Knowledge-Based Systems 22 (2009) 85–99

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Modeling interestingness of streaming association rules as a benefit-maximizing classification problem ☆

Tolga Aydın *, Halil Altay Güvenir

Department of Computer Engineering, Bilkent University, Ankara, Turkey

ARTICLE INFO

Article history:
Received 30 August 2007
Received in revised form 30 June 2008
Accepted 13 July 2008
Available online 19 July 2008

Keywords:
Interestingness learning
Incremental learning
Classification learning
Data mining

ABSTRACT

In a typical application of association rule learning from market basket data, a set of transactions for a fixed period of time is used as input to rule learning algorithms. For example, the well-known Apriori algorithm can be applied to learn a set of association rules from such a transaction set. However, learning association rules from a set of transactions is not a one time only process. For example, a market manager may perform the association rule learning process once every month over the set of transactions collected through the last month. For this reason, we will consider the problem where transaction sets are input to the system as a stream of packages. The sets of transactions may come in varying sizes and in varying periods. Once a set of transactions arrive, the association rule learning algorithm is executed on the last set of transactions, resulting in new association rules. Therefore, the set of association rules learned will accumulate and increase in number over time, making the mining of interesting ones out of this enlarging set of association rules impractical for human experts. We refer to this sequence of rules as "association rule set stream" or "streaming association rules" and the main motivation behind this research is to develop a technique to overcome the interesting rule selection problem. A successful association rule mining system should select and present only the interesting rules to the domain experts. However, definition of interestingness of association rules on a given domain usually differs from one expert to another and also over time for a given expert. This paper proposes a post-processing method to learn a subjective model for the interestingness concept description of the streaming association rules. The uniqueness of the proposed method is its ability to formulate the interestingness issue of association rules as a benefit-maximizing classification problem and obtain a different interestingness model for each user. In this new classification scheme, the determining features are the selective objective interestingness factors related to the interestingness of the association rules, and the target feature is the interestingness label of those rules. The proposed method works incrementally and employs user interactivity at a certain level. It is evaluated on a real market dataset. The results show that the model can successfully select the interesting ones.

1. Introduction

Data mining is the efficient discovery of patterns, as opposed to data itself, in large databases [8]. 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 [19]. In many domains, there is a continuous flow of data and therefore, learned patterns. This causes the number of patterns to be so huge that selection of the useful or interesting ones becomes difficult. In this paper, we deal with the interestingness issue of association rules discovered in domains from which information in the form of transactions is gathered at different time intervals. In a typical application of association rule learning from market basket data, a set of transactions for a fixed period of time is used as input to rule learning algorithms. For example, the well-known Apriori algorithm can be applied to learn a set of association rules from such a transaction set. However, learning association rules from a set of transactions is not a one time only process. For example, a market manager may perform the association rule learning process once every month over the set of transactions collected through the last month. For this reason, we will consider the problem where transaction sets are input to the system as a stream of packages. The sets of transactions may come in varying sizes and in varying periods. Once a set of transactions arrive, the association rule learning algorithm is executed on the last set of transactions, resulting in new association rules. Therefore, the set of association rules learned will accumulate and increase in number over time, making the mining of

☆ The authors gratefully acknowledge the TUBITAK (Scientific and Technical Research Council of Turkey) for providing funds to support this project under Grants 101E044 and 105E065.

* Corresponding author.
E-mail address: atolga@cs.bilkent.edu.tr (T. Aydın).

0950-7051/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.knosys.2008.07.003

interesting ones out of this enlarging set of association rules impractical for human experts. We refer to this sequence of rules as "streaming association rules" and the main motivation behind this research is to develop a technique to overcome the interesting rule selection problem.

The interestingness issue has been an important problem ever since the beginning of data mining research [9]. There are many factors contributing to the interestingness of a discovered pattern [9,33,43]. Coverage, confidence and strength belong to the family of objective interestingness factors. Actionability, related to the benefit we acquire by using the discovered pattern, unexpectedness and novelty are either regarded as subjective [26,30–32,40,44] or objective [1,4,7,10,11,21]. An objective interestingness factor can be measured independently of the domain and the user, while a subjective one is domain or user dependent.

An objective interestingness measure is generally constructed by employing a proper subset of the objective interestingness factors in a formula representation. For example, objective interestingness factor x can be multiplied by the square of another objective interestingness factor y to obtain an objective interestingness measure xy^2. Objective interestingness factors can also be used as an objective interestingness measure alone (e.g., *confidence*) [35,46]. Discovered patterns having interestingness value greater than the threshold are regarded as "interesting". Although the user determines the threshold, this is regarded as a small user intervention and the interestingness measure is still assumed to be an objective one. The objective measures need not always be formulated. For example, the work presented in [50] does not directly formulate a measure; however, it discovers interesting association rules by a clustering method objectively.

The existing subjective interestingness measures in the literature are generally constructed upon unexpectedness and actionability factors. Assuming the discovered pattern to be a set of rules induced from a domain, the user supplies his/her knowledge about the domain in terms of fuzzy rules [30] or general impressions [31,32,44]. The induced rules are then compared with user's existing domain knowledge to determine subjectively unexpected and/or actionable rules. The user may also present what he/she finds interesting or uninteresting as rule templates [26] and filter the induced rules according to these templates to discover the interesting ones. This is actually a query-based approach.

The interestingness measures can be employed during [28,36,42] or after [1,4,7,10,11,21,26,30–32,40,44] the data mining process. Employing those measures during the data mining process has the advantage of processing a small amount of data in the beginning. However, since we do not have the whole set of rules yet, some objective measures requiring the whole set cannot be computed (e.g., *confidence*). This is not a problem for post-processing systems. But, post-processing methods have the disadvantage of requiring more computing power to process large set of rules. Considering the increased computing power of today's computers, the disadvantage of post-processing is not a burden. Consequently, in this paper, we are concerned with post-processing of the induced patterns.

Both types of interestingness measures have some drawbacks. A particular objective interestingness measure is not sufficient by itself [30]. It may not be suitable on some domains. Authors in [22] investigate this issue and discover clusters of measures existing in a data set. An objective measure is generally used as a filtering mechanism before applying a subjective measure. In the case of subjective interestingness measures, user may not be competent in expressing his/her domain knowledge at the beginning of the interestingness analysis. Another drawback of a subjective measure is that the induced rules are compared against the domain knowledge that addresses the unexpectedness and/or actionability issues. Interestingness is assumed to depend only on these two factors. That is, if a rule is found to be unexpected, it is automatically regarded as interesting.

It would be better to view unexpectedness and actionability as two of the interestingness factors and to develop a system that takes a set of interestingness factors into account to learn the interestingness concept automatically with limited user interaction. The interaction can be realized by asking the user to classify some of the rules as "interesting" or "uninteresting". It is also apparent that the definition of interestingness on a given domain usually differs from one expert to another and also over time for a given expert. Therefore, we propose a post-processing method to learn a subjective model for the interestingness concept description of the streaming association rules. The uniqueness of the proposed method is its ability to formulate the interestingness issue of association rules as a benefit-maximizing classification problem and obtain a different interestingness model for each user. In this new classification scheme, the determining features are the selective objective interestingness factors related to the interestingness of the association rules, and the target feature is the interestingness label of those rules. The proposed method, called as "Benefit-Maximizing Interactive Rule Interestingness Learning" (*BM_IRIL*) algorithm, works incrementally and employs user interactivity at a certain level. It models the interestingness of association rules as a benefit-maximizing classification problem. Each rule is represented by an instance and a vector composed of a set of determining features and a target feature represents each instance. The target feature (class feature) takes the values of "interesting" or "uninteresting", and these values are initially unknown for each rule. The determining features consist of a set of objective interestingness factors. They play a key role in determining the target feature value.

BM_IRIL, whose schematic form is shown in Fig. 1, aims to achieve a specified level of accuracy of interestingness classification with a minimum number of queries. It takes the association rule set stream and the certainty threshold value ($MinC_v$) as the input parameters. Each association rule set is induced on the transaction set of the particular period by means of an association rule learning algorithm, such as Apriori. The output of the *BM_IRIL* system is the association rules classified with sufficient certainty at each period. The user can easily filter the rules classified as interesting among the outputted rules. The classification process continues as long as the transaction set stream is supplied to the system.

BM_IRIL employs a core classification algorithm inside. A new feature type is needed to represent the unexpectedness and actionability interestingness factors as determining features. Consequently, we also designed a suitable classifier, namely "Benefit-Maximizing Classifier by Voting Feature Projections" (*BMCVFP*). It is a feature projections based, incremental classification algorithm.

In our classification system, the rules induced at a particular period are regarded as query instances. If an association rule cannot be classified by the core classifier with sufficient certainty, we consult the user, who is generally the expert of the domain, about the interestingness label of the rule. The expert analyzes the objective interestingness factor values and the rule's content together to decide on the interestingness label. Once the expert labels this rule, it is regarded as a training instance and the interestingness concept description (interestingness model) is updated incrementally.

We proposed to model interestingness of patterns as a classification problem in [2]. To the best of our knowledge, none of the existing approaches in the literature had tried to model interestingness as a classification problem. The FPRC (Feature Projection Based Rule Classification) algorithm in [2] used a non-incremental

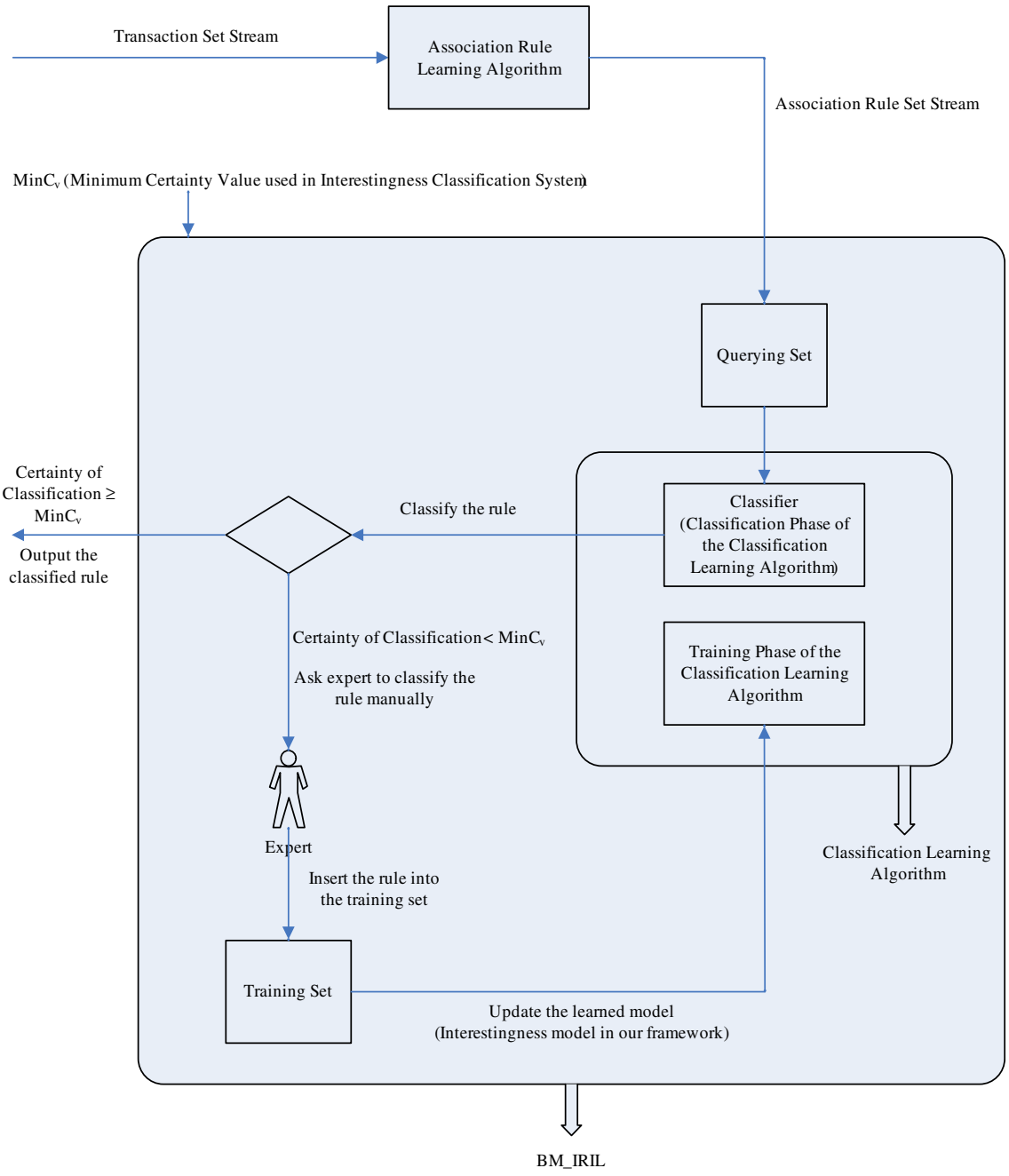

Fig. 1. The BM_IRIL algorithm in schematic form.

classifier. In order to handle the case of streaming rules to be classified, the IRIL algorithm, that used an incremental classifier, has been developed [3]. Both FPRC and IRIL are applicable to learning the interestingness of classification rules, while they are not suitable for association rules. The *BM_IRIL* proposed here is designed for learning the interestingness of association rules. Furthermore, it takes into account the benefit of classifying interestingness rules and subjective interestingness factors such as unexpectedness and actionability are incorporated into the vector representation of query rules. The core classifier is an incremental one as in IRIL.

We assume real human interest to depend both on a selective subset of the objective interestingness factors and the rule's content itself. But, in the literature, they seek to find a correlation between real human interest and objective interestingness measures [5,37–39]. BM_IRIL also proposes a new feature weighting technique that takes benefit maximization issues into account. Feature weights are dynamically updated upon arrival of each training instance. These contributions of the proposed interestingness concept learning system make *BM_IRIL* a novel approach in the literature. It specializes to learn the interestingness concept description. The learned interestingness concept description differs among the consulted users or domain experts.

The paper is organized as follows. Section 2 is devoted to modeling interestingness as a classification problem. Section 3 reviews the feature projections concept. Section 4 explains the basic concepts for benefit-maximizing classification by voting feature segments. Sections 5 and 6 are devoted to the training and the classification phases of the *BMCVFP* classifier, respectively. Section 7 investigates the *BM_IRIL* algorithm comprehensively. Presenting the experimental results in Section 8, we conclude.

2. Modeling interestingness as a classification problem

Interestingness of association rules is the central theme of our study. We first give some preliminaries on association rules. Let $I=\{item_1, item_2, \ldots, item_n\}$ be a set of items. Let S be a set of transactions, where each transaction $T \in I$. An association rule R is an implication of the form $A \rightarrow B$, where $A \subseteq I$, $B \subseteq I$ and $A \cap B = \phi$, satisfying predefined *support* and *confidence* thresholds. Association rule induction is a powerful method for so-called *market basket analysis*, which aims at finding regularities in the shopping behavior of customers of supermarkets, mail-order companies and the like. In an association rule of the form R: $A \rightarrow B$, A is called the antecedent or body of the rule; B is called the consequent or head of the rule.

In this study, we think of a domain from which transactions and association rules induced from these transactions are gathered at varying periods. Christian Borgelt's implementation of Apriori rule induction algorithm [20] is used to induce these association rules at each period. For each period p, the number of such rules is so huge that only a small percentage of them are really interesting for the end user, and most of them are actually uninteresting. It may be thought that the user can reduce the rules learned by changing the parameters of the rule-learning algorithm. However, this will miss many interesting rules. The user is not interested in small number of rules, but he is interested in interesting ones. For instance, while using the Apriori algorithm, support and confidence parameters can be set properly to satisfy some requirements. However, there are other objective and subjective factors related to the interestingness issue of association rules in addition to the support and confidence parameters.

The labeling of the association rules either as interesting or uninteresting can be modeled as a new classification problem where the target concept is the interestingness of the rules. In this new classification problem, each association rule R is seen as a query instance whose target feature value (which is either interesting or uninteresting) is unknown and whose determining features are the interestingness factors having the potential to determine the interestingness of R. There are so many objective interestingness factors influencing the interestingness of association rules, including *support*, *confidence*, *coverage*, *strength* and *size* of the rule. In the literature, some of them are also used as objective interestingness measures [35,46]. For instance, *support* and *confidence* can alone be used as objective interestingness measures [35,46].

We use *confidence*, *coverage*, *strength* and *size* of the rules among the determining features in modeling the interestingness of association rules as a classification problem. Each feature carries information about a specific property of the corresponding association rule. These are accuracy, applicability, independency and simplicity properties of the association rules, respectively. The computation of these features is given in Table 1, where N is the total number of transactions gathered at the current period and $m(X)$ is the number of transactions containing or matching the set of items $X \in I$. We avoid using *support* in our study to ensure all objective determining features to be independent of each other ($support = confidence * coverage$).

Table 1
Linear features and formulas

Linear feature	Short description or formula
Confidence	$\frac{m(A \cap B)}{m(A)}$
Coverage	$\frac{m(A)}{N}$
Strength	$\frac{m(A \cap B) * N}{m(A) * m(B)}$
Size	$\lvert A \rvert + \lvert B \rvert$

In addition to these objective interestingness factors, the rule itself is obviously very important to decide whether it is interesting or not, from the point of view of the user. Therefore, we construct three new determining features for the association rule R, namely *left-hand side* (*antecedent*), *right-hand side* (*consequent*) and *both sides* features of R. Although *confidence*, *coverage*, *strength* and *size* features are linear valued features, the new three features are not. We need to define a new feature type:

Definition 1. A feature f of a type corresponding to an ordered pair of sets is a feature whose values are of the form ($set1$, $set2$) where ($set1$, $set2$) $\neq$ ($set2$, $set1$).

For an association rule of the form R: $A \rightarrow B$, *left-hand side* feature value is (A, $\emptyset$), *right-hand side feature* value is ($\emptyset$, B) and *both sides* feature value is (A, B). These three features of a type corresponding to an ordered pair of sets are essential. Because, they constitute the actionability and unexpectedness interestingness factors in our framework. Users may be interested in the items occurring either on the antecedent or on the consequent part of the association rules; or they may want to see the association rule as a whole while deciding about the interestingness label. A particular user may see a rule actionable if the antecedent or consequent part includes some items that he/she is interested in. For example, in the market basket analysis framework, the user may want to see which items are also sold with the items that he/she is interested in. In such a case, the association rules including the interested items in the antecedent part are regarded as actionable and therefore interesting from the point of view of the user. Actionability is related to the benefit that the user acquires by using the induced association rule. The user may also see a rule interesting if the relationship between the antecedent and the consequent parts of that rule is surprising (unexpected) to him/her. *Left-hand side* and *right-hand side* features handle the actionability whereas *both-sides* feature handles the unexpectedness interestingness factor. Therefore, we do not simply represent the association rule R: $A \rightarrow B$ with two sets A and B instead of three ordered pairs of sets. These three new features are also objective since there is nothing from the domain or user here.

As a consequence, the query instance for the association rule R: $A \rightarrow B$ consists of seven determining features, four of which are of type linear and three of which are of a type corresponding to an ordered pair of sets. The query instance is represented by a vector $\langle confidence_R, coverage_R, strength_R, size_R, (A, \phi), (\phi, B), (A, B), ? \rangle$, where "?" means that the interestingness label will be determined. The interestingness of R depends on all those depending features.

3. Feature projections concept

Feature projections based classifiers are applicable to concepts where each feature, independent of other features, can be used to classify the concept. They project the training instances on each feature separately, and then generalize on these projections to form intervals [6,13–18,45]. In those studies, segments (intervals) are taken to be the basic unit of concept representation; and the classification knowledge is represented in the form of segments formed on each feature. The classification of an unseen instance is based on a majority voting done among individual predictions of features. All those classifiers construct a set of point segments on each nominal feature and a set of segments on each linear feature. A point segment represents a single feature value, whereas a segment represents a set of consecutive feature values.

A feature projections based classifier is a form of ensemble of weak classifiers, such as decision stumps in AdaBoost [25]. It partitions each feature into a set of segments and each segment distributes its vote among all possible classes. On the other hand, a decision stump in AdaBoost, votes only for a single class [25].

Feature projections based classification approach has been extended by other researchers. Pateritsas and Stafylopatis proposed a methodology that merges feature projections based approach with the Naïve Bayesian classifier, which also assumes the features are independent [41]. Note that votes are summed in feature projections approach, while probabilities are multiplied in Naïve Bayesian classifier. Naïve Bayesian is based on the estimation of the posterior probability of a data pattern to belong to a specified class by calculating the probabilities for each feature value of the input pattern [41]. Valev proposed the parallelized version of feature projections based approach, which processes each feature in parallel [48]. Ko and Seo applied feature projections to the text categorization problem [27].

Definition 2. A segment I on a feature f is represented by the following vector:

$$I = \langle lbv, ubv, N_1, N_2, \ldots, Ns, V_1, V_2, \ldots, Vs \rangle$$

where lbv and the ubv are the lower and upper bound values of the segment I, s is the number of classes in domain, N_c is the number of training instances of class c in the segment I and V_c is the vote of the segment I for class c.

In the work presented in [17,18,45], a segment represents examples from a single class, whereas the authors in [6,14,15] allow a segment to represent examples from a set of classes instead of a single class. We prefer to define the segment term to be a unit of concept description that represents examples from a set of classes.

Definition 3. A point segment I on a feature f is a segment such that $lbv = ubv$.

The existing feature projections based classifiers can be trained incrementally. However, they do not preserve order-independency [49]. That is, any change in the order of training instances leads to a different trained model on segments. Those classifiers preserve order-independency only for point segments. This fact motivated us to construct only point segments on the linear features. In the case of features of a type corresponding to an ordered pair of sets, each ordered pair ($set1$, $set2$) is assumed to be a point segment where $lbv = (set1, set2)$ and $ubv = lbv$.

On a feature of a type corresponding to an ordered pair of sets, the number of feature values is limited, so it is possible to save each observed feature value (each observed ordered pair of sets) as a point segment and also possible to compute the class distribution of the training instances on each point segment.

On a linear feature, the number of feature values is not limited as in the case of nominal features and features of a type corresponding to an ordered pair of sets. Their number may range from $-\infty$ to $+\infty$. So, it is not suitable to save each observed feature value as a point segment and to remember the class distribution of the training instances falling into this point segment. To remedy this problem, we propose using a Gaussian probability density function ($gpdf$) for each class on linear feature projections. We assumed that the linear feature projections of the training data exhibit a Gaussian (normal) probability distribution for each class and obtained satisfactory experimental results in our previous studies [2,3]. Therefore, each $x \in \Re$ is regarded as a point segment and the number of such point segments on a linear feature projection is therefore infinite.

For all $x \in \Re$ on a linear feature f, N_c, the number of training instances of class c in point segment x of feature f, is

$$N_c = classcount[c] * \lim_{\Delta x \to 0} gpdf_{f,c}(x)\Delta x, \quad (1)$$

where $gpdf_{f,c}(x)$ is the Gaussian (normal) probability density function of the$f$ values of training instances of class c, and $classcount[c]$ is the number of training instances of class c.

$$gpdf_{f,c}(x) = \frac{1}{\sigma[f,c]\sqrt{2\pi}} e^{-\frac{(x-\mu[f,c])^2}{2(\sigma[f,c])^2}}, \quad (2)$$

where $\mu[f, c]$ and $\sigma[f, c]$ are the mean and the standard deviation of the f values of training instances of class c.

4. Basic concepts for benefit-maximizing classification by voting feature segments

In a normal classification problem, the benefit of correctly classifying an unseen instance is 1 and the benefit of misclassifying an unseen instance is 0. However, in some domains the benefit of correctly classifying an unseen instance differs among the classes. Furthermore, we can obtain even some benefit for misclassifying an unseen instance. In modeling interestingness as a classification problem, the benefit of correctly predicting an interesting rule is much greater than the benefit of correctly predicting an uninteresting rule. Therefore, we employ a benefit-maximizing classification for learning the interestingness classification of the rules. Benefit-maximizing classifiers use a benefit matrix that is supplied externally. Another possibility is to use a cost sensitive approach [47]. Margineantu showed that cost based approaches are equivalent to benefit based approaches if the amount of benefit achieved after classification is not relevant [34]. In our framework, we chose to employ benefit-based model.

Definition 4. A benefit matrix B for a domain with k classes is a $k \times k$ matrix, where $B[i, j]$ is a real-valued number denoting the benefit attained for predicting an instance of class j as i.

In the literature, there are feature projections based, benefit-maximizing classifiers that vote feature segments [12,16,23,24]. However, the classification knowledge in the form of feature segments is obtained after a non-incremental training process. On the other hand, our study employs only point segments resulting in an order-independent incremental training process. Below we give the core definitions related to the benefit concept on feature segments. The definitions are generic and given for segments. However, we use them for point segments in our study.

Definition 5. Given a benefit matrix B, the minimum benefit attainable on a segment $I = \langle lbv, ubv, N_1, N_2, \ldots, N_s, V_1, V_2, \ldots, V_s \rangle$ is given as

$$MinBenefit(I) = \sum_c (N_c * B[\arg \min_i B[i, c], c]).$$

Definition 6. Given a benefit matrix B, the maximum benefit attainable on a segment $I = \langle lbv, ubv, N_1, N_2, \ldots, N_s, V_1, V_2, \ldots, V_s \rangle$ is given as

$$MaxBenefit(I) = \sum_c (N_c * B[c, c]).$$

Definition 7. Given a benefit matrix B, the benefit of classifying all instances of a segment $I = \langle lbv, ubv, N_1, N_2, \ldots, N_s, V_1, V_2, \ldots, V_s \rangle$ as class k is given as

$$SegmentBenefit(I, k) = \sum_c (N_c * B[k, c]).$$

Benefit-maximizing, feature projections based classifiers employ different types of voting methods [23]. We borrow and use the following voting method for a segment I

Definition 8. Vote of a segment I for the class k is given as

$$SegmentClassVote(I,k) = \frac{SegmentBenefit(I,k) - MinBenefit(I)}{MaxBenefit(I) - MinBenefit(I)}.$$

Although the benefit matrix B is usually supplied externally, we prefer to formulate it as in Eq. 3. This formulation ensures that the smaller the probability of a class is, the more the benefit of correctly classifying that class is.

$$B[i,j] = \begin{cases} 0 & \text{if } i \neq j, \\ \dfrac{1}{prob(j)} = \dfrac{\sum_c classcount[c]}{classcount[j]} & \text{else.} \end{cases} \quad (3)$$

Using Eq. 3, *MinBenefit*, *MaxBenefit*, *SegmentBenefit* and finally *SegmentClassVote* definitions simplify to the following:

$$\begin{aligned} MinBenefit(I) &= \sum_c (N_c * B[\text{argmin}_i B[i,c], c]) \\ &= \sum_c (N_c * 0) = 0, \end{aligned} \quad (4)$$

$$\begin{aligned} MaxBenefit(I) &= \sum_c (N_c * B[c,c]) \\ &= \sum_c \left(N_c * \frac{\sum_i classcount[i]}{classcount[c]} \right), \end{aligned} \quad (5)$$

$$\begin{aligned} SegmentBenefit(I,k) &= \sum_c (N_c * B[k,c]) = N_k * B[k,k] \\ &= N_k * \frac{\sum_i classcount[i]}{classcount[k]}, \end{aligned} \quad (6)$$

$$\begin{aligned} SegmentClassVote(I,k) &= \frac{SegmentBenefit(I,k) - MinBenefit(I)}{MaxBenefit(I) - MinBenefit(I)} \\ &= \frac{\dfrac{N_k}{classcount[k]}}{\sum_c \dfrac{N_c}{classcount[c]}}. \end{aligned} \quad (7)$$

In the simplified *SegmentClassVote* definition, the numerator is the ratio of the number of the training instances of class k falling into segment I, to the number of the training instances of class k. The denominator is the sum of these ratios computed for all classes and is used for vote normalization process.

Using Eq. 1 in Eq. 7, *SegmentClassVote* definition can be rewritten in a generic form for linear features as

$$\begin{aligned} SegmentClassVote(I,k) &= \frac{\left(\dfrac{N_k}{classcount[k]} \right)}{\sum_c \dfrac{N_c}{classcount[c]}} \\ &= \frac{\left(\dfrac{classcount[k] * \lim_{\Delta x \to 0} pdf_{f,k}(x)\Delta x}{classcount[k]} \right)}{\sum_c \dfrac{classcount[c] * \lim_{\Delta x \to 0} pdf_{f,c}(x)\Delta x}{classcount[c]}} \\ &= \frac{\lim_{\Delta x \to 0} pdf_{f,k}(x)\Delta x}{\sum_c (\lim_{\Delta x \to 0} pdf_{f,c}(x)\Delta x)} \\ &= \frac{\lim_{\Delta x \to 0} pdf_{f,k}(x)\Delta x}{\lim_{\Delta x \to 0} \sum_c pdf_{f,c}(x)\Delta x} \\ &= \lim_{\Delta x \to 0} \frac{pdf_{f,k}(x)\Delta x}{\sum_c pdf_{f,c}(x)\Delta x} \\ &= \frac{pdf_{f,k}(x)}{\sum_c pdf_{f,c}(x)}. \end{aligned} \quad (8)$$

5. Training in the BMCVFP Algorithm

There are various types of benefit-maximizing classifiers in the literature [12,16,23,24]. However, they do not preserve order-independency in the training phase and none of them are suitable for datasets including features of a type corresponding to an ordered pair of sets. Therefore, we design a new classifier, namely *BMCVFP*. This classifier is close to the family of the feature projections based benefit-maximizing classifiers using independent features' segment class votes. There are two properties discriminating *BMCVFP* from this family of classifiers. The first property is the ability of *BMCVFP* to work also with the features of a type corresponding to an ordered pair of sets. The second discriminating property is the construction of only point segments for linear features to ensure order-independent incremental training.

The training phase of *BMCVFP* is shown in Fig. 2. On each feature projection, training phase learns point segments and their class votes. The classification knowledge is in the form of point segments. A point segment represents examples from a set of classes.

The training phase is achieved incrementally. Let t having class t_c be the incoming training instance. If it is the first training instance, we perform the initialization tasks at lines 1–2. We keep the number of training instances of class c in $classcount[c]$. Therefore, $classcount[t_c]$ is incremented and benefit matrix is updated by the *UpdateBenefitMatrix* algorithm given in Fig. 3. The rest of the training phase differs according to the type of the features.

For a feature f of a type corresponding to an ordered pair of sets (lines 7–13), we search whether t_f exists as a point segment among the previously saved point segments. If it exists, the number of training instances of class t_c falling into point segment t_f of feature projection f, $segment\_class\_count[f, t_f, t_c]$, is incremented. Otherwise, a new point segment t_f consisting of a single training instance of class t_c is constructed. These tasks can be performed incrementally.

For a linear feature f (lines 14–15), we let $\mu[f, t_c]$ and $\sigma[f, t_c]$ to be the mean and the standard deviation of the feature values of the training instances of class t_c on feature projection f. When a new training instance t is processed, these values and $pdf_{f,t_c}(x)$ are updated incrementally as in Eq. 2.

Finally, the *UpdateSegmentsClassVotes* algorithm given in Fig. 4 uses Eq. 7 or Eq. 8 to update the class votes of point segments on feature projection f.

There is no need to maintain the training instances in the *BMCVFP* algorithm. We need to store $classcount[c]$ for each class c. In addition to this, we need to store $\mu[f, c]$ and $\mu^2[f, c]$ for each class c on a linear feature f. These parameters are used to update $\sigma[f, c]$ and finally $gpdf_{f,c}(x)$ as soon as a new training instance of class c is processed. For a feature f of a type corresponding to an ordered pair of sets, we need to store segments and class distribution of training instances on each segment s, $segment\_class\_count[f, s, c]$.

6. Classification in the BMCVFP Algorithm

The classification phase of *BMCVFP* is shown in Fig. 5. In this phase, the query instance q is projected on each feature dimension f, and each feature calculates its class votes (lines 5–6, and 8). The class c taking the highest vote from feature f is called the favored class of f for q. The class votes of features are summed up among all features to get the aggregate class votes (lines 13 and 17). The class c taking the highest aggregate vote is predicted as the class of q (line 18). For the predicted class c, the certainty value of classification denoted by C_v is taken as the ratio of the aggregate vote of c to the sum of the aggregate votes of all classes (line 22). However, if q cannot be classified, the predicted class and the certainty value of this classification are taken as "−1" (line 20). The classification phase of *BMCVFP* algorithm returns the predicted class c along with an associated certainty value (line 23).

```
Algorithm: BMCVFP_train (t)
Input:
The newly added training instance, t
Output:
Point segments' class votes on each feature
Method:
1)  if t is the first training instance then
2)     for each class c do initialize classcount[c] to 0
3)  let t_c be the class of t, increment classcount[t_c]
4)  UpdateBenefitMatrix(classcount)
5)  for each feature f do
6)     let t_f be the feature value of t on f
7)     if f is a feature of a type corresponding to an ordered pair of sets then
8)       if t_f exists as a point segment then
9)          increment segment_class_count[f, t_f, t_c]
10)      else
11)         add a new point segment t_f
12)         for each class c do initialize segment_class_count[f, t_f, c] to 0
13)         set segment_class_count[f, t_f, t_c] to 1
14)    else // f is a linear feature
15)       update probability distribution function of class t_c on f projection
16)    UpdateSegmentsClassVotes(f)
17)  return point segments' class votes on each feature
```

Fig. 2. The BMCVFP$_{train}$ algorithm.

```
Algorithm: UpdateBenefitMatrix(classcount)
Input:
Array of the class distribution of the training instances so far, classcount
Output:
Benefit matrix, B
Method:
1)  for each class i do
2)     for each class j do
3)        if i = j then
4)           B[i, j] ← 1 / prob(j) = Σ_c classcount[c] / classcount[j]
5)        else B[i, j] = 0
6)  return B
```

Fig. 3. The UpdateBenefitMatrix algorithm.

```
Algorithm: UpdateSegmentsClassVotes(f)
Input:
Feature, f
Output:
Updated class votes of point segments of f
Method:
1)  if f is a feature of a type corresponding to an ordered pair of sets then
2)     for each point segment p on feature projection f do
3)        for each class c do
4)           segment_class_vote[f,p,c] ← segment_class_count[f,p,c] / classcount[c]
                                         / Σ_i segment_class_count[f,p,i] / classcount[i]
5)     return segment_class_vote[f,p,c] (∀p, c)
6)  else // f is a linear feature
7)     for each class c do
8)        segment_class_vote[f,x,c] ← pdf_{f,c}(x) / Σ_i pdf_{f,i}(x)  (∀x∈ℜ) // Generic Computation
9)     return segment_class_vote[f,x,c] (∀c)
```

Fig. 4. The UpdateSegmentsClassVotes algorithm.

Class vote calculation differs among feature types. On a linear feature f, query instance has a value of q_f. This value is a real number and constitutes a point segment on feature projection f. Segment class vote calculation taking benefit maximization into account has been defined for linear features in Eq. 8. The feature class votes of f for the query instance q falling into the point segment q_f is the segment class votes of q_f for q.

Class vote calculation of the features of a type corresponding to an ordered pair of sets is shown in Fig. 6. Query instance has a value of q_f on feature dimension f. However, $q_f = (set1, set2)$. That is, it is not a real number as in the case of linear features. It is an ordered pair consisting of two sets of items. If q_f exists as a point segment in the saved point segments, then *segment_class_vote*[f, q_f, c] is used as the feature class vote of feature f for class c (lines 3–5). If q_f does not exist in the saved point segments, then we first multiply the

```
Algorithm: BMCVFP_query(q)
Input:
The query instance, q
Output:
Predicted class and the certainty value of this prediction, (prediction, C_v)
Method:
1)  for each class c do initialize final_vote[c] to 0
2)  for each feature f do
3)     let q_f be the feature value of q on f
4)     if f is a feature of a type corresponding to an ordered pair of sets then
5)        for each class c do
6)           feature_vote[f,c] ← CalculateOrderedPairofSetsTypeFeatureVote(f,c,q_f)
7)     else
8)        for each class c do feature_vote[f,c] ← segment_class_vote[f,q_f,c]
9)     if certainty factor on the base of each feature parameter is enabled then
10)       if feature_vote[f, arg max_c(feature_vote[f,c])] ≥ MinC_v then
11)          if feature weighting parameter is enabled then
12)             for each class c do feature_vote[f,c] ← feature_vote[f,c] * F_W[f]
13)          for each class c do final_vote[c] ← final_vote[c] + feature_vote[f,c]
14)    else
15)       if feature weighting parameter is enabled then
16)          for each class c do feature_vote[f,c] ← feature_vote[f,c] * F_W[f]
17)       for each class c do final_vote[c] ← final_vote[c] + feature_vote[f,c]
18)  prediction ← arg max_c(final_vote[c])
19)  if final_vote[prediction] = 0 then
20)     C_v ← -1 // All classes had a vote of 0, prediction and its C_v are taken as -1
21)  else
22)     C_v ← final_vote[prediction] / Σ_c final_vote[c]
23)  return (prediction, C_v)
```

Fig. 5. The BMCVFP$_{query}$ algorithm.

```
Algorithm:
CalculateOrderedPairofSetsTypeFeatureVote(f,c,q_f)
Input:
Feature, f; class, c; the query instance q's value
on feature f, q_f
Output:
Vote of feature f for class c for query instance q,
feature_vote[f,c]
Method:
1) for each point segment p on feature f do
2)    sim ← Similarity(q_f,p)
3)    if sim = 1 then
4)     feature_vote[f,c] ←segment_class_vote[f,p,c]
5)     return feature_vote[f,c]
6)    else
7)     feature_vote[f,c] ← feature_vote[f,c] +
            segment_class_vote[f,p,c] * sim
8)     sum_sim ← sum_sim + sim
10)feature_vote[f,c] ← feature_vote[f,c] / sum_sim
11)return feature_vote[f,c]
```

Fig. 6. The CalculateOrderedPairofSetsTypeFeatureVote algorithm.

similarity values between q_f and the saved point segments (ordered pairs of sets) by the segment class votes and sum them up. Finally, we normalize the sum to get the feature class vote.

Definition 9. Given two sets A and B, the similarity between these two sets is defined as

$$Set\_similarity(A,B) = \begin{cases} 1 & \text{if } A = B = \emptyset, \\ \min\left(\frac{|A\cap B|}{|A|}, \frac{|A\cap B|}{|B|}\right) & \text{else.} \end{cases}$$

Definition 10. Given two ordered pairs of sets $op_1 = (set1, set2)$ and $op_2 = (set3, set4)$, the similarity between these two ordered pairs is defined as

$$Similarity(op_1, op_2) = Set\_similarity(set1, set3) * Set\_similarity(set2, set4).$$

A small example showing the computation of similarity between two ordered pairs of sets is as follows:

Let $R1 : item1, item2, item3, item4 \rightarrow item6, item7, item9$, $R2 : item2, item4, item6, item10, item11 \rightarrow item1, item9$ be two association rules induced in a domain.

The *both-sides* feature (one of the determining features of the interestingness concept in our framework) values corresponding to these rules will be

$$V1 = (\{item1, item2, item3, item4\}, \{item6, item7, item9\}),$$
$$V2 = (\{item2, item4, item6, item10, item11\}, \{item1, item9\}).$$

$V1$ and $V2$ are two ordered pairs of sets. The similarity between these two values is computed as the multiplication of left- and right-hand side set similarities.

Letting $set1$ ($set2$) be the left- (right-) hand side set of $V1$ and $set3$ ($set4$) be the left- (right-) hand side set of $V2$:

$$set1 \cap set3 = \{item2, item4\} \quad \text{and} \quad set2 \cap set4 = \{item9\},$$
$$Set\_similarity(set1, set3) = \min\left(\frac{2}{4}, \frac{2}{5}\right) = \frac{2}{5},$$
$$Set\_similarity(set2, set4) = \min\left(\frac{1}{3}, \frac{1}{2}\right) = \frac{1}{3}.$$

Finally, the similarity between the ordered pairs $V1$ and $V2$ is

$$Similarity(V1, V2) = \frac{2}{5}\frac{1}{3} = 0.13.$$

Furthermore, the *left-hand side* feature (another of the determining features of the interestingness concept in our framework) values corresponding to these rules are

$V3 = (\{item1, item2, item3, item4\}, \emptyset)$,
$V4 = (\{item2, item4, item6, item10, item11\}, \emptyset)$.

Letting $set1$ ($set2$) be the left- (right-) hand side set of $V3$ and $set3$ ($set4$) be the left- (right-) hand side set of $V4$:

$$set1 \cap set3 = \{item2, item4\} \quad \text{and} \quad set2 \cap set4 = \emptyset,$$

$$Set\_similarity(set1, set3) = \min\left(\frac{2}{4}, \frac{2}{5}\right) = \frac{2}{5},$$

$$Set\_similarity(set2, set4) = 1.$$

Finally, the similarity between the ordered pairs $V3$ and $V4$ is

$$Similarity(V3, V4) = \frac{2}{5}1 = 0.4.$$

The classification phase of *BMCVFP* employs a *certainty factor on the base of each feature* parameter. If we enable this parameter, features whose favorite class takes a vote less than the *minimum certainty value* ($MinC_v$) threshold are not allowed to take place in the voting process. Their class votes are simply taken as zero in the voting process. In our experiments, we choose the threshold as 70%. Other values are also possible; however, we achieve better experimental results for this value.

We also employ a *feature weighting* parameter. If this parameter is enabled, the features multiply their class votes by their weights kept in $F\_W[f]$ (lines 12 and 16). Consequently, some features become more effective in the voting process. Feature weights are not supplied externally to the algorithm. They are computed dynamically. The details of this computation are explained in Section 7.

7. BM_IRIL Algorithm

BM_IRIL is a benefit-maximizing, interactive and incremental rule interestingness learning algorithm. It models the interestingness of association rules as a benefit-maximizing classification problem. Its benefit-maximizing and incremental learning properties are due to the core classifier *BMCVFP* used inside. *BM_IRIL* is interactive since it employs user participation when it is incapable of determining the interestingness label of an input association rule.

In situations where unlabeled data is abundant but labeling data is expensive, the learning algorithm can actively query the user/teacher for labels. In the literature, this type of supervised learning is called active learning [29]. In this respect, BM_IRIL approach can also be considered as an active learning approach.

BM_IRIL algorithm is shown in Fig. 7. In a particular period p, it takes the input parameters $MinC_v$ and R_p (association rules induced from the transactions gathered at period p) to execute. It regards each input association rule of the form R: $A \rightarrow B$ as a query instance and represents the query instance by a vector $\langle confidence_R, coverage_R, strength_R, size_R, (A, \emptyset), (\emptyset, B), (A, B), ?\rangle$. The target feature value "?" indicates that the interestingness label is initially unknown. The determining features of R take a role in deciding the interestingness label of R. *Confidence*, *coverage*, *strength* and *size* features of the rules are linear-valued objective interestingness factors. Each one carries information about a specific property of the corresponding association rule. These are accuracy, applicability, independency, and simplicity properties of the association rules, respectively. The remaining three features are directly related to the R's structure. *Left-hand side* and *right-hand side* features handle the actionability, whereas *both-sides* feature handles the unexpectedness interestingness factor. The way that they handle actionability and unexpectedness was explained in Section 2. Therefore, we do not simply represent the association rule R: $A \rightarrow B$ with two sets A and B instead of three ordered pairs of sets. These three new features are also objective since there is nothing from the domain or user here.

When *BM_IRIL* needs user participation to label a query rule (in fact, query instance), the user is expected to take all these interestingness factors into account.

In our framework, transaction sets come as a stream of packages. The sets of transactions may come in varying sizes and in varying periods. Once a set of transactions arrive, the association rule learning algorithm is executed on the last set of transactions, resulting in new association rules. Therefore, the set of association rules learned will accumulate and increase in number over time. We refer to this sequence of rules as "streaming association rules". *BM_IRIL* is run on each set of induced association rules, where each set belongs to a particular period. There are usually so many association rules induced in a particular period, most of which are obviously uninteresting.

We call R_{up}, R_{sp} and R_t the set of rules classified by user at period p, the set of rules classified by *BM_IRIL* with sufficient certainty at period p, and the set of training rules so far (the set of rules classified by user so far), respectively. At a particular period, each rule r is classified by the querying phase of the core classifier *BMCVFP*. If *certainty value* (C_v) of the classification is greater than or equal to the *minimum certainty value* ($MinC_v$), r is assumed to be classified with sufficient certainty and inserted into R_{sp}. Otherwise, we ask the user to classify r manually and insert r into R_{up}.

We make use of *instant concept update* and *feature weighting* parameters and have the flexibility to enable or disable them in the course of execution of the *BM_IRIL* algorithm. Both parameters are enabled by default unless indicated otherwise.

If *instant concept update* parameter is enabled, a query rule r classified manually by the user is inserted into R_{up} and R_t at the same time. Inserting the query rule r into R_t makes this rule a training rule, anymore. The interestingness model is incrementally updated upon each insertion of an association rule r into R_t. Therefore, each user classification results in an immediate update in the interestingness model. On the other hand, if this parameter is disabled, the rules classified manually by the user are inserted only into R_{up}, but not into R_t for the time being. However, after all the association rules of the period are classified either manually by the user or automatically by *BM_IRIL* with sufficient certainty, the rules in R_{up} are inserted into R_t one by one and the interestingness model is updated after each insertion into the R_t.

If the *feature weighting* parameter is enabled, feature weights can dynamically be updated each time an association rule r is inserted into R_t and the interestingness model is updated. Fig. 8 shows how to update the weight of a feature f. Eq. 9 constitutes the heart of the *UpdateFeatureWeight* algorithm.

$$F\_W[f] = \frac{\sum_c corr\_pred\_tr\_cnt[f, c] * B[c, c]}{\sum_c classcount[c] * B[c, c]}. \tag{9}$$

In Eq. 9, $B[c, c]$ is the benefit of classifying an instance of class c correctly, $classcount[c]$ is the number of training instances (or training rules in our framework) of class c so far and $corr\_pred\_tr\_cnt[f, c]$ is the number of training instances of class c that have been correctly classified by the trained interestingness model on feature projection f with $C_v \geqslant MinC_v$ so far.

The sets of association rules may come in varying periods and *BM_IRIL* is run on each set of induced association rules belonging to a particular period. At a particular period, *BM_IRIL* concludes by presenting the rules predicted as interesting in R_{sp}.

The idea to develop an algorithm like *BM_IRIL* was as follows: (1) to classify most of the input association rules automatically with sufficient certainty and to keep the user participation low,

```
Algorithm: BM_IRIL(R_p, MinC_v)
Input:
Rules induced at period p, R_p; minimum certainty value, MinC_v
Output:
Set of rules classified as interesting with sufficient certainty
Method:
1)  R_up ← ∅ // Set of rules classified by user at period p
2)  R_sp ← ∅ // Set of rules classified with sufficient certainty at period p
3)  for each rule r ∈ R_p do
4)     (prediction, C_v) ← BMCVFP_query(r)
5)     if C_v ≥ MinC_v then insert r into R_sp
6)     else
7)        ask the user to classify r and set C_v of this classification to 100%
8)        insert r into R_up
9)        if instant concept update parameter is enabled then
10)           insert r into R_t and BMCVFP_train(r)
11)           if feature weighting parameter is enabled then
12)              for each feature f do F_W[f] ← UpdateFeatureWeight(f, r)
13)  if instant concept update parameter is disabled then
14)     for each rule r ∈ R_up do
15)        insert r into R_t and BMCVFP_train(r)
16)        if feature weighting parameter is enabled then
17)           for each feature f do F_W[f] ← UpdateFeatureWeight(f, r)
18) return the rules classified as interesting among R_sp
```

Fig. 7. The BM_IRIL algorithm.

```
Algorithm: UpdateFeatureWeight(f, t)
Input:
Feature, f; training instance, t
Output:
Updated weight of f, F_W[f]
Method:
1)  let t_f be the feature value of t on f
2)  Call the lines 4-8 of Fig. 5 for t_f to obtain
    the class votes of f
3)  prediction ← arg max_c (feature_vote[f, c])
4)  if feature_vote[f, prediction] ≥ MinC_v and the
    prediction is correct then
5)     corr_pred_tr_cnt[f, prediction]++
6)  update the feature weight as in Eq.9
7)  return F_W[f]
```

Fig. 8. The UpdateFeatureWeight algorithm.

(2) to keep the benefit accuracy of the classifications high. Experimental results in Section 8 show that we achieve these goals.

8. Experimental results

In our experiments we used transactions recorded by a supermarket for 25 weeks. We decided to take each week as a period and used Christian Borgelt's implementation of Apriori rule induction algorithm [20] to induce association rules from transactions of each period. The example data set used has the common characteristics of market basket datasets. Therefore, we used this representative real world data set.

Table 2 gives the classification distribution statistics of the association rules between the domain expert and the *BM_IRIL* system for the minimum certainty value of 70%. Columns 3 and 4 of Table 2 give the interesting and uninteresting rule counts for each period. This is possible, because we presented each association rule along with its objective interestingness factor values (*confidence*, *coverage*, *strength* and *size* properties of the rule) to the user, who was also a domain expert, to mark its interestingness label. This lengthy and difficult process was necessary to measure the *Benefit Accuracy* values of *BM_IRIL* algorithm at each period. *Benefit accuracy* at a period p is computed as follows:

$$B\_Acc_p = \frac{\sum_c corr\_pred\_cnt[p,c] * B[c,c]}{\sum_c pred\_cnt[p,c] * B[c,c]}. \quad (10)$$

At each period p, all the induced association rules are regarded as query rules and are tried to be classified by *BMCVFP*. In Eq. 10, $B[c, c]$ is the benefit of classifying an instance of class c correctly, $pred\_cnt[p, c]$ is the number of query instances of class c (or query rules in our framework) at period p and $corr\_pred\_cnt[p, c]$ is the number of query instances of class c at period p that have been correctly classified by *BMCVFP* with $C_v \geqslant MinC_v$.

BM_IRIL attempted to classify a total of 1263 association rules, presented along with objective interestingness factor values, with sufficient certainty. Results in Table 2 show that most of the rules are classified automatically by *BM_IRIL*, and user participation to the classification process is low.

The success of the proposed interestingness classification system depends both on the high benefit accuracy values and the low user participation percentages. Because, it is possible to make the user classify most of the rules and have high benefit accuracy on the remaining small number of rules. Also, it is possible to make the user classify a few rules but have low benefit accuracy on the remaining huge number of rules. Neither of these two scenarios is desirable. Consequently, a new success criterion, namely *Performance*, is defined to combine the two success criteria.

$$Performance = B.Acc * (1 - UserParticipation), \quad (11)$$

where, user participation is the proportion of examples in the period that the user has labeled. Table 3 shows the three success criterion values attained in the experiments. Recall values among interesting and uninteresting rules are also given to show that the proposed interestingness classification system does not work in favor of an interestingness class.

Experimental results illustrate that *BM_IRIL* achieves high benefit accuracies while preserving user participation or interaction at low percentages. At each period p, *BM_IRIL* concludes by presenting the rules predicted as interesting in R_{sp}. In this paper, *Benefit accuracy* and *Performance* criterion were defined to have an intuition about the validity of the developed *BM_IRIL* system. It is normally unfeasible to compute these criteria values. Because, hundreds even thousands of association rules can be induced and no domain expert becomes willing to classify each rule by brute force. Even if the number of rules is small, user should not be expected to label each rule one by one. Otherwise, there would not be a need for a system modeling of the interestingness concept. The user should

Table 2
Classification distribution statistics of rules between user and the BM_IRIL system at $MinC_v = 70\%$

Period number	Number of rules	Number of interesting rules	Number of uninteresting rules	Number of interesting rules classified by user	Number of interesting rules classified by BM_IRIL	Number of uninteresting rules classified by user	Number of uninteresting rules classified by BM_IRIL
1	68	2	66	2	0	26	40
2	27	2	25	0	2	0	25
3	58	8	50	2	6	1	49
4	78	16	62	0	16	2	60
5	16	2	14	0	2	1	13
6	170	22	148	4	18	4	144
7	41	4	37	0	4	0	37
8	54	3	51	0	3	0	51
9	41	2	39	0	2	0	39
10	32	3	29	1	2	1	28
11	48	2	46	0	2	1	45
12	21	1	20	0	1	0	20
13	74	2	72	1	1	0	72
14	24	6	18	1	5	1	17
15	176	9	167	1	8	8	159
16	19	2	17	0	2	0	17
17	34	0	34	0	0	0	34
18	40	3	37	0	3	0	37
19	36	8	28	0	8	0	28
20	20	2	18	1	1	0	18
21	5	0	5	0	0	0	5
22	49	20	29	2	18	2	27
23	60	10	50	1	9	4	46
24	39	5	34	1	4	1	33
25	33	3	30	1	2	2	28
Total	1263	137	1126	18	119	54	1072

Table 3
User participation, recall, benefit accuracy and performance values at $MinC_v = 70\%$

Period number	User participation (%)	Recall among interesting rules (%)	Recall among uninteresting rules (%)	Benefit accuracy (%)	Performance (%)
1	41.18	0.00	60.61	43.48	25.58
2	0.00	100.00	100.00	100.00	100.00
3	5.17	75.00	98.00	86.06	81.61
4	2.56	100.00	88.71	96.07	93.60
5	6.25	100.00	92.86	96.55	90.52
6	4.71	81.82	95.27	90.06	85.82
7	0.00	100.00	100.00	100.00	100.00
8	0.00	100.00	100.00	100.00	100.00
9	0.00	100.00	100.00	100.00	100.00
10	6.25	33.33	93.10	75.96	71.21
11	2.08	100.00	97.83	98.15	96.10
12	0.00	100.00	100.00	100.00	100.00
13	1.35	50.00	98.61	94.19	92.92
14	8.33	83.33	94.44	88.57	81.19
15	5.11	88.89	92.22	91.66	86.97
16	0.00	100.00	94.12	95.92	95.92
17	0.00	–	100.00	100.00	100.00
18	0.00	100.00	94.59	95.85	95.85
19	0.00	100.00	96.43	98.28	98.28
20	5.00	50.00	100.00	86.11	81.81
21	0.00	–	100.00	100.00	100.00
22	8.16	85.00	93.10	87.56	80.42
23	8.33	90.00	90.00	90.00	82.50
24	5.13	80.00	97.06	91.77	87.06
25	9.09	66.67	93.33	87.18	79.25

be consulted for a small percentage of rules. This is what *BM_IRIL* actually achieves. User participation is kept at very low percentages.

Table 4 points out the *Performance* values at several minimum certainty threshold values. The value of $MinC_v$ that maximizes the *Performance* criterion is 70% and this value is used throughout the experiments. We used Friedman test, at $\alpha = 0.05$ significance level, to show the differences were actually significant. Asymp. Sig.=$2.015e^{-18} < 0.05$, implying that the differences are statistically significant.

Furthermore, we used *Naïve Bayesian*, as the core classifier in *BM_IRIL* and compared it against the *BM_IRIL* system employing *BMCVFP* as the core classifier inside. The *Naïve Bayesian* classifier computes the posterior probability values for the classes of the domain. We modified it slightly to proceed in a benefit-maximizing manner. For a two-class domain, using "interesting" and "uninteresting" as the class values, the posterior probability values are multiplied by the benefit matrix entries and then normalized to ensure that the probability values sum to one. The benefit matrix is computed again as in Eq. 3. The comparison results provided in Table 5 show that the *BMCVFP* classifier is better than the classical *Naïve Bayesian* classifier.

In this work, we also defined and analyzed the *feature weighting*, *certainty factor on the base of each feature* and *instant concept update* parameters. We enabled them by default in our experiments. Results in Tables 6–8 prove that disabling any of them degrades the performance of the *BM_IRIL* system, except the *feature weighting* parameter.

We used Wilcoxon Signed Ranks test, at $\alpha = 0.05$ significance level, to show the differences were actually significant in Tables 5–8 for the three comparison criteria. Asymp. Sig. values given in the corresponding tables are all less than 0.05 (except for Table 8), implying that the differences are statistically significant (except for Table 8). Using feature weighting does not lead to significantly better results.

In our statistical analysis of the results, we employed non-parametric test strategy because of non-normality of source data and violations of parametric test assumptions. To compare two related samples, we used Wilcoxon Signed Ranks test at $\alpha = 0.05$ significance level. To compare more than two related samples, we used Friedman test again at $\alpha = 0.05$ significance level.

Table 4
Performance comparison at various minimum certainty values

Period number	Minimum certainty value ($MinC_v$)						
	51%	55%	60%	65%	70%	75%	80%
1	45.92%	44.12%	39.12%	28.82%	25.58%	24.05%	21.19%
2	43.86%	42.37%	38.46%	77.51%	100.00%	76.14%	69.96%
3	28.09%	26.88%	23.81%	83.24%	81.61%	82.62%	79.49%
4	19.50%	18.56%	16.23%	95.32%	93.60%	96.70%	95.15%
5	30.43%	29.17%	25.93%	89.97%	90.52%	90.09%	100.00%
6	29.60%	28.35%	25.17%	84.85%	85.82%	87.11%	90.18%
7	36.63%	35.24%	31.62%	100.00%	100.00%	100.00%	81.80%
8	51.52%	50.00%	45.95%	96.53%	100.00%	100.00%	100.00%
9	54.93%	53.42%	49.37%	100.00%	100.00%	95.52%	100.00%
10	37.66%	36.25%	32.58%	74.01%	71.21%	70.88%	70.11%
11	58.97%	57.50%	53.49%	100.00%	96.10%	96.11%	96.07%
12	55.56%	54.05%	50.00%	100.00%	100.00%	100.00%	100.00%
13	69.23%	67.92%	64.29%	93.88%	92.92%	91.43%	89.35%
14	15.79%	15.00%	13.04%	81.83%	81.19%	87.28%	100.00%
15	53.70%	52.19%	48.13%	93.18%	86.97%	87.97%	85.68%
16	34.69%	33.33%	29.82%	95.62%	95.92%	95.92%	95.86%
17	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
18	43.53%	42.05%	38.14%	95.63%	95.85%	76.89%	84.13%
19	17.95%	17.07%	14.89%	98.05%	98.28%	95.51%	98.19%
20	36.00%	34.62%	31.03%	83.99%	81.81%	82.10%	82.58%
21	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	64.00%
22	8.31%	7.86%	6.76%	77.89%	80.42%	83.57%	72.72%
23	23.81%	22.73%	20.00%	86.77%	82.50%	80.31%	89.54%
24	29.82%	28.57%	25.37%	95.35%	87.06%	80.00%	82.93%
25	38.46%	37.04%	33.33%	71.92%	79.25%	85.04%	80.08%
Average	42.56%	41.37%	38.26%	88.17%	88.26%	86.61%	85.16%

Friedman test employed for significance test. Asymp. Sig. = $2.015e^{-18}$ at $\alpha = 0.05$.

Table 5
Comparison of standard BM_IRIL against BM_IRIL using Naïve Bayesian as the core classifier at $MinC_v = 70\%$

Period number	Comparison criterion					
	Benefit accuracy		User participation		Performance	
	Standard BM_IRIL (%)	BM_IRIL with Naïve Bayesian as the core classifier (%)	Standard BM_IRIL	BM_IRIL with Naïve Bayesian as the core classifier (%)	Standard BM_IRIL	BM_IRIL with Naïve Bayesian as the core classifier (%)
1	43.48	0.00	41.18	100.00	25.58	0.00
2	100.00	8.89	0.00	77.78	100.00	1.98
3	86.06	6.72	5.17	84.48	81.61	1.04
4	96.07	5.48	2.56	88.46	93.60	0.63
5	96.55	85.22	6.25	25.00	90.52	63.92
6	90.06	6.66	4.71	88.82	85.82	0.74
7	100.00	30.69	0.00	51.22	100.00	14.97
8	100.00	35.68	0.00	53.70	100.00	16.52
9	100.00	34.62	0.00	56.10	100.00	15.20
10	75.96	45.94	6.25	31.25	71.21	31.59
11	98.15	45.76	2.08	41.67	96.10	26.69
12	100.00	49.04	0.00	38.10	100.00	30.36
13	94.19	45.54	1.35	47.30	92.92	24.00
14	88.57	19.63	8.33	54.17	81.19	9.00
15	91.66	20.76	5.11	76.14	86.97	4.95
16	95.92	28.64	0.00	52.63	95.92	13.57
17	100.00	76.47	0.00	23.53	100.00	58.48
18	95.85	46.18	0.00	30.00	95.85	32.33
19	98.28	24.70	0.00	44.44	98.28	13.72
20	86.11	41.66	5.00	30.00	81.81	29.16
21	100.00	60.00	0.00	40.00	100.00	36.00
22	87.56	9.67	8.16	63.27	80.42	3.55
23	90.00	18.95	8.33	66.67	82.50	6.32
24	91.77	39.84	5.13	48.72	87.06	20.43
25	87.18	46.22	9.09	48.48	79.25	23.81

Wilcoxon test employed for significance test. For benefit accuracy comparison criterion, Asymp. Sig. (2-tailed) = $1.229e^{-5}$ at $\alpha = 0.05$. For user participation comparison criterion, Asymp. Sig. (2-tailed) = $1.228e^{-5}$ at $\alpha = 0.05$. For performance comparison criterion, Asymp. Sig. (2-tailed) = $1.23e^{-5}$ at $\alpha = 0.05$.

In the comparisons, standard BM_IRIL means the version of BM_IRIL that:

1. uses BMCVFP as the core classifier,
2. also employs certainty factor on the base of each feature,
3. employs feature weighting process, and
4. employs instant concept update.

9. Conclusions

In a typical application of association rule learning from market basket data, a set of transactions for a fixed period of time is

Table 6
Comparison of standard BM_IRIL against BM_IRIL that does not use certainty factor on the base of each feature

Period number	Comparison criterion					
	Benefit accuracy		User participation		Performance	
	Standard BM_IRIL (%)	BM_IRIL that does not use certainty factor on the base of each feature (%)	Standard BM_IRIL (%)	BM_IRIL that does not use certainty factor on the base of each feature (%)	Standard BM_IRIL (%)	BM_IRIL that does not use certainty factor on the base of each feature (%)
1	43.48	43.48	41.18	41.18	25.58	25.58
2	100.00	79.53	0.00	3.70	100.00	76.58
3	86.06	81.68	5.17	8.62	81.61	74.64
4	96.07	96.41	2.56	1.28	93.60	95.18
5	96.55	95.83	6.25	6.25	90.52	89.84
6	90.06	87.26	4.71	7.65	85.82	80.59
7	100.00	94.12	0.00	2.44	100.00	91.82
8	100.00	98.33	0.00	1.85	100.00	96.51
9	100.00	100.00	0.00	0.00	100.00	100.00
10	75.96	79.73	6.25	6.25	71.21	74.75
11	98.15	98.07	2.08	2.08	96.10	96.02
12	100.00	100.00	0.00	0.00	100.00	100.00
13	94.19	95.18	1.35	2.70	92.92	92.61
14	88.57	97.13	8.33	4.17	81.19	93.08
15	91.66	90.11	5.11	7.95	86.97	82.94
16	95.92	95.74	0.00	5.26	95.92	90.70
17	100.00	100.00	0.00	0.00	100.00	100.00
18	95.85	88.89	0.00	7.50	95.85	82.22
19	98.28	96.31	0.00	5.56	98.28	90.96
20	86.11	75.31	5.00	10.00	81.81	67.78
21	100.00	100.00	0.00	0.00	100.00	100.00
22	87.56	71.73	8.16	22.45	80.42	55.63
23	90.00	82.51	8.33	15.00	82.50	70.13
24	91.77	83.94	5.13	10.26	87.06	75.33
25	87.18	91.67	9.09	6.06	79.25	86.11

Wilcoxon test employed for significance test. For benefit accuracy comparison criterion, Asymp. Sig. (2-tailed) = 0.03 at $\alpha = 0.05$, For user participation comparison criterion, Asymp. Sig. (2-tailed) = 0.006 at $\alpha = 0.05$. For performance comparison criterion, Asymp. Sig. (2-tailed) = 0.009 at $\alpha = 0.05$.

Table 7
Comparison of standard BM_IRIL against BM_IRIL that does not use instant concept update

Period number	Comparison criterion					
	Benefit accuracy		User participation		Performance	
	Standard BM_IRIL (%)	BM_IRIL that does not use instant concept update (%)	Standard BM_IRIL (%)	BM_IRIL that does not use instant concept update (%)	Standard BM_IRIL (%)	BM_IRIL that does not use instant concept update (%)
1	43.48	0.00	41.18	100.00	25.58	0.00
2	100.00	43.10	0.00	7.41	100.00	39.91
3	86.06	47.30	5.17	12.07	81.61	41.59
4	96.07	88.62	2.56	7.69	93.60	81.80
5	96.55	96.15	6.25	6.25	90.52	90.14
6	90.06	70.98	4.71	20.59	85.82	56.37
7	100.00	82.81	0.00	7.32	100.00	76.75
8	100.00	96.85	0.00	3.70	100.00	93.26
9	100.00	78.98	0.00	9.76	100.00	71.28
10	75.96	85.62	6.25	6.25	71.21	80.27
11	98.15	94.43	2.08	6.25	96.10	88.53
12	100.00	100.00	0.00	0.00	100.00	100.00
13	94.19	93.99	1.35	1.35	92.92	92.72
14	88.57	88.32	8.33	8.33	81.19	80.96
15	91.66	83.37	5.11	10.80	86.97	74.37
16	95.92	91.92	0.00	5.26	95.92	87.08
17	100.00	100.00	0.00	0.00	100.00	100.00
18	95.85	95.89	0.00	0.00	95.85	95.89
19	98.28	67.79	0.00	19.44	98.28	54.61
20	86.11	86.20	5.00	5.00	81.81	81.89
21	100.00	80.00	0.00	20.00	100.00	64.00
22	87.56	74.07	8.16	22.45	80.42	57.44
23	90.00	86.33	8.33	13.33	82.50	74.82
24	91.77	85.61	5.13	10.26	87.06	76.83
25	87.18	94.95	9.09	3.03	79.25	92.07

Wilcoxon test employed for significance test. For benefit accuracy comparison criterion, Asymp. Sig. (2-tailed) = 0.001 at $\alpha = 0.05$, For user participation comparison criterion, Asymp. Sig. (2-tailed) = 0.001 at $\alpha = 0.05$. For performance comparison criterion, Asymp. Sig. (2-tailed) = 0.001 at $\alpha = 0.05$.

used as input to rule learning algorithms. For example, the well-known Apriori algorithm can be applied to learn a set of association rules from such a transaction set. However, learning association rules from a set of transactions is not a one time only process. For example, a market manager may perform the association rule learning process once every month over the set of transactions collected through the last month. For this reason, we considered the problem where transaction sets are input

Table 8
Comparison of standard BM_IRIL against BM_IRIL that does not use feature weighting

Period number	Comparison criterion					
	Benefit accuracy		User participation		Performance	
	Standard BM_IRIL (%)	BM_IRIL that does not use feature weighting (%)	Standard BM_IRIL (%)	BM_IRIL that does not use feature weighting (%)	Standard BM_IRIL (%)	BM_IRIL that does not use feature weighting (%)
1	43.48	50.00	41.18	35.29	25.58	32.35
2	100.00	75.61	0.00	11.11	100.00	67.21
3	86.06	82.78	5.17	12.07	81.61	72.79
4	96.07	96.96	2.56	3.85	93.60	93.23
5	96.55	100.00	6.25	0.00	90.52	100.00
6	90.06	87.81	4.71	7.06	85.82	81.61
7	100.00	87.00	0.00	4.88	100.00	82.76
8	100.00	100.00	0.00	0.00	100.00	100.00
9	100.00	100.00	0.00	0.00	100.00	100.00
10	75.96	97.44	6.25	3.13	71.21	94.39
11	98.15	94.10	2.08	2.08	96.10	92.14
12	100.00	100.00	0.00	0.00	100.00	100.00
13	94.19	94.95	1.35	2.70	92.92	92.39
14	88.57	97.22	8.33	4.17	81.19	93.17
15	91.66	92.72	5.11	5.11	86.97	87.98
16	95.92	95.77	0.00	0.00	95.92	95.77
17	100.00	100.00	0.00	0.00	100.00	100.00
18	95.85	82.01	0.00	5.00	95.85	77.91
19	98.28	98.13	0.00	0.00	98.28	98.13
20	86.11	87.50	5.00	5.00	81.81	83.13
21	100.00	100.00	0.00	0.00	100.00	100.00
22	87.56	78.92	8.16	16.33	80.42	66.04
23	90.00	85.29	8.33	11.67	82.50	75.34
24	91.77	90.78	5.13	5.13	87.06	86.13
25	87.18	86.55	9.09	6.06	79.25	81.31

Wilcoxon test employed for significance test. For benefit accuracy comparison criterion, Asymp. Sig. (2-tailed) = 0.433 at $\alpha = 0.05$. For user participation comparison criterion, Asymp. Sig. (2-tailed) = 0.331 at $\alpha = 0.05$. For performance comparison criterion, Asymp. Sig. (2-tailed) = 0.351 at $\alpha = 0.05$.

to the system as a stream of packages. The sets of transactions may come in varying sizes and in varying periods. Once a set of transactions arrive, the association rule learning algorithm is executed on the last set of transactions, resulting in new association rules. Therefore, the set of association rules learned accumulates and increases in number over time, making the mining of interesting ones out of this enlarging set of association rules impractical for human experts. We referred to this sequence of rules as "association rule set stream" or "streaming association rules" and the main motivation behind this research was to develop a technique to overcome the interesting rule selection problem.

Definition of interestingness of association rules on a given domain usually differs from one expert to another and also over time for a given expert. Therefore, this paper proposed a post-processing method to learn a subjective model for the interestingness concept description of the streaming association rules. The uniqueness of the proposed method is its ability to formulate the interestingness issue of association rules as a benefit-maximizing classification problem and obtain a different interestingness model for each user. In our opinion, it is better to learn user specific interesting rules rather than the generic interesting rules. The same system can be easily used to learn interesting rules for a user with different views or needs. In this new classification scheme, the determining features are the selective objective interestingness factors (including the rule's content itself) related to the interestingness of the association rules, and the target feature is the interestingness label of those rules. The proposed method, *BM_IRIL*, works incrementally and employs user interactivity at a certain level. In fact, *BM_IRIL*, can execute on association rules induced by any association rule-mining algorithm.

BM_IRIL was evaluated on a real market dataset. The results show that the model can successfully select the interesting ones. It may seem that the interestingness values are binary rather than ranks or numeric scores found in many contexts. However, we present each interesting rule along with an associated certainty factor. Therefore, the rules classified as interesting may also be ranked according to their associated certainty factor values. The rule classified as "interesting" with 100% certainty is absolutely the most interesting rule for the user analyzing the domain.

Furthermore, we used *Naïve Bayesian* as the core classifier in *BM_IRIL* and compared it against the *BM_IRIL* system employing *BMCVFP* as the core classifier inside. The *Naïve Bayesian* classifier computes the posterior probability values for the classes of the domain. We modified it slightly to proceed in a benefit-maximizing manner. For a two-class domain, using "interesting" and "uninteresting" as the class values, the posterior probability values are multiplied by the benefit matrix entries and then normalized to ensure that the probability values sum to one. The comparison results show that the *BMCVFP* classifier is better than the classical *Naïve Bayesian* classifier.

In this work, we also defined and analyzed the *feature weighting*, *certainty factor on the base of each feature* and *instant concept update* parameters. We enabled them by default in our experiments. Results proved that disabling any of them degrades the performance of the *BM_IRIL* system, except the *feature weighting* parameter.

The other contributions of the study are as follows: (1) Unexpectedness and actionability interestingness factors of association rules are handled by taking the rule's content into account. For this purpose, a new feature type is defined and a benefit-maximizing core classifier also capable of working with this type of feature is designed. (2) We assume real human interest to depend both on a selective subset of the objective interestingness factors and the rule's content itself.

These contributions of the proposed interestingness concept learning system make *BM_IRIL* a novel approach in the literature. As a future work, novelty interestingness factor may be incorporated into the system.

References

[1] A.S. Al-Hegami, V. Bhatnagar, N. Kumar, Novelty framework for knowledge discovery in databases, in: Y. Kambayashi et al. (Ed.), DaWak 2004, LNCS, vol. 3181, 2004, pp. 48–57.

[2] T. Aydin, H.A. Güvenir, Feature projection based rule classification, in: Proc. of the Twelfth Turkish Symposium on Artificial Intelligence and Neural Networks (TAINN'2003), Canakkale, Turkey, July 2–4, 2003, pp. 652–661.

[3] T. Aydin, H.A. Güvenir, Learning interestingness of streaming classification rules, in: Cevdet Aykanat, Tugrul Dayar, Ibrahim Korpeoglu (Eds.), Proc. of ISCIS 2004, LNCS, vol. 3280, 2004, pp. 62–71.

[4] V. Bhatnagar, A.S. Al-Hegami, N. Kumar, Novelty as a measure of interestingness in knowledge discovery, IJIT 2 (1) (2005).

[5] D.R. Carvalho, A.A. Freitas, N. Ebecken, Evaluating the correlation between objective rule interestingness measures and real human interest, in: A. Jorge et al. (Eds.), Proc. of PKDD 2005, LNAI, vol. 3721, 2005, pp. 453–461.

[6] G. Demiröz, H.A. Güvenir, Classification by voting feature intervals, in: Maarten van Someren, Gerhard Widmer (Eds.), Proc. of the 9th European Conference on Machine Learning, LNAI, vol. 1224, April 23–25, 1997, pp. 85–92.

[7] G. Dong, J. Li, Interestingness of discovered association rules in terms of neighborhood-based unexpectedness, in: Proc. of the 2nd Pacific-Asia Conference on Knowledge Discovery and Data Mining, 1998, pp. 72–86.

[8] U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, From data mining to knowledge discovery in databases, AI Magazine 17 (3) (1996) 37–54.

[9] W.J. Frawely, G. Piatetsky-Shapiro, C.J. Matheus, Knowledge discovery in databases: an overview, Knowledge Discovery in Databases (1991) 1–27.

[10] A.A. Freitas, On objective measures of rule surprisingness, in: Proc. of the Second European Conference on the Principles of Data Mining and Knowledge Discovery (PKDD'98), 1998, pp. 1–9.

[11] A.A. Freitas, On rule interestingness measures, Knowledge-Based Systems 12 (1999) 309–315.

[12] H.A. Güvenir, Benefit maximization in classification on feature projections, in: Proc. of the 3rd IASTED International Conference Artificial Intelligence and Applications (AIA'03), Sept. 8–10, 2003, pp. 424–429.

[13] H.A. Güvenir, S. Altingovde, I. Uysal, E. Erel, Bankruptcy prediction using feature projection based classification, in: Proc. of SCI/ISAS'99, Orlando, FL, July 31 – August 4, 1999, pp. 108–113.

[14] H.A. Güvenir, G. Demiröz, N. Ilter, Learning differential diagnosis of erythemato-squamous diseases using voting feature intervals, Artificial Intelligence in Medicine 13 (3) (1998) 147–165.

[15] H.A. Güvenir, N. Emeksiz, An expert system for the differential diagnosis of erythemato-squamous diseases, Expert Systems With Applications 18 (1) (2000) 43–49.

[16] H.A. Güvenir, N. Emeksiz, N. Ikizler, N. Örmeci, Diagnosis of gastric carcinoma by classification on feature projections, Artificial Intelligence in Medicine 31 (3) (2004) 231–240.

[17] H.A. Güvenir, H.G. Koç, Concept representation with overlapping feature intervals, Cybernetics and Systems: An International Journal 29 (3) (1998) 263–282.

[18] H.A. Güvenir, I. Şirin, Classification by feature partitioning, Machine Learning 23 (1) (1996) 47–67.

[19] R.J. Hilderman, H.J. Hamilton, Knowledge discovery and interestingness measures: a survey, Technical Report, Department of Computer Science, University of Regina, 1999.

[20] Available from: <http://fuzzy.cs.uni-magdeburg.de/~borgelt/apriori.html>.

[21] F. Hussain, H. Liu, E. Suzuki, H. Lu, Exception rule mining with a relative interestingness measure, in: Proc. of Pacific-Asia Conf. on Knowledge Discovery and Data Mining, 2000, pp. 86–97.

[22] X. Huynh, F. Guillet, H. Briad, A data analysis approach for evaluating the behavior of interestingness measures, in: A. Hoffman, H. Motoda, T. Scheffer, Proc. of DS 2005, LNAI, vol. 3735, 2005, pp. 330–337.

[23] N. Ikizler, Benefit maximizing classification using feature intervals, M.Sc Thesis, Department of Computer Engineering, Bilkent University, 2002.

[24] N. Ikizler, H.A. Güvenir, Maximizing benefit of classifications using feature intervals, in: Vasile Palade, Robert J. Howlett, Lakhmi Jain (Eds.), Proc. of the 7th International Conference on Knowledge-Based Intelligent Information & Engineering Systems (KES'2003), Oxford, United Kingdom (Sept. 3–5, 2003), LNAI, vol. 2773, no. 1, 2003, pp. 339–345.

[25] M. Kawakita, M. Minami, S. Eguchi, C.E. Lennert-Cody, An introduction to the predictive technique AdaBoost with a comparison to generalized additive models, Fisheries Research 76 (2005) 328–343.

[26] M. Klemettinen, H. Mannila, P. Ronkainen, H. Toivonen, A.I. Verkamo, Finding interesting rules from large sets of discovered association rules, in: Proc. of the 3rd Int. Conf. on Information and Knowledge Management, 1994, pp. 401–407.

[27] Y. Ko, J. Seo, Text categorization using feature projections, in: Proc. of the 19th Int. Conf. on Computational Linguistics, Taipei, Taiwan, 2002, pp. 1–7.

[28] Y. Lan, D. Janssens, G. Chen, G. Wets, Improving associative classification by incorporating novel interestingness measures, Expert Syst. Appl. 31 (1) (2006) 184–192.

[29] D.D. Lewis, W.A. Gale, A sequential algorithm for training text classifiers, in: Proc. of the 17th annual Int. ACM SIGIR Conf. on Research and development in information retrieval, 1994, pp. 3–12.

[30] B. Liu, W. Hsu, Post-analysis of learned rules, AAAI 1996, pp. 828–834.

[31] B. Liu, W. Hsu, S. Chen, Using general impressions to analyze discovered classification rules, in: Proc. of the 3rd Int. Conf. on KDD, 1997, pp. 31–36.

[32] B. Liu, H. Wynne, S. Chen, Y. Ma, Analyzing the subjective interestingness of association Rules, IEEE Expert Intelligent Systems & Their Applications 15 (5) (2000). Sep/Oct.

[33] J.A. Major, J.J. Mangano, Selecting among rules induced from a hurricane database, in: Proc. of AAAI Workshop on Knowledge Discovery in Databases, 1993, pp. 30–31.

[34] D.D. Margineantu, Methods for cost-sensitive learning, Ph.D. Dissertation, Oregon State University, 2002.

[35] K. McGarry, A survey of interestingness measures for knowledge discovery, The Knowledge Engineering Review 20 (1) (2005) 39–61.

[36] E. Noda, A.A. Freitas, H.S. Lopes, Discovering interesting prediction rules with a genetic algorithm, CEC-99.

[37] M. Ohsaki, S. Kitaguchi, K. Okamoto, H. Yokoi, T. Yamaguchi, Evaluation of rule interestingness measures with a clinical dataset on hepatitis, in: J.F. Boulicaut et al. (Eds.), PKDD 2004, LNAI, vol. 3202, 2004, pp. 362–373.

[38] M. Ohsaki, S. Kitaguchi, H. Yokoi, T. Yamaguchi, Investigation of rule interestingness in medical data mining, in: S. Tsumoto (Ed.), Proc. of the Am. 2003, vol. 3430, LNAI, 2005, pp. 174–189.

[39] M. Ohsaki, Y. Sato, S. Kitaguchi, H. Yokoi, T. Yamaguchi, Comparison between objective interestingness measures and real human interest medical data mining, in: R. Orchard (Ed.), IEA/AIE 2004, LNAI, vol. 3029, 2004, pp. 1072–1081.

[40] B. Padmanabhan, A. Tuzhilin, Unexpectedness as a measure of interestingness in knowledge discovery, Decision Support Systems 27 (3) (1999) 303–318.

[41] C. Pateritsas, A. Stafylopatis, A nearest features classifier using a self-organizing map for memory base evaluation, in: S. Kollias (Ed.), Proc. of ICANN 2006, LNCS, vol. 4132, 2006, pp. 391–400.

[42] I. Rahal, D. Ren, A. Perera, H. Najadat, W. Perrizo, R. Rahhal, W. Valdivia, Incremental interactive mining of constrained association rules from biological annotation data with nominal features, in: Proc. of the 2005 ACM Symposium on Applied Computing, 2005, pp. 123–127.

[43] G.P. Shapiro, C.J. Matheus, The interestingness of deviations, in: Proc. of AAAI Workshop on Knowledge Discovery in Databases, 1994, pp. 25–36.

[44] B. Shekar, R. Natarajan, A framework for evaluating knowledge-based interestingness of association rules, Fuzzy Optimization and Decision Making 3 (2004) 157–185.

[45] I. Sirin, H.A. Güvenir, Empirical evaluation of the CFP algorithm, in: C. Rowles, H. Liu, N. Foo (Eds.), Proc. of the Sixth Australian Joint Conference on Artificial Intelligence, World Scientific, Melbourne, Australia, 1993, pp. 311–315.

[46] P.N. Tan, V. Kumar, Interestingness measures for association patterns: a perspective, Technical Report TR00-036, KDD 2000 Workshop on Post-processing in Machine Learning and Data Mining.

[47] P. Turney, Cost-sensitive classification: empirical evaluation of a hybrid genetic decision tree induction algorithm, Journal of Artificial Intelligence Research 2 (1995) 369–409.

[48] V. Valev, Supervised pattern recognition by parallel feature partitioning, Pattern Recognition 37 (2004) 463–467.

[49] H. Watanabe, M. Arai, K. Okuda, Batch mode algorithms of classification by feature partitioning, IEICE Transactions on Information & Systems E81-D (1998) 1.

[50] Y. Zhao, C. Zhang, S. Zhang, Discovering interesting association rules by clustering, in: G.I. Webb, Xinghuo Yu (Eds.), Proc. of AI 2004, LNAI, vol. 3339, 2004, pp. 1055–1061.