the complexity of $\langle I, T, cnf, k, s\rangle$ is NP-complete.*

Proof. (*Sketch*) The proof, as in Theorem 1, is by reduction of CLIQUE. Let $G = (V, E)$ be an undirected graph, with set of nodes $V = \{v_1, \ldots, v_n\}$ and set of edges $E = \{e_1 = \{v_{p_1}, v_{q_1}\}, \ldots, e_m = \{v_{p_m}, v_{q_m}\}\}$. We build an instance $\langle I^{clq}, T^{clq}, cnf, k, s\rangle$ as follows. Let I^{clq} be $I' \cup \{I_{n+1}\}$, where $I' = \{I_1, \ldots, I_n\}$ is a set of numeric or categorical attributes, I_j represents the node v_j of G, for $j = 1, \ldots, n$, and I_{n+1} is a new attribute representing a new node v_{n+1}. Let $T^{clq} = T' \cup T''$, where T' includes the tuples t_{e_i} and t'_{e_i}, where $t_{e_i}[I_j] = \epsilon$ ($t'_{e_i}[I_j] = \epsilon$ resp.) if $v_j \in e_i$, and 1 otherwise, for $i = 1, \ldots, m$, $j = 1, \ldots, n+1$, (the tuples t_{e_i} and t'_{e_i} both denote the edge e_i of G), and T'' includes the tuples t_{v_i}, where $t_{v_i}[I_j] = \epsilon$ if $i = j$, and 1 otherwise, for $i = 1, \ldots, n+1$, $j = 1, \ldots, n+1$. (See Figure 2 for an example of the reduction we are employing here). It can be proved that there exists a clique of size k in G iff $\langle I^{clq}, T^{clq}, n-k+1, \frac{k^2}{k^2+1}\rangle$ is a YES instance.

$$G = (\{v_1, \ldots, v_4\}, \{(v_1, v_2), (v_1, v_3), (v_2, v_3), (v_1, v_4), (v_2, v_4), (v_3, v_4)\})$$

	I_1	I_2	I_3	I_4	I_5
$t_{\{v_1,v_2\}}$	ϵ	ϵ	1	1	1
$t'_{\{v_1,v_2\}}$	ϵ	ϵ	1	1	1
$t_{\{v_1,v_3\}}$	ϵ	1	ϵ	1	1
$t'_{\{v_1,v_3\}}$	ϵ	1	ϵ	1	1
$t_{\{v_2,v_3\}}$	1	ϵ	ϵ	1	1
$t'_{\{v_2,v_3\}}$	1	ϵ	ϵ	1	1
$t_{\{v_1,v_4\}}$	ϵ	1	1	ϵ	1
$t'_{\{v_1,v_4\}}$	ϵ	1	1	ϵ	1
$t_{\{v_2,v_4\}}$	1	ϵ	1	ϵ	1
$t'_{\{v_2,v_4\}}$	1	ϵ	1	ϵ	1
$t_{\{v_3,v_4\}}$	1	1	ϵ	ϵ	1
$t'_{\{v_3,v_4\}}$	1	1	ϵ	ϵ	1
t_{v_1}	ϵ	1	1	1	1
t_{v_2}	1	ϵ	1	1	1
t_{v_3}	1	1	ϵ	1	1
t_{v_4}	1	1	1	ϵ	1
t_{v_5}	1	1	1	1	ϵ

Fig. 2. An example of the reduction used in Theorem 3

Theorem 4. *The complexity of $\langle I, T, \rho, k, s\rangle$, with $\rho \in \{gain_\theta, laplace_h\}$, is NP-complete.*

Proof. (*Sketch*) The proof follows a similar line of reasoning as that employed in the proof of Theorem 1.

4 Further complexity results

In this section, we investigate the computational complexity of several interesting special cases of mining boolean association rules.

Sparse databases are of interest from a practical point of view as in many real applications each tuple contains only a limited number of non null attributes. Think, as an example, to databases of transactions from a megastore. For databases showing this property, complexity figures are quite different from what we have proved above.

Theorem 5. *Given a sparse boolean database T, the problem $\langle I, T, \rho, k, s\rangle$, with $\rho \in \mathbf{I}$, is in L.*

Given a sparse boolean database T on a set of attributes I, the previous result can be proved taking advantage of the fact that each condition C satisfiable by T can be implicitly represented employing a pointer to a tuple t of T (having size $\log |T|$) and a binary counter of size $\mathcal{O}(\log |I|)$ (whose ith bit states the presence of the ith non null attribute of t in the condition C) together.

The following results assume some parameters (e.g., the lower bound on the rule length k, the index value threshold s) of the general association rule mining problem to be fixed. The relevance of the analysis we present below is two-fold. First, it eases the task of detecting actual complexity

sources. Second, from a practical point of view, users are often interested in solving such simplified tasks.

Theorem 6. *Given s a fixed constant in $(0, 1)$, and T a boolean database, the problem $\langle I, T, sup, k, s\rangle$ is NP-complete.*

Unfortunately, the above result states that, even if we wish to mine only rules with a support larger than a fixed threshold, as, for instance, 75 percent, the problem remains intractable. Note that the special case $\langle I, T, sup, k, 1\rangle$ can be easily shown to be in P.

The forthcoming Theorems (7,8 and 9) associate some task related to mining association rules to very low complexity classes such as TC^0 and AC^0. It turns out that these problems are highly parallelizable (recall that $AC^0 \subset TC^0 \subseteq NC^1$, [9]).

Theorem 7. *Given k a fixed constant, and T a boolean database, the complexity of $\langle I, T, sup, k, s\rangle$ is in TC^0.*

In other words, in presence of a constant lower bound on the rule length, the problem of finding an high supported itemset lies very low in the complexity hierarchy.

It is of interest to investigate the complexity of mining association rules when the value $s|T|$ is fixed. In this case $\langle I, T, sup, k, s\rangle$ corresponds to the problem of finding an association rule satisfied by almost a fixed number of tuples. Such a problem becomes of relevance when it is necessary to find a fixed set of transactions satisfying a certain property (e.g. in statistic sampling, see [14]).

Theorem 8. *Given $\lceil s|T|\rceil$ a fixed constant, and T a boolean database, the complexity of $\langle I, T, sup, k, s\rangle$ is in TC^0.*

The previous result can be proved by reducing the problem $\langle I, T, sup, k, s\rangle$ to the problem $\langle I', T', sup, \lceil s|T|\rceil, k/|I|\rangle$, where I' and T' are obtained from I and T "exchanging" attributes with tuples and vice versa (think to the database as a boolean matrix M and consider the transposed of M), and then using Theorem 7.

Theorem 9. *Given k and $\lceil s|T|\rceil$ two fixed constants, and T a boolean database, the complexity of $\langle I, T, sup, k, s\rangle$ is in AC^0_2.*

5 Conclusions

In this paper, we have resumed some computational complexity results regarding the problem of mining association rules from transaction databases. We have a generalized form of association rules embracing both the quantitative and boolean approach, and we used well-known quality indexes, namely, support, confidence, gain and laplace. After having formally defined association rule mining problems, we have shown that the general versions of these problems are NP-complete when usual databases are considered.

Then, we have focused on several interesting restricted cases, for most of which lower complexity bounds have been proved to hold. It is relevant to note that these cases are often related to complexity classes for which the existence of highly parallelizable algorithms has been proved. For example, for sparse databases, the complexities of the mining problem lies within L. In some

other analyzed cases, where some of the parameters of the mining problems are considered as fixed constants, the mining problem lies in TC^0 or in AC^0.

The complexity analysis presented in this paper is not complete, though. For instance, it is relevant to analyze the complexity induced by adopting other indexes as, for instance, *entropy* and *improvement* [11, 10].

References

1. M. Agrawal, E. Allender, and S. Datta. On TC^0, AC^0 and arithmetic circuits. *Proc. of the 12th Annual IEEE Conf. on Computational Complexity*, pages 134–148, 1997.
2. Rakesh Agrawal, Tomasz Imielinski, and Arun N. Swami. Mining association rules between sets of items in large databases. In *Proc. of the ACM SIGMOD Intl. Conf. on Management of Data, Washington, D.C., May 26-28, 1993*, pages 207–216.
3. Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In *VLDB'94, Proc. of 20th Intl. Conf. on Very Large Data Bases, September 12-15, 1994, Santiago de Chile, Chile*, pages 487–499.
4. Roberto J. Bayardo, Jr and Rakesh Agrawal. Mining the most interesting rules. *Proc. of the Fifth ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining, August 15-18, 1999, San Diego, CA, USA*, pages 145–154, 1999.
5. Tom Brijs, Gilbert Swinnen, Koen Vanhoof, and Geert Wets. Using association rules for product assortment decisions: A case study. *Proc. of the Fifth ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining.*
6. U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy. *Advances in Knowledge Discovery and Data Mining.* AAAI Press/MIT Press, 1996.
7. M.R. Garey and D.S. Johnson. *Computers and Intractability, A Guide to the Theory of NP-Completeness.* W.H. Freeman and Company, San Francisco, Ca, 1979.
8. Dimitrios Gunopulos, Heikki Mannila, and Sanjeev Saluja. Discovering all most specific sentences by randomized algorithms. In *Database Theory - ICDT '97, 6th Intl. Conf., Delphi, Greece, January 8-10, 1997, Proceedings.*
9. A. Hajnal, W. Maass, P. Pudlak, M. Szegedy, and G. Turan. Threshold circuits of bounded depth. *Proc. of the 28th Annual IEEE Symp. on Foundations of Computer Science*, pages 99–110, 1987.
10. Roberto J. Bayardo Jr., Rakesh Agrawal, and Dimitrios Gunopulos. Constraint-based rule mining in large, dense databases. In *Proc. of the 15th Intl. Conf. on Data Engineering, 23-26 March 1999, Sydney, Austrialia*, pages 188–197.
11. Shinichi Morishita. On classification and regression. *Discovery Science*, pages 40–57, 1998.
12. Shinichi Morishita and Jun Sese. Traversing itemset lattice with statistical metric pruning. In *Proc. of the Nineteenth ACM SIGMOD-SIGACT-SIGART Symp. on Principles of Database Systems, May 15-17, 2000, Dallas, Texas, USA.*
13. Ramakrishnan Srikant and Rakesh Agrawal. Mining quantitative association rules in large relational tables. In *Proc. of the 1996 ACM SIGMOD Intl. Conf. on Management of Data, Montreal, Quebec, Canada, June 4-6, 1996*, pages 1–12.
14. Hannu Toivonen. Sampling large databases for association rules. In *VLDB'96, Proc. of 22th Intl. Conf. on Very Large Data Bases, September 3-6, 1996, Mumbai (Bombay), India.*
15. Jef Wijsen and Robert Meersman. On the complexity of mining quantitative association rules. *Data Mining and Knowledge Discovery*, 2(3):263–281, 1998.
16. M. Zaki and M. Ogihara. Theoretical foundations of association rules. *In Proc. of 3rd SIGMOD'98 Workshop on Research Issues in Data Mining and Knowledge Discovery (DMKD'98), Seattle, Washington, USA*, 1998.