A Novel Hybrid Approach for Interestingness Analysis of Classification Rules

Tolga Aydın and Halil Altay Güvenir

Department of Computer Engineering, Bilkent University,
06800 Ankara, Turkey
{atolga, guvenir}@cs.bilkent.edu.tr

Abstract. Data mining is the efficient discovery of patterns in large databases, and classification rules are perhaps the most important type of patterns in data mining applications. However, the number of such classification rules is generally very big that selection of interesting ones among all discovered rules becomes an important task. In this paper, factors related to the interestingness of a rule are investigated and some new factors are proposed. Following this, an interactive rule interestingness-learning algorithm (IRIL) is developed to automatically label the classification rules either as "interesting" or "uninteresting" with limited user participation. In our study, VFP (Voting Feature Projections), a feature projection based incremental classification learning algorithm, is also developed in the framework of IRIL. The concept description learned by the VFP algorithm constitutes a novel hybrid approach for interestingness analysis of classification rules.

1 Introduction

Data mining is the efficient discovery of patterns, as opposed to data itself, in large databases [4]. Patterns in the data can be represented in many different forms, including classification rules, association rules, clusters, sequential patterns, time series, contingency tables, and others [5]. However, the number of discovered patterns is usually very big and the user analyzing the patterns is generally interested in a subset of them. Therefore, selection of interesting patterns is an important research topic.

In this paper, we concentrate on the patterns represented by the classification rules and develop an interactive rule interestingness-learning algorithm (IRIL) to automatically classify these rules as interesting or uninteresting, with limited user participation. In our study, VFP (Voting Feature Projections), a feature projection based incremental classification-learning algorithm, was also developed in the framework of IRIL. Being specific to our concerns, VFP takes the rule interestingness factors as features and is used to learn the rule interestingness concept and to classify the newly learned classification rules. The concept description learned by the VFP algorithm constitutes a novel hybrid approach for interestingness analysis of the classification rules.

Section 2 describes the interestingness issue of patterns. Section 3 is devoted to the knowledge representation used in our study. Section 4 and 5 are related to the training

A. Sakurai et al. (Eds.): JSAI 2003/2004, LNAI 3609, pp. 485–496, 2007.

and classifying phases of the VFP algorithm. IRIL is explained in the following section. Giving the experimental results in Section 7, paper is concluded.

2 Interestingness Issue of Patterns

The interestingness issue has been important ever since the beginning of data mining research [1]. There are many factors contributing to the interestingness of a discovered pattern [1, 2, 3]. Some of them are coverage, confidence, completeness, action ability and unexpectedness. The first three factors are objective, action ability is subjective and unexpectedness is sometimes regarded as subjective [7, 8, 9] and sometimes as objective [10, 11]. Objective interestingness factors can be measured independently of the user and domain knowledge. However, subjective interestingness factors are not user and domain knowledge independent. The measurement of a subjective interestingness factor may vary among users analyzing a particular domain, may vary among different domains that a particular user is analyzing and may vary even for the same user analyzing the same domain at different times.

An objective interestingness measure is constructed by combining a proper subset of the objective interestingness factors in a suitable way. For example, objective interestingness factor x can be multiplied by the square of another objective interestingness factor y to obtain an objective interestingness measure of the form xy^2. It is also possible to use an objective interestingness factor x alone as an objective interestingness measure (e.g. *Confidence*). Discovered patterns having *Confidence* ≥ *threshold* are regarded as "interesting". Although the user determines the threshold, this is regarded as small user intervention and the interestingness measure is still assumed to be an objective one.

The existing subjective interestingness measures in the literature are constructed upon unexpectedness and action ability factors. Assuming the discovered pattern to be a set of rules induced from a domain, the user gives her knowledge about the domain in terms of fuzzy rules [9], general impressions [8] or rule templates [7]. The induced rules are then compared with user's existing domain knowledge to determine subjectively unexpected and/or actionable rules.

Both types of interestingness measures have some drawbacks. A particular objective interestingness measure is not sufficient by itself [9]. They are generally used as a filtering mechanism before applying a subjective measure. On the other hand, subjective measures are sometimes used without prior usage of an objective one. In the case of subjective interestingness measures, user may not be well in expressing her domain knowledge at the beginning of the interestingness analysis. It'd be better to automatically learn this knowledge based on her classification of some presented rules as "interesting" or "uninteresting". Another drawback of a subjective measure is that the induced rules are compared with the domain knowledge that addresses the unexpectedness and/or action ability issues. Interestingness is assumed to depend on these two issues. That is, if a rule is found to be unexpected, it is

automatically regarded as an interesting rule. However, it would be better if we learned a concept description that dealt with the interestingness issue directly and if we benefited from unexpectedness and action ability as two of the factors used to express the concept description. That is, interestingness of a pattern may depend on factors other than unexpectedness and action ability issues.

The idea of a concept description that is automatically determined and directly related with the interestingness issue motivated us to design IRIL algorithm. The concept description learned by the VFP algorithm, which was also developed in this framework, constitutes a novel hybrid approach for interestingness analysis of classification rules.

To ensure that the concept description is directly related to the rule interestingness issue, some existing and newly developed interestingness factors that have the capability to determine the interestingness of rules were used instead of the original attributes of the data set. Current implementation of IRIL does not incorporate unexpectedness and action ability factors, requiring no need for domain knowledge. Although all the interestingness factors are of type objective in the current version of IRIL, the thresholds of the objective factors are learned automatically rather than expressing them manually at the beginning. The values of these thresholds are based upon the user's classification results of some presented rules. So, although in the literature subjectivity is highly related to the domain knowledge, IRIL differs from them. IRIL's subjectivity is not related with the domain knowledge. IRIL makes use of objective factors (actually the current version makes use of only objective factors) but for each such factor, it subjectively learns what ranges of factor values (what thresholds) lead to interesting or uninteresting rule classifications if only that factor is used for classification purposes. That is, IRIL presents a hybrid interestingness measure.

IRIL proceeds interactively. An input rule is labeled if the learned concept descripion can label the rule with high certainty. If the labeling or classification certainty factor is not of sufficient strength, user is asked to classify the rule manually. The user looks at the values of the interestingness factors and labels the rule accordingly. In IRIL, concept description is learned or updated incrementally by using the interestingness labels of the rules that are on demand given either as "interesting" or "uninteresting" by the user.

3 Knowledge Representation

The aim of the study presented in this paper is to label a set of classification rules as interesting or uninteresting. This labeling problem is modeled as a new classification problem and a *rule set* is produced for the given rules, which are previously learned by applying a rule induction algorithm on a data set. Each instance of the rule set is represented by a vector whose components are the interestingness label and the interestingness factor values having the potential to determine the interestingness of the corresponding rule.

The classification rules used in the study are probabilistic and have the following general structure:

If (A_1 *op* *value*$_1$) AND (A_2 *op* *value*$_2$) AND …AND (A_n *op* *value*$_n$) THEN
($Class_1$: *probability*$_1$, $Class_2$: *probability*$_2$,…,$Class_k$: *probability*$_k$)

In the above structure, A_i's are the features, $Class_i$'s are the classes and *op* ∈ {=, ≠, <, ≤, >, ≥}.

The instances corresponding to probabilistic classification rules have either "interesting" or "uninteresting" as the interestingness label, and the interestingness factors shown in Table 1. In this new classification problem, these factors are treated as determining features, and interestingness label is treated as the target feature of the rule set.

Table 1. Features of the rule set

Feature	**Short description and/or formula**
Major Class	$Class_i$ that has the highest probability
Major Class Frequency	Ratio of the instances having $Class_i$ as the class label in the data set
Rule Size	Number of conditions in the antecedent part of the rule
Confidence with respect to Major Class	$\|Antecedent \,\&\, Class_i\| \,/\, \|Antecedent\|$
Coverage	$\|Antecedent\| \,/\, \|N\|$
Completeness with respect to Major Class	$\|Antecedent \,\&\, Class_i\| \,/\, \|Class_i\|$
Number of Classes with Zero Probability	Number of classes having zero probability
Standard Deviation of Class Probabilities	Standard deviation of the class probabilities
Major Class Probability	Maximum probability value
Minor Class Probability	Minimum probability value
Decisive	True if Std.Dev.of Class Probabilities > s_{min}

Each feature carries information about a specific property of the corresponding rule. For example, if we let $Class_i$ to take the highest probability, it then becomes the *Major Class* of that classification rule. If we shorten the representation of any rule as "If *Antecedent* THEN $Class_i$" and assume the data set to consist of *N* instances, we can define *Confidence*, *Coverage* and *Completeness* as in Table 1. Furthermore, a rule is decisive if the standard deviation of the class probabilities is greater than s_{min}, whose definition is given in the following equation:

$$s_{min} = \frac{1}{(Class\ Count - 1)\sqrt{Class\ Count}} \tag{1}$$

If all the classes have equal probability in a rule, then the standard deviation of the probabilities becomes zero and the rule becomes extremely indecisive. This is the worst distribution that can happen. The next worst distribution is obtained if exactly one class has a zero probability, and the remaining classes have equal probability. The standard deviation of the probability values in such a situation is called s_{min}.

4 Training in VFP Algorithm

VFP (Voting Feature Projections) is a feature projection based classification-learning algorithm developed in our study. It is used to learn the rule interestingness concept and to classify the unlabeled rules in the context of modeling rule interestingness problem as a new classification problem.

The training phase of VFP, given in Figure 3, is achieved incrementally. On a nominal feature, concept description is shown as the set of points along with the numbers of instances of each class falling into those points. On the other hand, on a numeric feature, concept description is shown as the normal (gaussian) probability density functions for each possible class. Training can better be explained by looking at the sample data set in Figure 1, and the associated learned concept description in Figure 2.

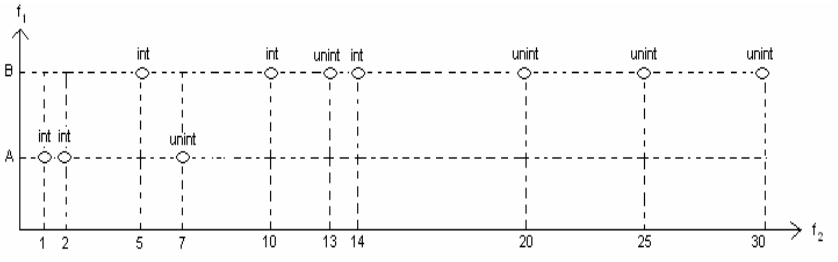

Fig. 1. Sample data set

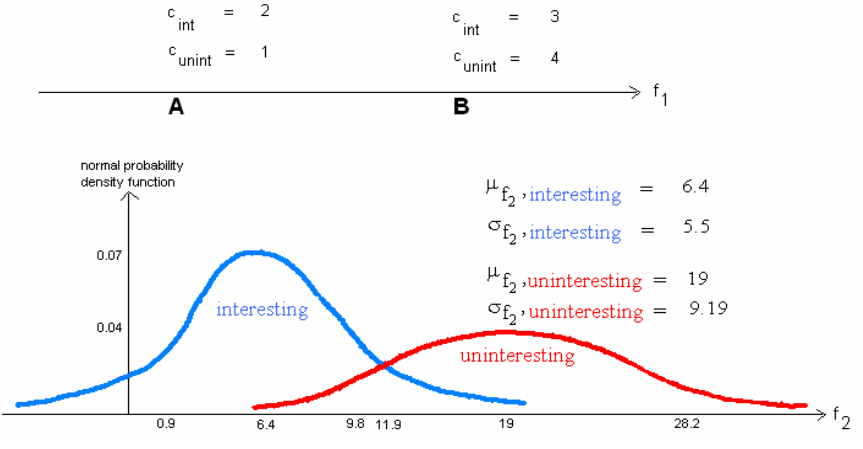

Fig. 2. Concept description learned for the sample data set

The example data set consists of 10 training instances, having nominal f_1 and numeric f_2 features. f_1 takes two values: "A" and "B", whereas f_2 takes some integer values. There are two possible classes: "interesting" and "uninteresting". f_2 is assumed to have gaussian probability density functions for both classes.

```
VFP_train (t)        /* t: newly added training instance */
begin
   let c be the class of t
   let others be the remaining classes other than c
   if training set = {t}
      for each class s
         class_count[s] = 0

   class_count[c]++

   for each feature f
      if f is nominal
         p = find_point(f,t_f)
         if such a p exists
        /* if t_f value exists in the training set */
            point_class_count [f,p,c] ++
         else  /* add new point for f */
            add a new p' point
            point_class_count [f,p',c] = 1
            point_class_count [f,p',others] = 0
      else if f is numeric
        if training set = {t}
           μ_f,c = t_f          ,     μ_f,others = 0
           μ²_f,c = t_f²        ,     μ²_f,others = 0
           σ_f,c = Undefined
           norm_density_func._f,c = Undefined
        else
           n = class_count[c]
           μ_f,c = (μ_f,c * (n-1) + t_f) / n       /*update*/
           μ²_f,c = (μ²_f,c * (n-1) + t_f²) / n    /*update*/
           σ_f,c = sqrt( n/(n-1) (μ²_f,c - (μ_f,c)²) )

           norm_density_func._f,c = 1/(σ_f,c sqrt(2π)) e^( -(x-μ_f,c)² / (2σ²_f,c) )

   return  For numeric features:
           norm_density_func._f,c (∀f, c)
           For nominal features:
           point_class_count[f, p, c] (∀f, p, c)
end.
```

Fig. 3. Incremental training in VFP

In Figure 3 for a nominal feature *f*, *find_point* (*f*, t_f) procedure tries to find (t_f), the new training instance's value at feature *f*, in the *f* projection. If t_f is found at a point *p*, then *point_class_count* [*f*, *p*, *c*] is incremented, assuming that the training instance is of class *c*. If t_f is not found, then a new point *p'* is constructed and *point_class_count* [*f*, *p'*, *class*] is initialized to 1 for *class* = *c*, and to 0 for all other classes. In our study, features used in VFP are the interestingness factor values computed for the classification rules, and we have only "interesting" and "uninteresting" as the classes.

For a numeric feature *f*, if a new training instance *t* of class *c* is examined, we let the previous training instances of class *c* to construct a set *P* and let $\mu_{f,c}$ and $\sigma_{f,c}$ to be the mean and the standard deviation of the *f* feature projection values of the instances in *P*, respectively. The previous training instances' values on *f* need not be stored anywhere, so $\mu_{f,c}$ and $\sigma_{f,c}$ are updated incrementally. Updating $\sigma_{f,c}$ incrementally requires $\mu^2_{f,c}$ to be updated incrementally, as well.

5 Classification in VFP Algorithm

Classification phase of VFP is shown in Figure 4. The query instance is projected on all features and each feature gives votes for each class. If a feature is not ready for classification process, it gives zero, otherwise gives normalized votes. Normalization ensures that each feature has the same weight in classifying the query instances. However, if a feature is not ready, it is not involved in the classification process, therefore need not give normalized votes. For a feature to be ready for the classification process, it should have at least two different values for each class.

The classification starts by giving zero votes to classes on each feature projection. The features that are not ready do not participate in the classification process. The participating features are handled accordingly. For a nominal feature *f*, *find_point* (*f*, q_f) procedure is used to search whether q_f exists in the *f* projection. If q_f is found at a point *p*, feature *f* gives votes for each class as shown in the equation below, and then these votes are normalized to ensure equal voting power among features.

$$feature\_vote\,[f, c] = \frac{point\_class\_count\,[f, p, c]}{class\_count\,[c]} \tag{2}$$

In the above equation, we divide the number of class *c* instances on point *p* of feature projection *f* by the total number of class *c* instances to find the class conditional probability of falling into the *p* point. For a linear feature *f*, each class gets the vote given in equation 3. Normal probability density function values are used as the vote values. These votes are then normalized, too.

$$feature\_vote\,[f, c] = \lim_{\Delta x \to 0} \int_{q_f}^{q_f + \Delta x} \frac{1}{\sigma_{f,c}\sqrt{2\pi}} e^{-\frac{(q_f - \mu_{f,c})^2}{2\sigma_{f,c}^2}} dx \tag{3}$$

VFP$_{query}$(*q*) /* *q*: query instance*/
begin

for each feature *f*
for each class *c*
feature_vote[*f*,*c*] = 0

if *feature_ready_for_query_process*(*f*)

if *f* is nominal
p = *find_point*(*f*,q_f)
if such a *p* exists
/* if q_f value exists in the *training set* */
for each class *c*

$$feature\_vote\ [f,c] = \frac{point\_class\_count\,[f,p,c]}{class\_count\,[c]}$$

normalize_feature_votes (*f*)
/* such that $\sum_c feature\_vote\,[f,c] = 1$ */

else if *f* is numeric
for each class *c*

$$g = \frac{1}{\sigma_{f,c}\sqrt{2\pi}} e^{-\frac{(q_f-\mu_{f,c})^2}{2\sigma_{f,c}^2}}$$

$$feature\_vote\ [f,c] = \lim_{\Delta_x \to 0} \int_{q_f}^{q_f+\Delta x} g\,dx$$

normalize_feature_votes (*f*)

for each class *c*

$$final\_vote\ [c] = \sum_{f=1}^{\#Features} feature\_vote\,[f,c]$$

for each class *c*
if $\min_{i=1}^{\#Classes} final\_vote[i] < final\_vote\ [c] = \max_{i=1}^{\#Classes} final\_vote[i]$
classify *q* as "*c*" with a certainty factor C_f
return C_f
else
C_f = -1
return C_f
end.

Fig. 4. Classification in VFP

Final vote for any class c is the sum of all votes given by the features. If there exists a class c that gets the highest vote and there also exists at least one other class that gets a lower vote than c, then class c is predicted to be the class of the query instance. The certainty factor of the classification (C_f) is computed as follows:

$$C_f = \frac{final\_vote\,[c]}{\sum_{i=1}^{\#Classes} final\_vote\,[i]} \tag{4}$$

If no prediction is made, certainty factor is taken as "–1" to indicate this situation.

6 IRIL Algorithm

IRIL algorithm, shown in Figure 5, needs two input parameters: R (The set of classification rules) and $MinC_t$ (Minimum Certainty Threshold). It tries to classify the rules in R. If $C_f \geq MinC_t$ for a query rule r, this rule is inserted into the successfully classified rules set (R_s). Otherwise, two situations are possible: either the concept description is not able to classify r (C_f = -1), or the concept description's classification (prediction of r's interestingness label) is not of sufficient strength. If $C_f < MinC_t$, rule r is presented, along with its computed eleven interestingness factor values such as *Coverage*, *Rule Size*, *Decisive* etc., to the user for classification. This rule or actually the instance holding the interestingness factor values and the recently

```
IRIL ( R, MinC_t)
begin
   R_t ←∅,    R_s ← ∅
   repeat
      for each rule r ∈ R
         C_f ←VFP_query (r)
            if  C_f < MinC_t
               ask the user to classify r
               set C_f of  this classification to 1
               insert r into R_t
               VFP_train (r)
            else
               add r into R_s
            remove r from R

      for each rule r ∈  R_s
         C_f ←VFP_query (r)
            if  C_f  < MinC_t
               remove r from R_s
               add r into R
   until R is empty
   output rules in R_s
end.
```

Fig. 5. IRIL algorithm

determined interestingness label of this rule is then inserted into the training rule set R_t and the concept description is reconstructed incrementally.

All the rules in R are labeled either automatically by the classification algorithm, or manually by the user. User participation leads rule interestingness learning process to be an interactive one. When the number of instances in the training rule set increases, the concept description learned tends to be more powerful and reliable. When the labeling of the rules ends, the rules in R_s are relabeled by the latest version of the concept description. Because there may exist some rule r that was classified as "interesting" with a sufficient certainty factor by a weak version of the concept description, but now labeled as "interesting" or "uninteresting" with an insufficient certainty factor by the latest and the most reliable version of the concept description. Such rules called as R_{exc} are excluded from R_s and inserted into R. Therefore, we have $R = R_{exc}$ and $R_s = R_s - R_{exc}$. The cycle is repeated until R gets empty and IRIL concludes by presenting the labeled rules in R_s. It is guaranteed that the number of cycles is not infinite and R eventually gets empty. Proof is as follows:

At the end of any cycle, if $R_{exc} = \{\}$ then we are done. If $R_{exc} \neq \{\}$, then at least one rule will be classified by the user and then added into the R_t since the current version of the concept description could not classify the rules in R_{exc} with sufficient certainty. Unless $R_{exc} = \{\}$, at the end of each cycle R_t will expand by at least one element. Therefore, the cycle will be repeated $|R|$ times at most.

7 Experimental Results

IRIL algorithm was tested to classify 184 classification rules induced from a financial distress domain using a benefit maximizing feature projection based rule learner proposed in [6]. The data set of the financial distress domain is a comprehensive set consisting of 25632 data instances and 164 determining features (159 numeric, 5 nominal). There are two classes: "Profit" and "Loss". The data set includes some financial information about 3000 companies collected during 10 years and the class feature states whether the company made a profit or loss in a particular year. Domain expert previously labeled all the 184 induced rules to make accuracy measurement possible. The expert labeled 50 rules (27.17%) as "interesting" and 134 rules (72.83%) as "uninteresting".

The results for $\text{MinC}_t = 60\%$ shows that the user classifies 54 rules with 100% certainty, and 130 rules are classified automatically with $C_f > \text{MinC}_t$. User participation is 29% in the classification process. While labeling the rules, user participation increases in proportion to the $MinC_t$ as expected. In the classification process, it is always desired that rules are generally classified automatically, and user participation is low.

If we look at the accuracy results for $\text{MinC}_t = 60\%$, they are measured as 80%, 94.87% and 73.63% for the rules in R_s (overall accuracy), for the actually interesting rules in R_s (accuracy among interesting rules) and for the actually uninteresting rules in R_s (accuracy among uninteresting rules), respectively. It is important to keep the three accuracy values close to each other. For instance, if the above three accuracy values were 65%, 20% and 75%, respectively, we would easily claim that IRIL made

Table 1. Results for IRIL

	$MinC_t$ 60%	$MinC_t$ 65%	$MinC_t$ 75%
Number of rules	184	184	184
Number of rules classified automatically with high certainty	130	108	90
Number of rules classified by user	54	76	94
User participation	29%	41%	51%
Overall Accuracy	80%	87.04%	90%
Accuracy among interesting rules	94.87%	95.45%	95.12%
Accuracy among uninteresting rules	73.63%	81.25%	85.71%

biased classifications in favor of "uninteresting" class. Because, accuracy among uninteresting rules is too high, whereas accuracy among interesting rules is too low. Furthermore, user herself labels 134 of the rules (72.83%) as uninteresting, so we could label all the rules as "uninteresting" without using IRIL that would result in an accuracy value of 72.83%, which is very close to the overall accuracy of 65%. Fortunately, IRIL makes unbiased classifications since the three accuracy values are balanced. The accuracy values generally increase in proportion to the $MinC_t$. Because the higher the $MinC_t$, is, the higher the user participation is. And higher user participation leads to learning a more powerful and predictive concept description.

8 Conclusion

(IRIL feature projection based interactive rule interestingness learning algorithm) was developed and gave promising experimental results. The concept description learned by the VFP algorithm, also developed in the framework of IRIL, constitutes a novel hybrid approach for interestingness analysis of classification rules. The concept description differs among the users analyzing the same domain. That is, IRIL determines the important rule interestingness factors for a given domain subjectively, by making use of objective factors.

As future work, other classification learning algorithms, which need not be feature projection based, can be used in the framework of IRIL. On the other hand, other objective and subjective interestingness factors, especially unexpectedness, may be used as the features of the rule sets.

References

1. Frawely, W.J., Piatetsky-Shapiro, G., and Matheus, C.J., "Knowledge discovery in databases: an overview" *Knowledge Discovery in Databases*, AAAI/MIT Press, 1991, 1-27
2. Major, J.A., and Mangano, J.J., "Selecting among rules induced from a hurricane database" *Proceedings of AAAI Workshop on Knowledge Discovery in Databases*, 1993, 30-31

3. Piatetsky-Shapiro, G., and Matheus, C.J., "The interestingness of deviations" *Proceedings of AAAI Workshop on Knowledge Discovery in Databases*, 1994, 25-36
4. Fayyad, U., Shapiro, G., and Smyth, P., "From data mining to knowledge discovery in databases" *AI Magazine 17(3)*, 1996, 37-54
5. Hilderman, R.J., and Hamilton, H.J., "Knowledge discovery and interestingness measures: a survey" *Technical Report*, Department of Computer Science, University of Regina, 1999
6. Güvenir, H.A., "Benefit Maximization in Classification on Feature Projections" *Proceedings of the 3rd IASTED International Conference on Artificial Intelligence and Applications* (AIA'03), 2003, 424-429.
7. Klemettinen, M., Mannila, H., Ronkainen, P., Toivonen, H., and Verkamo, A.I., "Finding interesting rules from large sets of discovered association rules" *Proceedings of the 3rd Int. Conf. on Information and Knowledge Management*, 1994, 401-407.
8. Liu, B., Hsu, W., and Chen, S., "Using general impressions to analyze discovered classification rules" *Proceedings of the 3rd Int. Conf. on KDD*, 1997, 31-36.
9. Liu, B., and Hsu, W., "Post-analysis of learned rules", *AAAI*, 1996, 828-834.
10. Hussain, F., Liu, H., Suzuki, E., and Lu, H., "Exception rule mining with a relative interestingness measure" *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2000, 86-97.
11. Dong, G., and Li, J., "Interestingness of discovered association rules in terms of neighborhood-based unexpectedness" *Proceedings of the 2nd Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 1998, 72-86.