A Concise Representation of Association Rules using Minimal Predictive Rules [*]

Iyad Batal and Milos Hauskrecht

Department of Computer Science
University of Pittsburgh
{iyad,milos}@cs.pitt.edu

Abstract. Association rule mining is an important branch of data mining research that aims to extract important relations from data. In this paper, we develop a new framework for mining association rules based on minimal predictive rules (MPR). Our objective is to minimize the number of rules in order to reduce the information overhead, while preserving and concisely describing the important underlying patterns. We develop an algorithm to efficiently mine these MPRs. Our experiments on several synthetic and UCI datasets demonstrate the advantage of our framework by returning smaller and more concise rule sets than the other existing association rule mining methods.

1 Introduction

The huge amounts of data collected today provide us with an opportunity to better understand the behavior and structure of many natural and man-made systems. However, the understanding of these systems may not be possible without automated tools that enable us to extract the important patterns in the data and present them in a concise and easy to understand form.

Rule induction methods represent a very important class of knowledge discovery tools. The advantage of these methods is that they represent the knowledge in terms of if-then rules that are easy to interpret by humans. This can facilitate the process of discovery and utilization of new practical findings. As an example, consider a knowledge discovery problem in medicine. Assume a rule mining algorithm identifies a subpopulation of patients that respond better to a certain treatment than the rest of the patients. If the rule clearly and concisely defines this subpopulation, it can speed up the validation process of this finding and its future utilization in patient-management.

Association rule mining is a very popular data mining technique to extract rules from the data. The original framework [1] has been extended to mine patterns from various domains [28, 4, 16]. The key strength of association rule mining is that it searches the space of rules completely by examining all patterns that occur frequently in the data. However, the main disadvantage is that the number of association rules it finds is often very large. Moreover, many rules are redundant because they can be naturally explained by other rules. This may hinder the discovery process and the interpretability of the results. The objective of this work is to filter out these redundant rules and provide

[*] The original publication is available at www.springerlink.com

the user with a small set of rules that are sufficient to capture the essential underlying structure of the data.

In this work, we focus on the problem of mining association rules that target a specific class variable of interest. We call these rules the *class association rules*. To achieve our goal, we first introduce the concept of the *minimal predictive rules* (MPR) set that assures a good coverage of the important patterns with very small number of rules. After that, we propose an algorithm for mining these rules. Briefly, our method builds the MPR set by examining more general rules first and gradually testing and adding more specific rules to the set. The algorithm relies on a statistical significance test to ensure that every rule in the result is significantly better predictor than any of its generalizations.

2 Methodology

In this section, we first define basic terminology used throughout the paper. After that, we present an example illustrating the challenges of rule mining and the limitations of existing methods. Next, we propose the minimal predictive rules (MPR) framework to address them. Finally, we present an algorithm for mining the MPRs.

2.1 Definitions

Our work focuses on mining relational databases, where each record is described by a fixed number of attributes. We assume that all attributes have discrete values (numeric attributes must be discretized [14]). We call an attribute value pair an *item* and a conjunction of items a *pattern* (sometimes patterns are also called *itemsets*). If a pattern contains k items, we call it a *k-pattern* (an item is a 1-pattern). We say that pattern P' is a subpattern of pattern P if $P' \subset P$ (P is a superpattern of P'). A rule is defined as R: $A \Rightarrow c$, where A is a pattern and c is the class label that R predicts. We say that rule R': $A' \Rightarrow c'$ is a subrule of rule R: $A \Rightarrow c$ if c'=c and $A' \subset A$.

A pattern P can be viewed as defining a subpopulation of the instances (records) that satisfy P. Hence, we sometimes refer to pattern P as group P. If P' is a subpattern of P, then P' is a supergroup of P. Note that the empty pattern Φ defines the entire population. The support of pattern P, denoted as $sup(P)$, is the ratio of the number of records that contain P to the total number of records: $sup(P) \approx \Pr(P)$. The confidence of rule R: $A \Rightarrow c$ is the posterior probability of class c in group A: $conf(R) = sup(A \cup c)/sup(A) \approx \Pr(c|A)$. Note that confidence of the empty rule is the prior probability of the class: $conf(\Phi \Rightarrow c) \approx \Pr(c)$.

2.2 Example

Assume our objective is to identify populations which are at high risk of developing coronary heart disease (CHD). Assume that our dataset contains 200 instances and that the CHD prior is Pr(CHD)=30%. We want to evaluate the following 3 rules:

R1: Family history=yes ⇒ CHD
[*sup*=50%, *conf*=60%]

R2: Family history=yes $\wedge$ Race=Caucasian $\Rightarrow$ CHD
[*sup*=20%, *conf*=55%]
R3: Family history=yes $\wedge$ Race=African American $\Rightarrow$ CHD
[*sup*=20%, *conf*=65%]

From the above rules, we can see that a positive family history is probably an important risk factor for CHD because the confidence of R1 (60%) is two times higher than CHD prior (30%). However, the problem is that we expect many rules that contain a positive family history in their antecedents to have a high confidence as well. So how can we know which of these rules are truly important for describing the CHD condition?

The original association rules framework [1] outputs all the frequent rules that have a higher confidence than a minimum confidence threshold (*min_conf*). For instance, if we set *min_conf*=50%, all of three rules will be returned to the user.

In order to filter out some of the uninteresting associations, the original support-confidence framework is sometimes augmented with a correlation measure. Commonly, a χ^2 test is used to assure that there is a significant positive correlation between the condition of the rule and its consequent [8, 24, 21, 18]. However, because the posteriors of all three rules are much higher than the prior, we expect all of them to be statistically significant! Moreover, these rules will be considered interesting using most existing interestingness measures [15]. The main problem with this approach is that it evaluates each rule individually without considering the relations between the rules. For example, if we are given rule R2 by itself, we may think it is an important rule. However, by looking at all three rules, we can see that R2 should not be reported because it is more specific than R1 (applies to a smaller population) and has a lower confidence.

To filter out such redundant rules, [6] defined the confidence improvement constraint:

$$imp(A \Rightarrow c) = conf(A \Rightarrow c) - \max_{A' \subset A}\{conf(A' \Rightarrow c)\} > min\_imp$$

In practice, it is not clear how to specify this *min_imp* parameter. So the common convention is to set to zero ([18, 17]). This means that we only report the rules that have a higher confidence than all of their subrules. If we applied the confidence improvement constraint to our working example, rule R2 will be removed and rule R3 will be retained. However, R3 may also be unimportant and its observed improvement in the confidence can be due to chance rather than actual causality. In fact, there is a high chance that a refinement of a rule, even by adding random items, leads to a better confidence. We will see later in the analysis in section 2.4 and in the experimental evaluation that the confidence improvement constraint can still output many spurious rules. So should we report rule R3? To answer this question, we define the *minimal predictive rules* concept.

2.3 Minimal Predictive Rules (MPR)

Definition 1. *A rule R: A⇒c is a minimal predictive rule (MPR) if and only if* R *predicts class* c *significantly better than all its sub-rules.*

This definition implies that every item in the condition (*A*) is an important contributor to the predictive ability of the rule. We call these rules *minimal* because removing any non-empty combination of items from the condition would cause a significant drop in the predictability of the rule. An MPR can be viewed as defining a “surprising” subpopulation, where the posterior probability of class c in the group A ($\Pr(c|A)$) is unexpected and cannot be explained by any convex combinations of A's subpatterns.

The MPR significance test: In order to check if a rule is significant with respect to its subrules, we use the binomial distribution as follows: Assume we are interested in testing the significance of rule R: $A \Rightarrow c$. Assume that group A contains N instances, out of which N_c instances belong to class c. Assume P_c represents the highest confidence achieved by any subrule of R: $P_c = \max_{A' \subset A} \Pr(c|A')$. The null hypothesis presumes that N_c is generated from N according to the binomial distribution with probability P_c. The alternative hypothesis presumes that the true underlying probability that generated N_c is significantly higher than P_c. Hence, we perform a *one sided* significance test (we are interested only in increases in the proportion of c) and calculate a p-value as follows:

$$p = Pr_{binomial}(x \geq N_c | N, P_c)$$

If this p-value is significant (smaller than a significance level α), we conclude that R significantly improves the predictability of c over all its subrules, hence is an MPR.

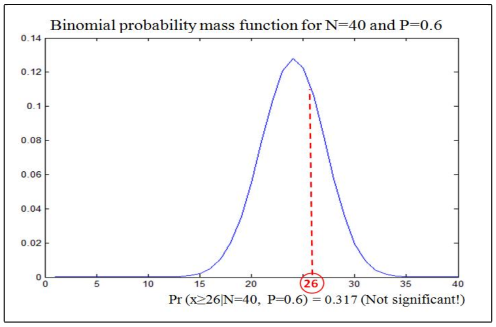


Fig. 1. The MPR significance test for rule R3

Example: Going back to our CHD example, rule R3 covers 40 instances, out of which 26 have CHD. For R3 to be an MPR, it should be significantly more predictive than all its simplifications, including rule R1. By applying the MPR significance test we get: $Pr_{binomial}(x \geq 26|40, 0.6) = 0.317$. As illustrated in Figure 1, we can see that R3 is not an MPR at significance level $\alpha = 0.05$. On the other hand, if we use the same binomial significance test to evaluate each rule individually against the CHD prior (by always setting P_c=Pr(CHD)), the p-values we get for R1, R2 and R3 are respectively, 5.13e-10, 8.54e-4 and 5.10e-6, meaning that all three rules are (very) significant!.

2.4 Spurious patterns and redundant rules

In this section, we discuss and analyze the serious problem of redundancy in association rules. This problem is due to the manner in which large numbers of spurious rules are formed by adding irrelevant items to the antecedent of other rules. We show the deficiencies of the current approaches to dealing with this problem. Finally, we show how MPR can overcome the problem.

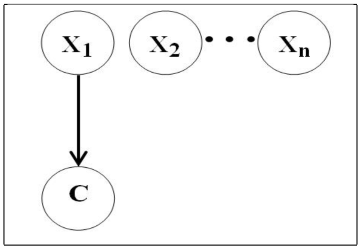


Fig. 2. Illustrating the redundancy problem in association rules

Consider a Bayesian belief network example in Figure 2, where we have a causal relation between variable X_1 and the class variable C. Assume that item X_1=1 is highly predictive of class c_1, so that Pr(C=c_1 | X_1=1) is significantly larger than Pr(C=c_1). Assume we have another variable X_2 that is completely independent of C: $X_2 \perp\!\!\!\perp C$. If we add any instantiation v_2 of variable X_2 to item X_1=1, the posterior distribution of c_1 is Pr(C=c_1 | X_1=1 $\wedge$ X_2=v_2) $\approx$ Pr(C=c_1 | X_1=1), i.e., *conf*(X_1=1 $\wedge$ X_2=v_2 $\Rightarrow$ c_1) $\approx$ *conf*(X_1=1 $\Rightarrow$ c_1).

More generally, if we have many irrelevant variables such that $X_i \perp\!\!\!\perp C$: $i \in \{2..n\}$, the network structure implies that Pr(C=c_1 | X_1=1 $\wedge$ X_2=v_2 ... $\wedge$ X_n=v_n) $\approx$ Pr(C=c_1 | X_1=1) for every possible instantiation X_i=v_i. Clearly, the number of such spurious rules can become huge, which can easily overwhelm the user and prevent him from understanding the real patterns and causalities in the data.

Even by requiring the complex rules to have a higher confidence [6, 17, 18] or lift score [9] than their simplifications, the problem still exists and many of these redundant rules can easily satisfy this constraint. In fact, if X_i is a binary variable and *conf*(X_1=1 $\wedge$ X_i=0 $\Rightarrow$ c_1) $<$ *conf*(X_1=1 $\Rightarrow$ c_1), then we know for sure that *conf*(X_1=1 $\wedge$ X_i=1 $\Rightarrow$ c_1) $>$ *conf*(X_1=1 $\Rightarrow$ c_1). The same situation happens if we use the lift score instead of the confidence! Post-pruning the rules individually based on their correlations with the class (e.g. using the χ^2 test [8, 24, 21, 18]) or based on their difference from the prior (e.g. using our binomial test) is not going to not help in this case.

Our frameworks tackles this problem because every item in an MPR should significantly contribute to improving the predictability of the rule. This means that if rule R: X_{q_1}=v_{q_1} ... $\wedge$ X_{q_k}=$v_{q_k}$$\Rightarrow$$C$=$c$ is an MPR, then there should exist a path from each variable X_{q_i} to the class C that is not blocked (d-separated) by the other variables in the rule: X_{q_i} not $\perp\!\!\!\perp C \mid \{ X_{q_1}, X_{q_{i-1}}, ..., X_{q_{i+1}}, X_{q_k} \}$. Therefore, redundant rules are likely to be filtered out.

2.5 The Algorithm

In this section we explain our algorithm for mining the MPR set. The algorithm is outlined in Figure 3. Briefly, the algorithm explores the space by performing a level wise Apriori-like search. At each level (*l*), we first remove the candidate *l-patterns* that do not pass the minimum support threshold (line 6). Then we extract all MPRs from these frequent *l-patterns* (line 7). Finally, we generate the candidates for the next level (line 8).

Extracting MPRs The process of testing if a rule $P \Rightarrow c$ is an MPR is not trivial because the definition requires us to check the rule against all its proper subrules. This

Algorithm 1: Mine all *MPRs*

```
Input: dataset: $D$, minimum support: min_sup
Output: minimal predictive rules: MPR
      // global data structure
01:   MPR=$\Phi$, tbl_max_conf=hashtable()
      // the prior distribution of the classes
02:   tbl_max_conf[$h(\Phi)$]=calculate_class_distribution($\Phi$, $D$)
03:   $Cand$=generate_1_patterns()
04:   $l = 1$
05:   while (Cand $\neq \Phi$)
          // remove candidates that are not frequent
06:       FP[$l$]=prune_infrequent(Cand, $D$, min_sup)
          // find all MPRs at level $l$
07:       extract_MPR(FP[$l$], $D$)
          // generate candidate (l+1)_patterns
08:       Cand=generate_candidates(FP[$l$])
09:       $l = l + 1$
10:   MPR=FDR_correction(MPR)
11:   return MPR
```

Fig. 3. The algorithm for mining MPRs from a dataset

would require to check 2^l subpatterns if P has length l. Our algorithm avoids this inefficiency by caching the statistics needed to perform the check within the *(l-1)-subpatterns* from the previous level. This part of the algorithm is outlined in Figure 4.

To explain the method, it is useful to envision the progress of the algorithm as gradually building a lattice structure level by level, starting from the empty pattern Φ. An example lattice is shown in Figure 5. Every frequent *l-pattern P* is a node in the lattice with *l* children: one child for each of its *(l-1)-subpatterns*. The key idea of our algorithm is to store in each node P the maximum confidence score for every class that can be obtained in the sublattice with top *P* (including P itself): $max\_conf_P[c]$=max(Pr($c \mid P'$)): $\forall\, P' \subseteq P$. These *max_conf* values are computed from the bottom up as algorithm progresses. Initially, max_conf_Φ for the empty pattern is set to be the prior distribution of the class variable. In order to compute max_conf_P for pattern P, we first compute $conf_P$ (line 2), the distribution of the class variable in group P: $conf_P[c]$=Pr($c \mid P$). Then we use the *max_conf* values of P's direct children to compute $max\_conf\_children_P$ (line 3) so that $max\_conf\_children_P[c]$=max(Pr($c \mid P'$)): $\forall\ P' \subset P$. Finally, we compute max_conf_P by taking the element-wise maximum of two arrays: $conf_P$ and $max\_conf\_children_P$ (line 4).

After assigning the *max_conf* value for pattern P, we want to check if P forms an MPR. So for each class label c, we perform the MPR significance test to check if P predicts c significantly better than $max\_conf\_children_P[c]$. If the test is positive, we add the rule $P \Rightarrow c$ to the set of MPRs (line 8).

Please note that in our pseudo-code, we do not explicitly build the lattice. Instead, we use a hash table *tbl_max_conf* (Figure 3: line 1) to provide direct access to the *max_conf* values, so that *tbl_max_conf*[$h(P)$]=max_conf_P, where h is a hash function. Also note that none of the functions (*calculate_class_distribution*, *is_MPP* and *loss-*

```
Algorithm 2: extract_MPR (FP[$l$], D)
//add pattern $P \in$ FP[$l$] to MPR (a global variable) if $P$ is
significantly more predictive than all its subpatterns
for each $P \in$ FP[$l$]
    conf=calculate_class_distribution(P, D)
    max_conf_children=max {tbl_max_conf[$h(S_{l-1})$]} : $S_{l-1} \subset P$
    max_conf=max{ conf, max_conf_children }
    tbl_max_conf[$h(P)$]=max_conf
    for each class label $c$
        if ( is_MPR(P, $c$, max_conf_children, D) )
            MPR=MPR $\cup$ ($P \Rightarrow c$)
    lossless_pruning(P, max_conf, D, FP[$l$])

Function is_MPR(P, $c$, max_conf_children, D)
  //return true if $P$ predicts $c$ significantly better than all its subpatterns
  N=count($P$, D)
  $N_c$=count($P \cup c$, D)
  p_value=$Pr_{binomial}(x \geq N_c \mid N$, max_conf_children[$c$])
  if(p_value $< \alpha$)
     return true
  return false

Function lossless_pruning(P, max_conf, D, FP[$l$])
  //prune $P$ if it cannot produce any MPR
  for each class label $c$
     $N_c$=count($P \cup c$, D)
     p_value=$Pr_{binomial}(x \geq N_c \mid N_c$, max_conf[$c$])
     //exit if $P$ can potentially produce an MPR for any class $c$
     if(p_value $< \alpha$)
        return ;
  //Prune $P$
  remove($P$, FP[$l$])
```

Fig. 4. The algorithm for extracting MPRs from the frequent patterns at level l

less_pruning) requires another scan on the data because we can collect the class specific counts during the same scan that counts the pattern.

Figure 5 illustrates the algorithm using a small lattice on a dataset that contains 200 instances from class c_1 and 300 instances from class c_2. For each pattern P (node), we show the number of instances in each class, the distribution of the classes (*conf*) and the maximum confidence from P's sublattice (*max_conf*). Let us look for example at pattern $I_1 \wedge I_2$. This pattern is predictive of class c_2: *conf*$(I_1 \wedge I_2 \Rightarrow c_2) = 0.75$. However, this rule is not an MPR since it does not significantly improve the predictability of c_2 over the subrule $I_2 \Rightarrow c_2$: $Pr_{binomial}(x \geq 75|100, 0.7) = 0.16$ (not significant at α=0.05). The MPRs from this example are: $I_1 \Rightarrow c_1$, $I_2 \Rightarrow c_2$ and $I_1 \wedge I_3 \Rightarrow c_1$.

Mining at low support It is well know that the performance of the Apriori algorithm highly depends on the minimum support (*min_sup*) parameter. However, setting this parameter is not straightforward because the optimal *min_sup* can vary greatly across

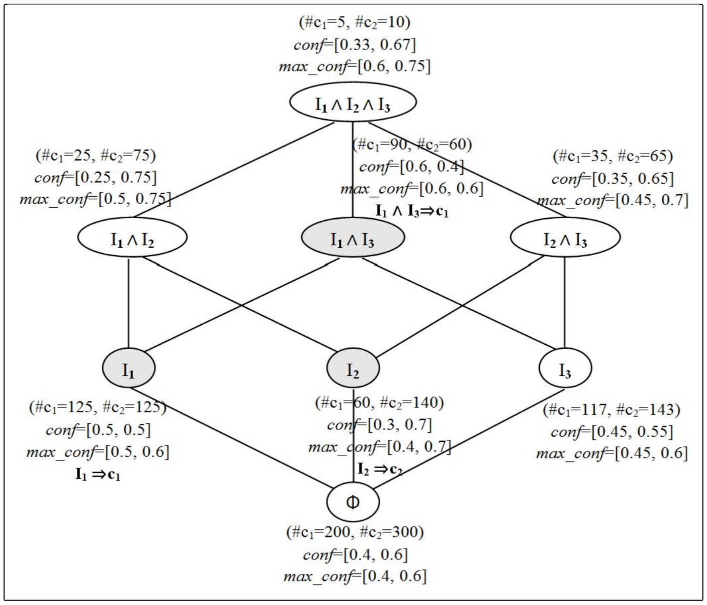


Fig. 5. An illustrative example showing the lattice associated with frequent pattern $I_1 \wedge I_2 \wedge I_3$. The MPRs are shaded.

different datasets. In fact, [11, 9] argued that it is sometimes of interest to mine low support patterns. Therefore, in order not to miss any important pattern, the user may choose a low *min_sup* value. However, low *min_sup* raises two important concerns. First, the algorithm may return a huge number of rules, most which are very complex. Second, the algorithm may take very long time to finish.

In the following we argue that the MPR framework and our algorithm address both of these concerns. First, by requiring each MPR to be significantly better than all its subrules, the algorithm is biased towards choosing simple rules over more complex rules. Moreover, the MPR significance test incorporates the pattern's support because the chance of passing the test is lower for low support patterns. This acts as if the *min_sup* filter was built into the statistical significance test. Hence, very low support patterns are likely to be filtered out unless they are extremely predictive (with a very surprising distribution).

Second, the MPR significance test can help us to prune the search space. This early pruning is implemented by the *lossless_pruning* function in Figure 4. The idea is that we can prune pattern *P* if we guarantee that *P* cannot produce any MPR. However, because the pruning is applied while generating the patterns, we do not know what subgroups P will generate further in the lattice. To overcome this, let us define the *optimal subgroup* $P^*_{c_i}$ in group P with respect to class c_i to be the subgroup that contains all the instances from c_i and none of the instances from any other classes. Clearly, P cannot generate any subgroup better than $P^*_{c_i}$ for predicting class c_i. Now, we prune P if for every class c_i, $P^*_{c_i}$ is not significant with respect to the best confidence so far *max_conf*$[c_i]$. Please note that this pruning technique does not miss any MPR because the *lossless_pruning* test is anti-monotonic.

As an example, consider pattern $P = I_1 \wedge I_2 \wedge I_3$ in Figure 5. This pattern contains 15 examples, 5 from class c_1 and 10 from class c_2. Both $P^*_{c_1}$ and $P^*_{c_2}$ are not significant with respect to the current best predictions 0.6 and 0.75 (respectively). Therefore, there is no need to further explore P's subgroups and the entire sublattice can be safely pruned.

Correcting for multiple testing When multiple rules are tested in parallel for significance, it is possible to learn a number of false rules by chance alone. This is a common problem for all techniques that rely on statistical tests. For example, assume we have 10,000 random items (independent of the class variable). If we test the significance of each of them with a significance level α=0.05, then we expect about 500 items to be significant just by chance!

One approach to deal with the multiple hypothesis testing problem is to adjust the significance level at which each rule is tested. The most common way is the Bonferroni correction [25], which divides the significance level (α) by the number of tests performed. This approach is not suitable for rule mining because the number of rules tested is usually very large, resulting in an extremely low α and hence very few rules discovered. The other more recent approach, directly controls the false discovery rate (FDR) [7]: the expectation of the proportion of false discoveries in the result. We adopt the FDR correction method because it is less stringent and more powerful (has a lower Type II error) than the Bonferroni correction. We apply FDR as a post-processing step (Figure 3: line 10). It takes as input all potential MPRs with their p-values and outputs a subset of MPRs that satisfy the FDR criteria.

3 Experiments

In this section we present our experimental evaluation, first on synthetic datasets with known underlying patterns, and after that on several UCI classification datasets [2]. The experiments compare the performance of MPR against the following methods:

- ***complete:*** The set of all rules that cover more than *min_sup* examples in the data. We filter out useless rules by only including the ones that positively predict the class label (with a lift score [15] higher than one).
- ***closed:*** A subset of *complete* that corresponds to non-redundant rules based on the concept of closed frequent patterns [3].
- ***corr_chi:*** A subset of *complete* that contains the rules with significant positive correlations between the condition and conclusion according to the χ^2 test [8, 24, 21].
- ***prior_binom:*** A subset of *complete* that contains the rules that are significantly different from the prior according to the binomial statistical test (section 2.3).
- ***prior_FDR:*** A subset of *prior_binom* that is post-processed using the false discovery rate (FDR) technique [7] to correct for the multiple testing problem.
- ***conf_imp:*** A subset of *complete* that satisfies the confidence improvement constraint [6, 18, 17]: each rule should have a higher confidence than all its subrules.

For all methods, we set the minimum support threshold (*min_sup*) to 10% the number of records in the dataset (unless otherwise stated). For the methods that use a statistical test: *corr_chi*, *prior_binom*, *prior_FDR* and *MPR*, we use the conventional significance level $\alpha = 0.05$.

3.1 Experiments on synthetic data

The experiments on synthetic data (generated from pre-defined patterns) allow us to judge more objectively the quality of the algorithms' outputs by comparing the original and recovered patterns.

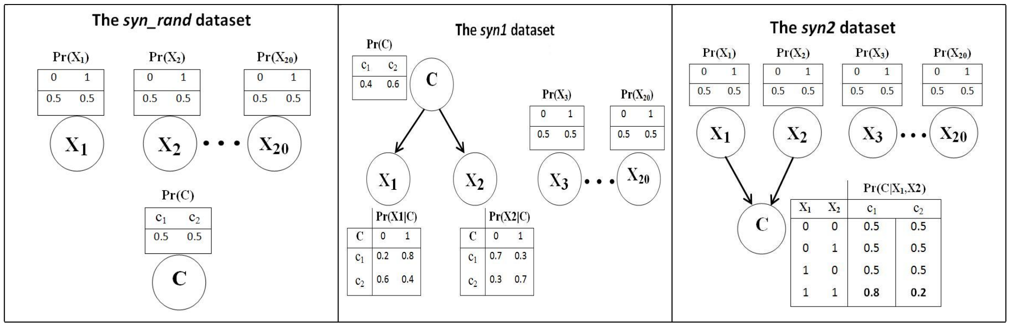


Fig. 6. Three Bayesian belief networks used to generate synthetic datasets: *syn_rand*, *syn1* and *syn2* (from left to right)

Data description The synthetic data were generated from the three Bayesian belief networks (BBNs) in Figure 6. Each network consists of 20 random binary variables $\{X_1, ..., X_{20}\}$ and two class labels c_1 and c_2. The three networks are:

- ***syn_rand:*** In this network, all the attribute variables X_i and the class variable C are independent (*syn_rand* does not contain any pattern).
- ***syn1:*** In this network, X_1=1 is more frequent in class c_1: Pr(X_1=1 | C=c_1)=0.8 > Pr(X_1=1 | C=c_2)=0.4 and item X_2=1 is more frequent in class c_2: Pr(X_2=1 | C=c_2)=0.7 > Pr(X_2=1 | C=c_1)=0.3. Besides, the distribution of the class variable C is more biased towards c_2: Pr(C=c_2)=0.6. The attributes $\{X_3, ..., X_{20}\}$ are independent of each other and the class variable C.
- ***syn2:*** The main pattern in *syn2* is a conjunction of X_1=1 and X_2=1 that predicts class c_1: Pr(C=c_1 | X_1=1 and X_2=1)=0.8. For all other values of X_1 and X_2, the two classes (c_1 and c_2) are equally likely. The attributes $\{X_3, ..., X_{20}\}$ are independent of each other and the class variable C.

The datasets we analyze consist of 1000 examples randomly generated from these three networks.

Results: Table 1 summarizes the number of rules that each method produces on the three synthetic datasets. First notice that the number of rules in *complete* and *closed* are the same for all these datasets (since very few correlations exist). Also notice that *corr_chi* and *prior_binom* have similar results. This shows that the choice of the statistical test does not matter much and that the real problem with these approaches is that they evaluate each rule separately without considering the nested structure of the rules.

Let us look first at the results on the *syn_rand* dataset, which we know does not contain any pattern. MPR does not output any rule which is what we expect given that

Dataset	*complete*	*closed*	*corr_chi*	*prior_binom*	*prior_FDR*	*conf_imp*	*MPR*
syn_rand	9,763	9,763	516	651	0	4,748	**0**
syn1	9,643	9,643	2,989	3,121	2,632	4,254	**6**
syn2	9,868	9,868	1,999	2,225	422	3,890	**5**

Table 1. The number of rules for the different methods on the synthetic datasets

the class is independent of all attributes. On the other hand, *conf_imp* returns a huge number of rules. In fact, *conf_imp* returns almost half the total number of associations. This result agrees with our analysis in section 2.4. *prior_FDR* correctly removes all false positives from *prior_binom* on this random data.

Now consider the *syn1* dataset. Our algorithm extracts the following 6 MPRs: {X1=1 ⇒ c1, X1=0 ⇒ c2, X2=1 ⇒ c2, X2=0 ⇒ c1, X1=0 ∧ X2=1 ⇒ c2, X1=1 ∧ X2=0 ⇒ c1}[1]. In comparison, all other methods extract a vast number of rules. For example, even by using the FDR correction, the number of rules is 2,632 rules!

MPR	*prior_binom*	*conf_imp*
X1=1 ∧ X2=1 ⇒ c1 [*sup*=26.5%, *conf*=81.1%]	X1=1 ∧ X2=1 ⇒ c1 [*sup*=26.5%, *conf*=81.1%]	X1=1 ∧ X2=1 ∧ X8=0 ⇒ c1 [*sup*=12.3%, *conf*=85.4%]
X2=1 ⇒ c1 [*sup*=50.1%, *conf*=66.7%]	X1=1 ∧ X2=1 ∧ X14=0 ⇒ c1 [*sup*=14.5%, *conf*=84.8%]	X1=1 ∧ X2=1 ∧ X14=0 ⇒ c1 [*sup*=14.5%, *conf*=84.8%]
X1=1 ⇒ c1 [*sup*=51.2%, *conf*=63.9%]	X1=1 ∧ X2=1 ∧ X13=1 ⇒ c1 [*sup*=13.4%, *conf*=84.3%]	X1=1 ∧ X2=1 ∧ X13=1 ⇒ c1 [*sup*=13.4%, *conf*=84.3%]
X2=0 ⇒ c2 [*sup*=49.9%, *conf*=51.5%]	X1=1 ∧ X2=1 ∧ X9=1 ⇒ c1 [*sup*=14.1%, *conf*=83.7%]	X1=1 ∧ X2=1 ∧ X9=1 ⇒ c1 [*sup*=14.1%, *conf*=83.7%]
X1=0 ⇒ c2 [*sup*=48.8%, *conf*=49.0%]	X1=1 ∧ X2=1 ∧ X8=0 ⇒ c1 [*sup*=12.3%, *conf*=85.4%]	X1=1 ∧ X2=1 ∧ X18=0 ⇒ c1 [*sup*=13.9%, *conf*=83.5%]

Table 2. The syn2 dataset: on the left is the set of all MPRs, in the middle is the top 5 *prior_binom* rules (out of 2,225 rules) and on the right is the top 5 *conf_imp* rules (out of 3,890 rules)

Finally, let us look more closely at the results on the *syn2* dataset. Table 2 shows all MPRs (left), the top 5 ranked rules for *prior_binom* (same for *prior_FDR*) according to the p-values (center) and the top 5 ranked rules for *conf_imp* according to the confidence (right). Notice that *prior_binom* correctly ranks the real pattern (X_1=1 ∧ X_2=1) at the top. However, the following rules are redundant. For *conf_imp*, the real pattern is buried inside many spurious rules. This example clearly shows the deficiencies of these methods in concisely representing the actual patterns. On the contrary, by investigating the small number of MPRs, we can easily recover the structure of the underlying BBN.

3.2 Experiments on UCI datasets

Data description To further test the different methods, we use 9 public datasets from the UCI Machine Learning repository [2]. We discretize the numeric attributes into

[1] The last 2 rules combine the evidence of the two main patterns, hence improve the predictability. For example, Pr(c1|x1=1 ∧ x2=0) > Pr(c1|x1=1).

intervals by minimizing the entropy based on the minimum description length principle [14] (supervised discretization). Table 3 shows the main characteristics of the datasets. The number of items in column 3 is the number of all distinct attribute value pairs.

dataset	# attributes	# items	# records	# classes
Adults	14	154	32,561	2
Heart disease	13	33	303	2
Lymphography	18	57	142	2
Pima diabetes	8	19	768	2
Breast cancer	9	41	286	2
Dermatology	12	47	366	6
Wine	13	39	178	3
Glass	10	22	214	2
Credit	15	69	690	2

Table 3. UCI Datasets characteristics

Results Table 4 shows the number of rules for each method on the 9 UCI datasets. First notice that evaluating the rules individually based on their statistical significance (*corr_chi*, *prior_binom* and *prior_FDR*) does not help much in reducing the number of rules (even by using the FDR correction). It is clear from the table that the number of MPRs is much smaller than the number of rules in the other approaches. On average, MPRs are about two orders of magnitude smaller than *complete* and about one order of magnitude smaller than *conf_imp*.

Dataset	*complete*	*closed*	*corr_chi*	*prior_binom*	*prior_FDR*	*conf_imp*	*MPR*
Adults	2,710	2,042	2,616	2,620	2,619	374	**152**
Heart disease	5,475	5,075	4,820	4,825	4,784	1,021	**79**
Lymphography	31,594	5,840	15,740	15,032	11,627	978	**24**
Pima diabetes	466	448	345	350	337	144	**36**
Breast cancer	420	379	122	124	44	158	**10**
Dermatology	5,350	3,727	3,820	3,717	3,544	2,076	**96**
Wine	1,140	1,057	975	971	968	520	**116**
Glass	2,327	1,141	2,318	2,311	2,311	97	**20**
Credit	8,504	3,271	6,885	6,964	6,839	926	**49**
Average	6,443	2,553	4,182	4,102	3,675	699	**65**

Table 4. The number of rules for the different algorithms on several UCI datasets

Now we need a way to check if this small set of MPRs can adequately describe the datasets. However, we do not know what are the real patterns for these datasets. Hence, to evaluate the usefulness of the rules, we use the classification accuracy of the corresponding rule based classifier. We define two simple classifiers:

- Weighted confidence classification (*w_conf*): To classify instance x, we weight each rule that satisfy x by its confidence and we choose the class that maximizes this weighted sum:

$$w\_conf(x) = \arg\max_{c_i}\{ \sum_{x \subseteq A} conf(A \Rightarrow c_i) \}$$

– Highest confidence classification (*h_conf*): We classify instance x according to the highest confidence rule that satisfy x (this method is used in [20]):

$$h\_conf(x) = \arg\max_{c_i}\{ \max_{x \subseteq A} conf(A \Rightarrow c_i) \}$$

We compare the classification accuracies obtained by using all association rules, using *conf_imp* rules (this approach was used in [17] for classification), and using MPRs[2]. All of the reported results are obtained using 5-folds cross validation. Remember that the lift score for all rules is bigger than one (the condition and consequent of each rule are positively correlated). Hence, *w_conf* consults only predictive rules.

	complete		*conf_imp*		MPR	
Dataset	w_conf	h_conf	w_conf	h_conf	w_conf	h_conf
Adults	77.3	75.9	80.6	75.9	80.8	75.9
Heart disease	80.9	80.5	80.2	80.5	82.2	81.5
Lymphography	81.2	83.6	71.8	83.6	86.2	85.9
Pima diabetes	71.4	74.4	72.8	74.4	71.8	72.9
Breast cancer	73.8	72.0	74.1	72.0	72.4	73.4
Dermatology	68.0	69.4	58.2	69.4	63.4	65.6
Wine	88.8	93.3	86.4	93.3	88.2	92.8
Glass	94.4	100	93.5	100	95.8	100
Credit	80.4	85.4	76.7	85.4	77.8	85.4
Average	79.6	81.6	77.1	81.6	79.8	81.5

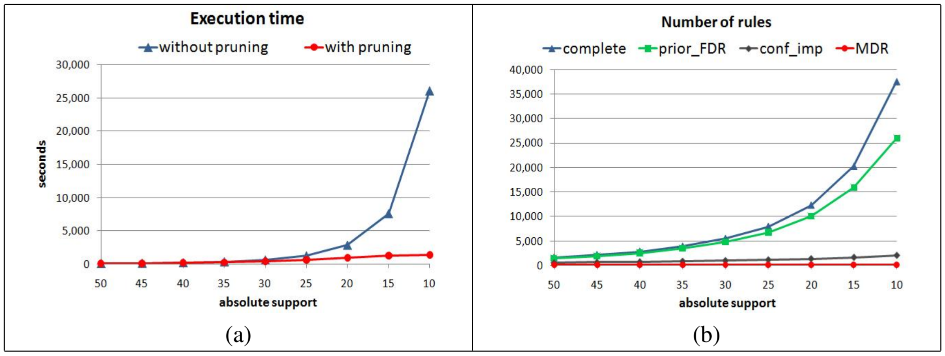
Table 5. The classification accuracies (%) for *complete*, *conf_imp* and MPR using two rule classification techniques: weighted confidence classification (*w_conf*) and highest confidence classification (*h_conf*)

From Table 5, we can see that MPR does not sacrifice the classification accuracy. On average, all approaches produce comparable results for both *w_conf* and *h_conf*. An important benefit of using the compact set of MPRs for classification is that the classification time is very fast. For example, consider the Lymphography dataset. Instead of consulting 31,594 rules to classify each instance, MPR summarizes the classifier in only 24 rules. It is interesting to see that these 24 rules outperform the complete set of rules for both *w_conf* and *h_conf*.

Please note that we are not claiming that our approach can outperform the state-of-the-art frequent pattern-based classifiers [13, 10]. Our objective is just to show that even though MPRs provide a huge compression of the association rules, they can still capture the essential underlaying patterns in the data by providing comparable classification performance to using all association rules.

[2] *corr_chi*, *prior_binom* and *prior_FDR* gave similar classification results as *complete*, hence we excluded them from this table to save space.

Mining at low support In this section, we study the performance and the output of the different methods under different support thresholds. Due to the space limitation, we only show the results on the *Heart disease* dataset.



(a) (b)

Fig. 7. The execution times (a) and the number of rules (b) of the algorithms on the heart disease dataset for different support thresholds

Figure 7:a shows the execution times using a Dell Precision T7500 machine with an Intel Xeon 3GHz CPU and 16GB of RAM. All algorithms are implemented in matlab. The "without pruning" chart corresponds to the execution of the Apriori algorithm that relies only on the support of the patterns to prune the search space. The "with pruning" chart corresponds to the execution of the algorithm that applies the additional MPR pruning technique in section 2.5.

We can see that the execution time of Apriori exponentially blows up for low support values. On the other hand, the MPR pruning controls the complexity and the execution time increases very slowly for low support values. For example, when the absolute support threshold is 15, which corresponds to *min_sup* = 5% on this dataset, applying the MPR pruning makes the algorithm about 6 times faster.

Figure 7:b shows the number of rules generated by the different methods. To improve the visibility, we did not include *closed*, *prior_binom* and *corr_chi* in this graph. We can see that the output of MPR does not change much when *min_sup* is very low. For example, by changing *min_sup* from 10% (absolute support = 30) to 5% (absolute support=15), the number of MPRs increases from 79 to 83 rules. In comparison, the same change causes the number of all association rules to increase from 5,475 to about 20,000 rules! Clearly, this large number of rules is overwhelming the user.

To summarize, our framework relieves the user from the burden of deciding the optimal *min_sup* by allowing him to conservatively set the support very low without drastically affecting the performance or the results of the algorithm.

4 Related Research

Several research attempts have been made to reduce the large number of association rules in order to make the results more suitable for knowledge discovery. Constrained associations rules methods [22] allow the user to define his own constraints and retrieve the rules that match these constraints. An opposite approach [23] mines the rules that are most different from the user's expectations. Maximal frequent itemsets [19] is a lossy compression of the frequent itemsets and cannot be used to generate rules. The profile based approach [27] is another lossy compression method.

The work in [6] aimed to reduce the number of class association rules by defining the confidence improvement constraint. This constraint was adopted by [18, 17]. As we showed in the analysis and experiments, this approach can still generate many redundant rules. [21] defined the concept of direction setting (DS) rules in order to make browsing the class association rules easier for the user. However, their objective is different from ours because non-DS rules can indeed be significant MPRs. [5] extends the problem of association rule mining to the problem of mining contrasting sets. [26] defines a measure to rank the patterns by predicting the support of a pattern from the support of its subpatterns and measuring the deviation between the actual support and the prediction.

5 Conclusion

In this paper, we have developed a new framework for mining association rules based on the minimal predictive rules (MPR) concept. We showed that our method can produce a small set of predictive rules. Most importantly, each rule in the result is important because it concisely describes a distinct pattern that cannot be explained by any other rule in the set.

Motivated by our results, we plan to investigate the benefits of MPR for classification. In particular, we plan on incorporating the MPRs as additional features with the SVM classifier and comparing it against the state-of-the-art classifiers [13, 10]. In addition, we plan to apply our method for anomaly detection in categorical data [12].

Acknowledgment
This research work was supported by grants 1R21LM009102-01A1, 1R01LM010019-01A1, and 1R01GM088224-01 from the NIH. Its content is solely the responsibility of the authors and does not necessarily represent the official views of the NIH.

References

1. R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *Proceedings of SIGMOD*, pages 207–216, 1993.
2. A. Asuncion and D.J. Newman. UCI machine learning repository, 2007.
3. Y. Bastide, N. Pasquier, R. Taouil, G. Stumme, and L. Lakhal. Mining minimal non-redundant association rules using frequent closed itemsets. In *Proceedings of the First International Conference on Computational Logic*, 2000.
4. I. Batal, L. Sacchi, R. Bellazzi, and M. Hauskrecht. Multivariate time series classification with temporal abstractions. In *FLAIRS*, 2009.
5. S. Bay and M. Pazzani. Detecting group differences: Mining contrast sets. *Data Mining and Knowledge Discovery*, 5(3):213–246, 2001.
6. R. J. Bayardo, R. Agrawal, and D. Gunopulos. Constraint-based rule mining in large, dense databases. In *Proceedings of ICDE*, pages 188–197, 1999.

7. Y. Benjamini and Y. Hochberg. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society. Series B (Methodological)*, 57(1):289–300, 1995.
8. S. Brin, R. Motwani, and C. Silverstein. Beyond market baskets: Generalizing association rules to correlations. In *Proceedings of SIGMOD*, 1997.
9. R. Castelo, A. J. Feelders, and A. Siebes. Mambo: Discovering association rules based on conditional independencies. In *Advances in Intelligent Data Analysis (IDA)*, 2001.
10. H. Cheng, X. Yan, J. Han, and C. Hsu. Discriminative frequent pattern analysis for effective classification. In *Proceedings of ICDE*, 2007.
11. E. Cohen, M. Datar, S. Fujiwara, A. Gionis, P. Indyk, R. Motwani, J. D. Ullman, and C. Yang. Finding interesting associations without support pruning. In *Proceedings of ICDE*, 2000.
12. K. Das, J. Schneider, and D. Neill. Anomaly pattern detection in categorical datasets. In *Proceedings of SIGKDD*, 2008.
13. W. Fan, K. Zhang, H. Cheng, J. Gao, X. Yan, J. Han, P. Yu, and O. Verscheure. Direct mining of discriminative and essential frequent patterns via model-based search tree. In *Proceedings of SIGKDD*, 2008.
14. U. Fayyad and K. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *Proceedings of IJCAI*, 1993.
15. L. Geng and H. Hamilton. Interestingness measures for data mining: A survey. *ACM Comput. Surv.*, 38(3), 2006.
16. M. Kuramochi and G. Karypis. Frequent subgraph discovery. In *Proceedings of ICDM*, 2001.
17. J. Li, H. Shen, and R. Topor. Mining optimal class association rule set. In *Proceedings of PAKDD*, 2001.
18. W. Li, J. Han, and J. Pei. CMAR: Accurate and efficient classification based on multiple class-association rules. In *Proceedings of ICDM*, 2001.
19. D. Lin and Z. Kedem. Pincer-search: A new algorithm for discovering the maximum frequent set. In *Proceedings of EDBT*, pages 105–119, 1997.
20. B. Liu, W. Hsu, and Y. Ma. Integrating classification and association rule mining. In *Knowledge Discovery and Data Mining*, pages 80–86, 1998.
21. B. Liu, W. Hsu, and Y. Ma. Pruning and summarizing the discovered associations. In *Proceedings of SIGKDD*, 1999.
22. R. Ng, L. Lakshmanan, J. Han, and A. Pang. Exploratory mining and pruning optimizations of constrained associations rules. In *Proceedings of SIGMOD*, 1998.
23. B. Padmanabhan and A. Tuzhilin. A belief-driven method for discovering unexpected patterns. In *Proceedings of SIGKDD*, 1998.
24. G. Piatetsky-Shapiro. In *AAAI'91 workshop on Knowledge Discovery in Databases*, 1991.
25. J. P. shaffer. Multiple hypothesis testing: A review. *Annual Review of Psychology*, 1995.
26. N. Tatti. Maximum entropy based significance of itemsets. *Knowledge Information System*, 17(1):57–77, 2008.
27. X. Yan, H. Cheng, J. Han, and D. Xin. Summarizing itemset patterns: a profile-based approach. In *Proceedings of SIGKDD*, 2005.
28. M. J. Zaki. Spade: an efficient algorithm for mining frequent sequences. In *Machine Learning Journal*, pages 31–60, 2001.