

Association Rule Mining Algorithms for Set-Valued Data

Christopher A. Shoemaker¹ and Carolina Ruiz¹

Department of Computer Science, Worcester Polytechnic Institute
Worcester, MA 10609 USA

ruiz@cs.wpi.edu <http://www.cs.wpi.edu/~ruiz>

Abstract. This paper presents an association rule mining system that is capable of handling set-valued attributes. Our previous research has exposed us to a variety of real-world biological datasets that contain attributes whose values are sets of elements, instead of just individual elements. However, very few data mining tools accept datasets that contain these *set-valued attributes*, and none of them allow the mining of association rules directly from this type of data. We introduce in this paper two algorithms for mining (classification) association rules directly from set-valued data and compare their performance. We have implemented a system based on one of these algorithms and have applied it to a number of biological datasets. We describe here our system and highlight its merits by means of comparing the results achieved with it and the failed attempts to mine association rules from those datasets using standard tools. Our system makes the creation of input files containing set-valued data much easier, and makes the mining of association rules directly from these data possible.

1 Introduction

Since the application of association rules to market basket analysis in [2], association rules have been the subject of active research in the data mining community, and the Apriori algorithm [1] has been the standard association rule mining algorithm. Apriori has been used to discover association rules from data in many different domains, but the tools that implement the Apriori algorithm always view the data in one of two forms: a list of transactions, where each transaction is a group of items, or a list of records, where each record has an individual elementary value for each attribute in the dataset. However, there are certain data types that do not easily fit into either of these two forms. Our research involves datasets from biological domains [13, 14] that provide examples of these data types, because they contain set-valued attributes. *Set-valued attributes* are attributes whose values are sets of elements.

Sets are natural representations for the values of many important attributes in various datasets. One example of such a *set-valued attribute*, or simply, *set-attribute*, is the set of alleles present at a particular genomic location for each person in a database of people. Another example is the set of leading actors who starred in each movie of a movie database [8]. See Figure 1. We use here this toy dataset as a running example in order to provide simple illustrations of some of the concepts involved in mining association rules from set-valued data without the added complexity of the biology terminology. The value of a set-attribute is a set. We call this value a *set-value*. For example, the

Table 1. An example dataset with set-valued attributes

Movie Title	Leading Actors	Year
Rocky	{Sylvester Stallone}	1976
Lethal Weapon	{Mel Gibson, Danny Glover}	1987
Lethal Weapon 3	{Mel Gibson, Danny Glover}	1992
Lethal Weapon 4	{Mel Gibson, Danny Glover, Chris Rock}	1998

“Leading Actors” set-attribute for “Lethal Weapon” has set-value {Mel Gibson, Danny Glover}. A set-attribute can accompany other *normal* (non-set) attributes in a dataset, e.g. the year in which the movie was released.

Several other researchers have recognized both the prevalence of set-valued data in real-world applications and the lack of data mining algorithms and tools that accommodate set-valued data. [12] describes work on instance-based learning with set-valued attributes. [4] addresses rule induction from decision trees, where the attributes of the instances are set-valued. [7] covers induction of decision trees in numeric domains where attributes are set-values. To the best of our knowledge, none of the association rule mining tools directly support set-valued attributes.

In this paper we describe our data mining system, which is able to mine (classification) association rules directly from set-valued data. For this purpose, we introduce the *Set Based Apriori* (SBA) algorithm. In order to define a control environment for evaluating the performance of our SBA algorithm, we also describe techniques for transforming set-valued attributes into normal attributes that the Apriori algorithm can handle. We show how we have combined one of those transformations with Apriori to give Apriori access to set-valued data. We call this combination *Transformation Based Apriori* (TBA). We perform a complexity analysis of both TBA and SBA. This analysis motivates our system design decisions.

2 Background

The classical definition of association rules is as follows [2]: Let I be a set of items. Each *transaction*, T , is also a set of items, called an *itemset*, and $T \subset I$. The entire dataset, D , consists of uniquely identifiable transactions. An association rule is a rule of the form $A \Rightarrow B$ between the itemsets A and B , where $A \cap B = \emptyset$. Various metrics describe the utility of an association rule. The most common ones are the percent of all transactions containing $A \cup B$, called the *support*, and the percent of transactions containing B among transactions containing A , called the *confidence* of the rule. A *classification association rule* [9] is a rule whose consequent consists of a single value from a pre-specified collection of (target) values.

The *Apriori algorithm* [1] to mine association rules takes the dataset as input, along with minimum support and minimum confidence thresholds, and its final output is a list of all association rules whose confidence and support are above the minimum thresholds. As an intermediate step, it produces frequent itemsets. An itemset is *frequent* if its support is greater than or equal to the user-specified minimum support. An itemset of size k is called a k -itemset.

3 Our Algorithm: Set Based Apriori (SBA)

The basic idea of our algorithm is to separate the search for frequent combinations of set elements *within* the set-attributes from the search for frequent combinations of set-values *between* the set-attributes. The underlying assumption is that the dependencies within the set-attributes, i.e. the frequent set-values, are more interesting than some of the dependencies between elements from different set-attributes and dependencies between set-values and normal values. SBA first mines frequent sets within each set-attribute and then examines the dependencies between frequent set-values, normal values, and frequent set-values from other set-attributes. One result of this ordering is that the final rules contain literals that refer directly to set-values instead of only elements. An example of the type of rule mined by our algorithm is: $\text{Leading Actors} = \{\text{Mel Gibson, Danny Glover}\} \Rightarrow \text{Year} \geq 1980$. Our algorithm does not, however, limit the expressiveness of the rules.

SBA: Phase I First, the SBA algorithm processes each set-attribute separately, producing set-values that will finally appear as literals in the association rules. This phase applies the Apriori principle. However, this phase differs from the Apriori algorithm by repeating the mining process for each set-attribute and by limiting the search to only the current set-attribute. Pseudocode for this phase is in Figure 1. While Apriori only counts an instance toward the support of an itemset if every item-value pair appears in the instance exactly as in the itemset, the SBA algorithm allows instances to contribute to the support of an itemset when every set-value in the itemset is exactly equal to *or a subset of* the corresponding set-value in the instance.

Input: D : all instances in a set-valued dataset; min_support : minimum support threshold
Output: F : a collection of all frequent set-values
Variables: C_k : set of candidate k -sets.
 F_k : set of frequent k -sets. C_i : set of candidate itemsets in instance i .
 i_A : set-value for set-attribute A in instance i . F_A : set of frequent set-values for attribute A .

```

for all set-attributes,  $A$ , in  $D$  {
     $F_1 = \{\text{frequent elements of attribute } A\}$ 
    for ( $k = 2; F_{k-1} \neq \emptyset; k++$ ) {
         $C_k = \text{GenerateCandidates}(F_{k-1})$ 
        for all instances,  $i, i \in D$  {
             $C_i = \{c \in C_k | c \subseteq i_A\}$ 
            for all candidate itemsets,  $c, c \in C_i$ 
                 $c.\text{count}++$ 
            }
             $F_k = \{c \in C_k | c.\text{count} \geq \text{min\_support}\}$ 
             $F_A = \cup_k F_k$ 
        }
    }
return  $F = \cup_A F_A$ 

```

Fig. 1. Phase I of the SBA algorithm

SBA: Phase II When Phase I of SBA is complete, our algorithm uses the resulting frequent itemsets as input to a modified Apriori algorithm. The important aspect here is that the itemsets from the first phase become atomic items in the second phase. Itemsets in Phase II include or exclude entire itemsets from the first phase as indivisible items. The pseudocode for Phase II appears in Figure 2. Association rules are produced from the resulting frequent itemsets in the standard fashion described in [1].

Input: D : a dataset of instances
 S : a collection of all the frequent set-values in D . $min_support$: minimum support threshold.
Output: F : a collection of all the frequent itemsets
Variables: C_k : collection of candidate k -itemsets
 F_k : collection of frequent k -itemsets. C_i : collection of candidate itemsets in instance i .

```

for ( $k = 2; F_{k-1} \neq \emptyset; k++$ ) {
   $C_k = \text{GenerateCandidates}(F_{k-1})$ 
  for all instances,  $i, i \in D$  {
     $C_i = \text{Subset}(C_k, i)$ 
    for all candidate itemsets,  $c, c \in C_i$ 
       $c.count++$  }
   $F_k = \{c \in C_k | c.count \geq min\_support\}$ 
  if ( $k = 2$ ) then {
     $j = 0$ 
    for all frequent normal items,  $n \in F_1$  {
       $j++$ 
       $B_n = \{u \in S | |u| = j\}$ 
      for all  $s \in B_n$ 
        if ( $\text{isFrequent}(n \cup s)$ )
          then  $F_2 = F_2 \cup (n \cup s)$ 
          else  $B_n = \{B_n - t | s \subseteq t\}$  } } }
return  $F = \cup_k F_k$ 

```

Fig. 2. Phase II of the SBA algorithm

4 Transformation Based Apriori (TBA)

We need to establish a “control” environment in which we can compare our SBA algorithm to the Apriori algorithm on the same dataset. Since Apriori does not directly handle set-valued attributes, we need to use a consistent transformation of set-valued attributes into normal attributes. It is not trivial to decide which transformation to use.

The One-to-Many Transformation replaces a set-valued attribute with many normal attributes. k binary attributes replace a set-valued attribute whose element domain is of size k . Each binary attribute represents one element from the set-value’s domain. The value for each of the normal attributes reflects whether the corresponding element is present in or absent from the original set-value. See Table 2.

Table 2. The results of a one-to-many transformation

Movie Title	<i>Stallone</i>	<i>Gibson</i>	<i>Glover</i>	<i>Rock</i>	Year
Rocky	<i>Yes</i>	<i>No</i>	<i>No</i>	<i>No</i>	1976
Lethal Weapon	<i>No</i>	<i>Yes</i>	<i>Yes</i>	<i>No</i>	1987
Lethal Weapon 3	<i>No</i>	<i>Yes</i>	<i>Yes</i>	<i>No</i>	1992
Lethal Weapon 4	<i>No</i>	<i>Yes</i>	<i>Yes</i>	<i>Yes</i>	1998

We propose a combination of one-to-many transformation with Apriori to give Apriori access to datasets with set-values. We call it *Transformation Based Apriori* (TBA).

Comparison of TBA vs. SBA In order to compare SBA with TBA, we identify two alternative choices at the interface between Phase I and Phase II of SBA. The choice to be made is simply whether to pass all or just some of the frequent set-values from Phase I to Phase II. When Phase II of SBA uses all of the frequent set-values from Phase I, SBA examines the same combinations of set-values with normal values that transformational Apriori does. The order, however, is very different. When Phase II of SBA uses only some of the frequent set-values from Phase I, SBA might examine fewer combinations of set-values with normal values. The best strategy for selecting which frequent set-values to discard is generally domain specific. In order to maintain general applicability, we must allow a user to keep all of the frequent set-values. Since in this case SBA performs with the same computational complexity as TBA, we have chosen TBA as the basis for our rule mining system.

5 Implementation and Evaluation of our Rule Mining System

We implemented our TBA system based on the excellent open-code provided with the WEKA [16] and the ARMiner [5] tools. We extended WEKA's attractive attribute-relation file format (ARFF) to include set-valued attributes. This extended format concisely and naturally represents set-valued data. We extended the ARMiner system to make it capable of mining classification rules and of mining rules from set-valued data.

5.1 The Spinal Muscular Atrophy (SMA) Dataset

Spinal muscular atrophy (SMA) is a genetic disease that affects cells in the spinal cord called anterior horn cells. It is estimated that 1 out of every 40 people carry the recessive gene for SMA. SMA is often fatal and death usually occurs in the very early childhood. There are different types of SMA, depending on the severity of the disease. Type I is the most severe. The SMA dataset for our experiments comes from [15] and gives the genotypic characteristics of 42 patients with SMA Types I, II, and III. The SMA dataset contains information about 9 types of SNP mutations and alleles for 2 microsatellite markers: C212 and Ag1-CA. C212 showed 15 different alleles and Ag1-CA showed 10 different alleles. The microsatellite marker for each person had between 2 and 4 allele values. The dataset also indicated with an underline which, if any, of the alleles

Table 3. A reduced sample of the SMA dataset

ID	Sex	Mutation	C212	Ag1-CA	SMA Type
1	female	Y272C	31- 28 29	102- 108 112	I
2	male	Y272C	28 29-34	108 112-106	I
3	male	Y272C	27 29- 28 29	114 116- 108 112	II
4	male	Y272C	28 29-29 30	108 112-110	III
5	male	T274I	del- 25 29	del- 108 114	II
...

came from the same parent as any observed mutation. For phenotypic characteristics, the dataset gives the SMA Type and the gender of the patient. See Table 3.

[6] mined classification rules, where the SMA Type (or severity) was the classification attribute. They transformed the original set-valued dataset into a normal one using what we call the one-to-many transformation. Their transformed dataset contains 71 binary attributes. They attempted to mine association rules using the original WEKA, but the tool could not complete the task due to the large number of frequent itemsets found in the transformed dataset, even when high values of minimal support were used. They then used CBA [3] to mine association rules from these data. Their results are difficult to interpret because since all of the alleles were absent more frequently than they were present, the rules which described allele absence “crowded out” rules that described allele presence. In contrast, the dataset represented in our extended ARFF contains just 6 attributes (2 of which are set-valued). We mined for classification rules with a support of at least 7% and a confidence of at least 50%. With these thresholds we discovered 120 rules, 56 of which had a confidence of 100%. A sample of the rules we discovered appear in Table 4. We found direct agreement with the literature stating that females were strongly association with the Type I SMA [11], and that alleles 28 and 29 for marker C212 and alleles 108 and 112 for marker Ag1-CA are related to SMA severity [15]. The allele 28 for marker C212 appears in 34 of the rules, while allele 29 occurs in 4 rules. For marker Ag1-CA, allele 108 is in 39 of the rules, and allele 112 is in 13 rules. Gender appeared in 52 of the 120 rules we mined.

Table 4. Sample classification rules obtained with our system for the SMA dataset.

Sample Rules	Supp %	Conf %
male & Mutation=none & Ag1-CA={108, 114} ⇒ SMA-III	14.3	100
C212={27, 28} ⇒ SMA-III	14.3	86
Ag1-CA={110} ⇒ SMA-III	14.3	75
female & C212={28u} & Ag1-CA={108u} ⇒ SMA-I	7.1	75

5.2 Gene Expression Data: The Promoters Dataset

We are interested in characterizing the expression patterns of genes based on their promoter regions. The promoter region of a gene controls both how and in which cells that

gene is expressed. The collection of cells in which a gene is expressed is called the *expression pattern* of the gene. A gene is expressed when a protein binds to the gene sequence and copies it. The promoter region contains shorter subsequences called motifs to which gene regulatory proteins may bind. Therefore, the pattern of motifs in the promoter region is said to regulate the expression of the gene.

Phu [10] collected genetic information about the promoter regions of many of the genes of the nematode *C.Elegans*, a microscopic worm. From the gene sequences, Murphy et al. identified potential motif sequences. In this Promoters dataset, there is one instance for each gene sequence. The attributes of the dataset are only the name of the gene sequence, and two set-valued attributes: the expression pattern and the motif pattern. There were 78 gene sequences in the dataset. The expression pattern for each sequence was which of 7 different cell types the gene was expressed in; the motif pattern for each sequence was which of 368 motifs were contained, at least once, in the sequence. Figure 5 shows a small sample of the Promoters dataset.

Table 5. A sample of the Promoters dataset

Sequence	Expression Pattern	Motif Pattern
+c32e8.7	Pan-Neural	M10 M12 M14 M15 M18 ...
+ceh-22	M-Cells	M1 M8 M10 M11 M17 M20 ...
+che-3	OLL ASK ASE	M1 M2 M3 M4 M6 M10 ...
+eat-4	OLL ASK	M1 M4 M6 M11 M20 M29 ...

Murphy et al. tried to mine classification association rules using CBA. However, there was no straightforward way to represent the data for input to CBA. They ended up dividing the dataset into multiple datasets, one for each cell or cell type used for classification. They had to combine the results of multiple mining trials for the multiple datasets. They were also faced with the problem of what they called “irrelevant rules”, that is rules that either referred to the absence of a motif or classified for the absence of expression in the cell type. For example, from the dataset for the OLL neural cell, CBA discovered 5596 classification rules, but they discarded 5518 of them as irrelevant.

In our experiments, we were able to exercise our system’s ability to mine classification rules that classify for a set-valued attribute. We constrained our rules to have consequents that were subsets of the element domain of the expression pattern attribute. Some of the rules our system discovered from this dataset appear in Table 6. No “irrelevant” rules were mined.

Table 6. Classification rules that our system discovers for the promoters dataset

Sample Rules (set-valued classification)	Supp.	Conf.
{M1, M2, M10, M20, M29, M36} ⇒ {Pan-Neural}	10%	62%
{M1, M2, M20, M29} ⇒ {ASE}	10%	67%
{M1} ⇒ {OLL, ASK}	5%	6%
...

6 Conclusions

The main contributions of our work are: (1) We have developed a system that mines association rules directly from set-valued data. To the best of our knowledge, this is the first system with that property. (2) Our system is unique in its ability to mine *classification* rules from data with a set-valued classification attribute. (3) Our system discovers association and classification rules that are easier to read and to interpret than rules mined by standard systems over transformed data.

References

1. R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In *Proc. 20th VLDB Conference*, pages 487–499, 1994.
2. Rakesh Agrawal, Tomasz Imielinski, and Arun Swami. Mining association rules between sets of items in large databases. *SIGMOD Records*, 22(2):207–216, June 1993.
3. Liu Bing, Wynne Hsu, Ma Yiming, Wong Ching Kian, Hu Mingqing, Xia Yiyuan, and Liu Jing. Classification based on associations (CBA). <http://www.comp.nus.edu.sg/~dm2/>.
4. W. Cohen. Learning trees and rules with set-valued features. In *Proc. 13th AAAI Conf.*, 1996.
5. L. Cristofor and D. Cristofor. Association rules miner (ARMiner). <http://www.cs.umb.edu/laur/ARMiner/>.
6. D. Doyle, J. Judecki, J. Lund, and B. Padovano. Genomic data mining. Undergraduate Graduation Project (MQP). Worcester Polytechnic Institute, April 2001.
7. D. Kalles and A. Papagelis. Induction of decision trees in numeric domains using set-valued attributes, 2000.
8. W. Lin, S.A. Alvarez, and C. Ruiz. Efficient adaptive-support association rule mining for recommender systems. *Data Mining and Knowledge Discovery*, 6(1):83–105, Jan. 2002.
9. Bing Liu, Wynne Hsu, and Yiming Ma. Integrating classification and association rule mining. In *Proc. 4th KDD Conf.*, pages 80–86, New York, August 1998.
10. B. Murphy, D. Phu, I. Pushee, and F. Tan. Motif- and expression-based classification of DNA. Undergraduate Graduation Project (MQP). Worcester Polytechnic Institute, April 2001.
11. G. Novelli, S. Semprini, F. Capon, and B. Dallapiccola. A possible role of naip gene deletions in sex-related spinal muscular atrophy phenotype variation. *Neurogenetics*, 1(1):29–30, 1997.
12. Terry R. Payne. Instance-based prototypical learning of set valued attributes, 1995. URL=citeseer.nj.nec.com/payne95instancebased.html.
13. C. Shoemaker, M. Pungliya, M. Sao Pedro, C. Ruiz, S.A. Alvarez, M. Ward, E. Ryder, and J. Krushkal. Computational methods for single point and multipoint analysis of genetic variants associated with a simulated complex disorder in a general population. *Genetic Epidemiology*, 21 (Suppl. 1):S738–S745, 2001.
14. C.A. Shoemaker, M.A. Sao Pedro, S.A. Alvarez, and C. Ruiz. Prediction vs. description: Two data mining approaches to the analysis of genetic data. In *Proc. 12th Genetic Analysis Workshop*, pages 449–453. Southwest Foundation for Biomedical Research, Oct. 2000.
15. B. Wirth, M. Herz, A. Wetter, S. Moskau, E. Hahnen, S. Rudnik-Schoeneborn, T. Wienker, and K. Zerres. Quantitative analysis of survival motor neuron copies: Identification of subtle smn1 mutations in patients with spinal muscular atrophy, genotype-phenotype correlation, and implications for genetic counseling. *American Journal of Human Genetics*, 64:1340–1356, 1999.
16. Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann Publishers, October 1999.