MODELING INTERESTINGNESS OF STREAMING ASSOCIATION RULES AS A BENEFIT MAXIMIZING CLASSIFICATION PROBLEM

A DISSERTATION SUBMITTED TO
THE DEPARTMENT OF COMPUTER ENGINEERING
AND THE INSTITUTE OF ENGINEERING AND SCIENCE
OF BILKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

By
Tolga Aydın
January, 2009

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of doctor of philosophy.

Prof. Dr. Halil Altay Güvenir(Advisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of doctor of philosophy.

Prof. Dr. Özgür Ulusoy

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of doctor of philosophy.

Prof. Dr. İhsan Sabuncuoğlu

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of doctor of philosophy.

Assoc. Prof. Dr. Ferda Nur Alpaslan

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of doctor of philosophy.

Asst. Prof. Dr. Çiğdem Gündüz Demir

Approved for the Institute of Engineering and Science:

Prof. Dr. Mehmet B. Baray
Director of the Institute

ABSTRACT

Modeling Interestingness of Streaming Association Rules as a Benefit Maximizing Classification Problem

Tolga Aydın
PhD. in Computer Engineering
Supervisor: Prof. Dr. Halil Altay Güvenir
January, 2009

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 previous 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 arrives, the association rule learning algorithm is run on the last set of transactions, resulting in a new set of 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 the other and also over time for a given expert. In this thesis, 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, 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 works incrementally and employs user interactivity at a certain level. It is evaluated on a real supermarket dataset. The results show that the model can successfully select the interesting ones.

Keywords: Interestingness learning, incremental learning, classification learning, association rules, data mining.

ÖZET

Akan İlişkisel Kuralların İlginçliğini Fayda Maksimizasyonu Tabanlı bir Sınıflandırma Problemi Olarak Modelleme

Tolga Aydın
Bilgisayar Mühendisliği, Doktora
Tez Yöneticisi: Prof. Dr. Halil Altay Güvenir
Ocak, 2009

Market sepet verisinden ilişkisel kural öğrenme gibi tipik bir uygulamada, sabit bir zaman dilimi için toplanan işlemler kümesi kural öğrenme algoritmalarına girdi olarak kullanılır. Örneğin, yaygın olarak bilinen Apriori algoritması böyle bir işlem kümesinden ilişkisel kural kümesi öğrenmek üzere uygulanabilir. Ancak, işlemler kümesinden ilişkisel kurallar öğrenme işlemi bir kerelik bir işlem değildir. Örneğin, herhangi bir market yöneticisi her ay bir kez, son bir ay süresince toplanan işlemler kümesi üzerinde ilişkisel kural öğrenme işlemini gerçekleştirebilir. Bu nedenden dolayı, işlem kümelerinin sisteme akan paketler şeklinde girdi olduğu bir problemi ele alacağız. İşlemler kümeleri değişiklik gösteren büyüklükte ve zaman dilimlerinde sisteme gelebilir. Herhangi bir işlemler kümesi sisteme vardığında, ilişkisel kural öğrenme algoritması bu son işlemler kümesi üzerinde çalıştırılarak yeni ilişkisel kurallar öğrenilir. Bu yüzden, öğrenilen ilişkisel kurallar kümesi zaman içinde gitgide büyümekte ve bunların içinden ilginç olanlarının elde edilmesi uzmanlar için pratik bir işlem olmaktan çıkmaktadır. Bu kurallar dizisinden "ilişkisel kural kümesi akımı" veya "akan ilişkisel kurallar" olarak bahsedebiliriz ve bu araştırmamızın ardındaki ana motivasyon, ilginç kural seçme probleminin üstesinden gelebilecek bir teknik geliştirmektir. Başarılı bir ilişkisel kural

madenciliği sistemi ilginç kuralları seçerek konunun uzmanlarına sunabilmelidir. Ancak, belli bir alanda ilişkisel kuralların ilginçliğinin tanımı uzmandan uzmana ve hatta aynı uzman için zaman içinde farklılık gösterebilir. Bu tezde, akan ilişkisel kuralların ilginçlik konsepti tanımı için kişisel bir model öğrenmek üzere sonradan-işlemli bir metod önermekteyiz. Önerilen metodun özgünlüğü ilişkisel kuralların ilginçlik kavramını fayda maksimizasyonu tabanlı bir sınıflandırma problemi olarak formüle edebilme ve her bir kullanıcı için farklı bir ilginçlik modeli elde edebilme yeteneğidir. Bu yeni sınıflandırma planında, belirleyici öznitelikler ilişkisel kuralların ilginçliği ile alakalı seçici nesnel ilginçlik faktörleridir ve hedef öznitelik bahsi geçen kuralların ilginçlik etiketinden oluşmaktadır. Önerilen metod artımlı bir şekilde çalışarak belli bir seviyede kullanıcı etkileşimi içermektedir. Metod gerçek bir süpermarket veri kümesi üzerinde değerlendirilmekte ve sonuçlar modelin ilginç kuralları başarılı bir biçimde seçebildiğini göstermektedir.

Anahtar sözcükler: İlginçlik öğrenimi, artımlı öğrenim, sınıflandırma öğrenimi, ilişkisel kurallar, veri madenciliği.

Aileme...

Acknowledgement

I would like to express my gratitude to Prof. Dr. H. Altay Güvenir, from whom I have learned a lot, due to his supervision, suggestions, and support during this research.

I am grateful to the members of my thesis committee, Prof. Dr. Özgür Ulusoy, Prof. Dr. İhsan Sabuncuoğlu, Assoc. Prof. Dr. Ferda Nur Alpaslan, Asst. Prof. Dr. Çiğdem Gündüz Demir for accepting to read and review this thesis and for their valuable comments.

I would like to thank to my family for their morale support and for many things.

This thesis was supported, in part, by TUBITAK (Scientific and Technical Research Council of Turkey) under Grant Grants 101E044 and 105E065.

Contents

List of Figures

List of Tables

List of Symbols and Abbreviations

AQ15 : An incremental learner of attribute-based descriptions from examples

B : Benefit Matrix

$BenefitAccuracy_p$: Benefit Accuracy at period p

BM_IRIL : Benefit-Maximizing, Interactive Rule Interaction Learning algorithm

BMCVFP : Benefit-Maximizing Classifier by Voting Feature Projections

c : Class

C4.5/C5.0 : An industrial-quality descendant of ID3

DBSCAN : Density-Based Spatial Clustering of Applications with Noise

FCLS : Flexible Concept Learning System

gpdf : Gaussian Probability Distribution Function

I : Item Set

ID3 : Iterative Dichotomiser 3 algorithm

KDD : Knowledge Discovery in Databases

KMEANS : Partitioning clustering algorithm

$MinC_v$: Minimum Certainty Value Threshold

$m(X)$: Number of transactions containing or matching the set of items $X \in I$

N : Number of transactions

p : Period

r : Rule

R : Rule

R_p : Set of rules induced at period p

R_{sp} : Set of rules classified with sufficient certainty by *BMCVFP* at period p

R_t : Set of training rules

R_{up} : Set of rules classified by the user at period p

t : Training instance

T : Transaction

μ : Mean

σ : Standard deviation

Chapter 1

Introduction

Data mining is the process of efficient discovery of patterns, as opposed to data itself, in large databases [16]. It encompasses many different techniques and algorithms. They differ in the kinds of data that can be analyzed and the kinds of knowledge representation used to convey the discovered knowledge. 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 [28]. 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.

Selecting the interesting patterns among the induced ones is important because only a few of the total induced patterns are likely to be of any interest to the domain expert analyzing the data. The remaining patterns are either irrelevant or obvious, and do not provide any new knowledge.

To increase the utility, relevance, and usefulness of the discovered patterns, techniques are required to reduce the number of patterns. Some good metrics that measure the interestingness of a pattern are needed.

There are two aspects of pattern interestingness, objective and subjective aspects and this fact is what makes finding interesting patterns a challenging issue. Objective measures rate the patterns based on some statistics computed from the observed data. They are domain-independent and require minimal

user participation. On the other hand, subjective measures are user-driven; they are based on user's belief in data, e.g., unexpectedness, novelty, and actionability.

Both types of interestingness measures have some drawbacks. A particular objective interestingness measure is not sufficient by itself [41]. It may not be suitable on some domains. Authors in [33] 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, a 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 patterns 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 pattern 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 of the induced patterns automatically with limited user interaction. The interaction can be realized by asking the user to classify some of the patterns as "interesting" or "uninteresting".

Association rules are among the important pattern types and employed today in many application areas including web usage mining, intrusion detection, filtering, screening, and bioinformatics.

Let us 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 $T \subseteq I$. An association rule R is an implication of the form $A \rightarrow B$, where $A \subseteq I$, $B \subseteq I$ and $A \cap B = \emptyset$, 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.

A common application domain of association rules is sales data, known as basket data. 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 learning 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 previous month. For this reason, it is worthwhile to consider the problem where transaction sets are input to rule learning algorithms as a stream of packages.

In this thesis, 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.

The sets of transactions may come in varying sizes and in varying periods. That is, the number of transactions in a particular period may be quite different from those of the other periods. For example, if we think of the purchasing trends of customers, sales generally increase towards the end of the year and decrease in case of an economical crisis. Furthermore, it may not be possible to receive the transactions regularly. For example, if we think of a shopping center, there will not be any sales while the center is out of service, possibly for some constructional works.

Once a set of transactions arrives, the association rule learning algorithm is run on the last set of transactions, resulting in a new set of 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 "streaming association rules" and the main motivation behind this research is to develop a technique to overcome the interesting rule selection problem.

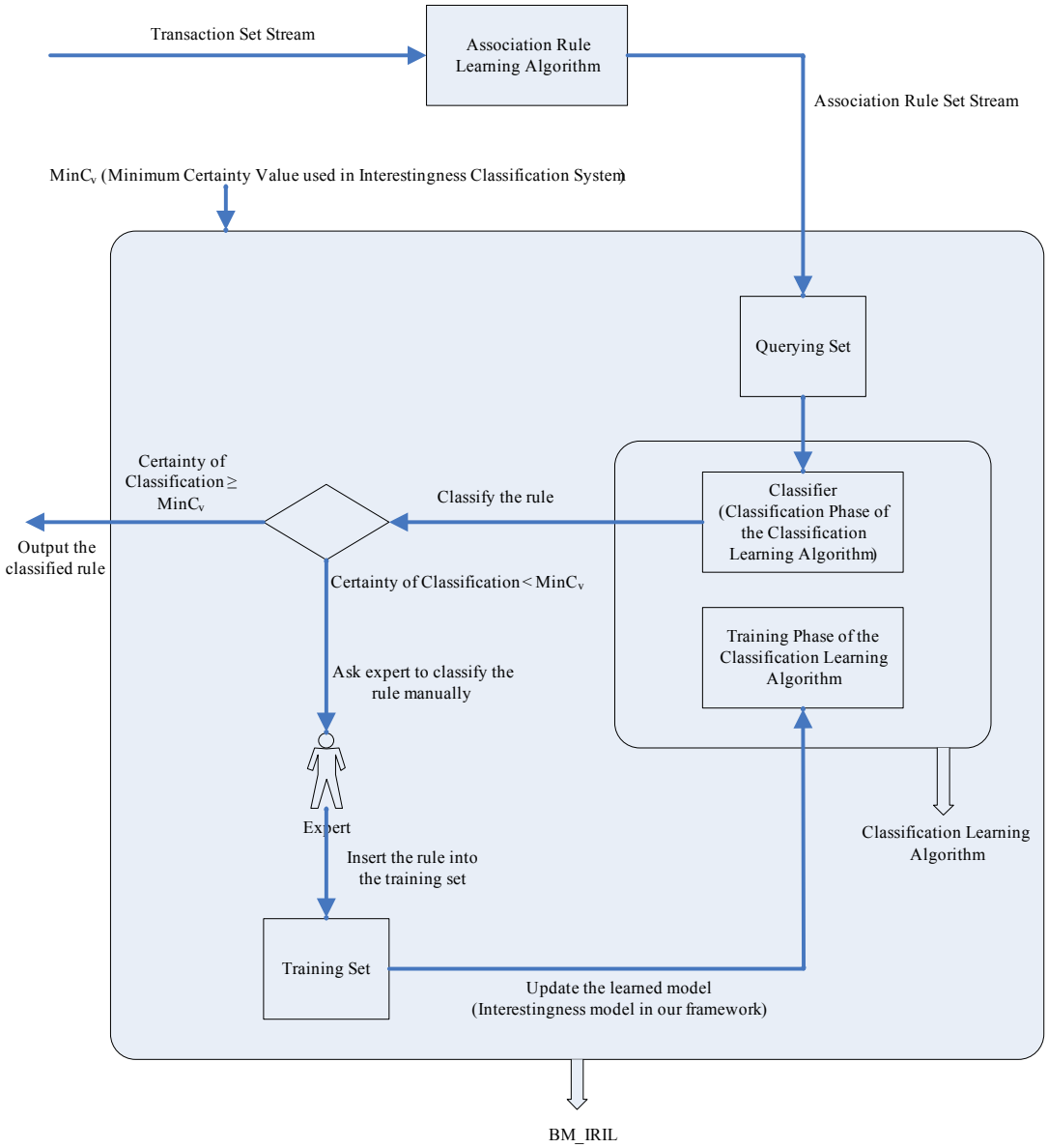

Figure 1.1: The BM_IRIL Algorithm in schematic form

The definition of interestingness on a given domain usually differs from one expert to another and also over time for a given expert. The proposed system, "Benefit-Maximizing Interactive Rule Interestingness Learning" *(BM_IRIL)* algorithm, learns a subjective model for the interestingness concept description of the induced rules. The interaction with the user ensures this subjectivity. It formulates the interestingness concept of streaming association rules as a benefit-maximizing classification problem and learns a different interestingness model for each user.

BM_IRIL, whose schematic form is shown in Fig.1.1, is a post-processing system that works in an incremental manner, employs a benefit-maximizing classifier inside, tries to classify the incoming rules with sufficient certainty and keeps user interactivity at a certain level.

BM_IRIL was evaluated on a real supermarket dataset. We recorded the customer transactions for 25 weeks. Each week is taken as the unit of period and has its own set of transactions. Following this, we induced association rules from the set of transactions for each period and used them as input to the *BM_IRIL* algorithm.

A transaction set consists of transactions, and a transaction (in the supermarket domain) contains the items bought by a customer at one swoop. An example transaction is:

{milk, egg, detergent, butter, coke, beer}

A transaction set belongs to a particular period, and association rules are learned from each of these sets. An example association rule is:

{tomato, cucumber, potato} $\rightarrow$ {lemon, onion}

Each association rule is regarded as a query instance and tried to be classified by the inside classifier of the *BM_IRIL* algorithm with sufficient certainty. If the classifier is not so certain to say "interesting" or "uninteresting" for the interestingness label of the association rule, user participation is put into use.

The user evaluates the rule not only by looking at its content, but also by

analyzing the objective factor values of this rule computed beforehand. The objective factors used in this dissertation are confidence, coverage, strength, and size. Each selected objective factor carries information about a specific property of the association rules. These are accuracy, applicability, independency, and simplicity properties of the association rules, respectively. The details about the computation of these factors are given in Chapter 4.

Once the user evaluates a rule as "interesting" or "uninteresting", this rule becomes a training instance for the inside classifier of the *BM_IRIL* algorithm, and the interestingness model is updated. The inside classifier proposed in this dissertation is a feature projections based classifier. It employs confidence, coverage, strength, size, and the rule's content itself as the determining features. The learned model is the combination of the local models learned on each feature projection.

An example of a learned interestingness model is:

if the confidence $> 70\% \rightarrow$ rule is interesting

if the coverage $> 50\% \rightarrow$ rule is interesting

if the strength $< 1 \rightarrow$ rule is uninteresting

if the size of the rule $> 6 \rightarrow$ rule is uninteresting

if the antecedent of the rule is similar to {milk, egg} $\rightarrow$ rule is uninteresting (user is not interested in rules containing milk and egg in the body part of the rule)

if consequent of the rule is similar to {beer} $\rightarrow$ rule is interesting (user is interested in rules containing beer in the head part of the rule. He would like to know which items lead to the sales of beer)

if the rule is similar to ({tomato} $\rightarrow$ {cucumber}) $\rightarrow$ rule is uninteresting (user is not interested in rules containing such a trivial implication)

1.1 Contributions

The contributions of this dissertation can be listed as follows:

- modeling the interestingness concept of association rules as a classification problem and proposing an active learning approach (*BM_IRIL*) for this purpose,
- suitability of the *BM_IRIL* approach for domains where enormous amount of transactions arrives in varying time periods,
- learning the interestingness model rather than using a given interestingness model that addresses only unexpectedness and/or actionability aspects of association rules,
- selecting both objective and subjective interestingness factors of association rules as determining features in modeling the interestingness concept as a classification problem,
- handling the unexpectedness and actionability subjective interestingness factors by proposing a new feature type, ordered-pair of sets. This new feature type is quite different from the classical linear and nominal feature types that are already in use by the machine learning community,
- proposing a new classifier suitable to work with ordered-pair of sets type features,
- learning user specific interesting rules rather than the generic interesting rules.

1.2 Outline of the Thesis

In the next chapter, we review the data mining techniques briefly and highlight the interestingness concept defined to overcome the huge number of patterns induced as a result of the mining process. In Chapter 2, we also review the categorization of the interestingness measures and give numerous examples in

the literature. In this thesis, we propose a method that has the ability to formulate the interestingness issue of association rules as a benefit-maximizing classification problem. Therefore, it makes sense to choose an appropriate classifier to be used in the proposed method. Chapter 3 is devoted to the choice of an appropriate benefit-maximizing classifier. Upon selection of an appropriate classifier, modeling interestingness of streaming association rules as a benefit-maximizing classification problem is explained in Chapter 4. Giving the empirical evaluations in Chapter 5, we conclude the thesis with Chapter 6.

Chapter 2

Previous Work

Data mining encompasses many different techniques and algorithms. They differ in the kinds of data that can be analyzed and the kinds of knowledge representation used to convey the discovered knowledge. In this chapter, we review these techniques briefly and highlight the interestingness concept defined to overcome the huge number of patterns induced as a result of the mining process. We also review the categorization of the interestingness measures and give numerous examples in the literature.

2.1 Basic Data Mining Techniques

Knowledge discovery in databases (KDD) refers to the overall process of discovering useful knowledge from data, and data mining refers to a particular step in this process [16]. KDD involves many steps including data preparation, data selection, data cleaning, incorporation of appropriate prior knowledge, data mining and proper interpretation of the results of mining. The data mining step is the efficient discovery of previously unknown, valid, interesting, novel, potentially useful, and understandable patterns in large databases. The knowledge that we seek to discover describes patterns in the data as opposed to knowledge about the data itself. Patterns in the data can be represented in many different forms, including classification rules, association rules, clusters,

sequential patterns, time series, contingency tables, summaries obtained using some hierarchical or taxonomic structure, and others [28].

2.1.1 Classification

Classification is perhaps the most commonly applied data mining technique. Early examples of classification techniques from the literature include Quinlan's ID3 [58], Michalski et al.'s AQ15 [48] and Naive Bayes Classifier [30]. ID3 induces a decision tree. An object is classified by descending the tree until a branch leads to a leaf node containing the decision. AQ15 induces a set of decision rules. An object is classified by selecting the most preferred decision rule according to user-defined criteria. Naive Bayes Classifier's approach to classifying the new instance is to assign the most probable target value, given the predicting attribute values that describe the instance. Later examples of classification techniques from the literature include Zhang and Michalski's FCLS [71], and Quinlan's C4.5/C5.0 [59]. FCLS induces a weighted threshold rule. The threshold determines the number of conditions that must be satisfied in a valid rule. An object is classified by generalizing and specializing examples until the number of incorrectly classified examples is below some user-defined error rate. C4.5/C5.0 is an industrial-quality descendant of ID3 that has seen widespread use in the research community.

2.1.2 Association

Association is another commonly applied data mining technique. The problem is typically examined in the context of discovering purchasing patterns of customers from retail sales transactions, and is commonly referred to as market basket analysis. The association task involves the discovery of knowledge with a user-defined accuracy (confidence factor) and relative frequency (support factor).

Much of the literature focuses on the Apriori algorithm [1] and its descendants containing various refinements and improvements. Apriori extracts the

set of frequent itemsets from the set of candidate itemsets generated. A frequent itemset is an itemset whose support is greater than some user-defined minimum and a candidate itemset is an itemset whose support has yet to be determined. It has an important property that if any subset of a candidate itemset is not a frequent itemset, then the candidate itemset is also not a frequent itemset.

2.1.3 Clustering

Clustering, also a frequently used data mining technique, is the process of identifying objects that share some distinguishing characteristics. There are numerous techniques of clustering in the literature.

A well-known example of clustering in the literature is the k-means algorithm [31]. Given a set of objects X and an integer number k, the k-means algorithm searches for a partition of X into a predefined k number of clusters that minimizes the inter-cluster distance of squared errors and maximizes the intra-cluster similarity measure.

More recent examples from the literature include DBSCAN [15] by Ester et al. DBSCAN is a density-based approach that utilizes user-defined parameters for controlling the density of the discovered clusters. This approach allows adjacent regions of sufficiently high density to be connected to form clusters of arbitrary shape and is able to differentiate noise in regions of low density.

2.1.4 Correlation

Correlation is a statistical measurement of the relationship between two variables. Possible correlations range from "+1" to "-1". A zero correlation indicates that there is no relationship between the variables. A correlation of "-1" indicates a perfect negative correlation, meaning that as one variable goes up, the other goes down. A correlation of "+1" indicates a perfect positive correlation, meaning that both variables move in the same direction together.

Statistically oriented in nature, correlation has also seen increasing use as a data mining technique. Even though the analysis of multi-dimensional categorical data is possible, the most commonly employed method is that of two-dimensional contingency table analysis of categorical data using the chi-square statistic as a measure of significance [67].

2.1.5 Summarization

Summarization involves the discovery of high-generality knowledge, i.e. a discovered rule should cover a large amount of data (which is not the main requirement of other tasks such as classification).

In the summarization task, the goal is to produce some characteristic description of a class. This description is a kind of summary, describing some properties shared by all (or most) tuples belonging to that class. In the case of discovered summaries expressed in the form of rules, the discovered rules can be interpreted in the following way: "If a tuple belongs to the class indicated in the antecedent of the rule, then the tuple has all the properties mentioned in the consequent of the rule". Hilderman and Hamilton give various heuristic measures of interestingness for summarization task [27]. These measures include *Simpson*, *Shannon*, *Theil*, *Atkinson*, etc. A characteristic rule does not aim at discriminating classes, as classification rules do. This stems from the fact that a characteristic rule for a given class is produced by taking into account only tuples belonging to that class. This is in contrast with classification rules, where each rule is produced by taking into account tuples belonging to different classes.

Classification, regression and summarization can be regarded as a form of supervised discovery, since the user specifies the goal attribute (or class attribute) and the system has to discover some relationship between that attribute and the other attributes. However, unsupervised learning tasks can be transformed into supervised ones by the user, if this is convenient (e.g. by specifying the constraint that a given goal item be the only item in the consequent of an association rule).

2.2 Interestingness Concept

Typically, the number of patterns generated by the data mining techniques is very large, but only a few of these patterns are likely to be of any interest to the domain expert analyzing the data. The reason for this is that many of the patterns are either irrelevant or obvious, and do not provide any new knowledge. To increase the utility, relevance, and usefulness of the discovered patterns, techniques are required to reduce the number of patterns that are necessary to be considered. In order to satisfy this goal, some good metrics that measure the interestingness of a pattern are needed.

A pattern is interesting if it is easily understood, unexpected, potentially useful and actionable, novel, or it validates some hypothesis that a user seeks to confirm. There are two aspects of pattern interestingness, objective and subjective aspects and this fact is what makes finding interesting patterns a challenging issue. Objective measures rate the patterns based on some statistics computed from the observed data. They include measures such as support, confidence, chi-square, and correlation. Objective measures are domain-independent and require minimal user participation (aside from specifying the quality measure threshold). Some objective measures are symmetric with respect to permutation of the items while others are not. From an association rule mining perspective, symmetric measures are often used for itemsets whereas asymmetric measures are applied to rules. These measures can be applied during mining or post-processing steps of the knowledge discovery process. On the other hand, subjective measures are user-driven; they are based on user's belief in data, e.g., unexpectedness, novelty, and actionability.

Interesting patterns should be selected among the generated ones. Although there are several studies on this subject, there is not a standard or universally best measure to decide which pattern is interesting or which one is not, yet.

	B	¬B	
A	f_{11}	f_{10}	f_{1+}
¬A	f_{01}	f_{00}	f_{0+}
	f_{+1}	f_{+0}	N

Table 2.1: 2 x 2 contingency table for binary variables

2.2.1 Objective Measures From Statistics

2.2.1.1 Goodness of fit test

This class of methods compare the actual distribution of a data set to its expected distribution under a null hypothesis. To test for item dependence, the null hypothesis assumes that a pattern consists of items that are independent of each other. Then, based on evidence provided by the observed data, one can determine whether to accept or reject the independence assumption. Although this class of methods are good, the techniques that directly estimate the degree of dependence are often more desirable.

To illustrate the test, let A and B denote a pair of binary variables. The data set that contains these variables can be summarized into a 2x2 contingency table as shown in Table 2.1. Each cell represents the four possible combinations of A and B values. f_{ij} corresponds to the frequency (support count) for each cell; while $f_{i+} = f_{i0} + f_{i1}$ and $f_{+j} = f_{0j} + f_{1j}$. Also, N refers to the size of the database.

Pearson's χ^2 statistic is often used for goodness of fit test:

$$\chi^2 = \sum_{j,k} \frac{(f^o_{jk} - f^e_{jk})^2}{f^e_{jk}}$$

This statistic measures the sum of the normalized deviation between the observed support count, f^o_{jk}, of each cell in the contingency table from its expected support count, f^e_{jk}. To test for variable dependencies, it is first hypothesized that the variables are independent of each other. In this case, the expected support count for each contingency table cell entry is $f^e_{jk} = N(\frac{f^o_{j+}}{N})(\frac{f^o_{+k}}{N})$. For

the 2x2 contingency table shown in Table 2.1, its χ^2 value can be simplified into the following expression:

$$\chi^2 = \frac{N(f_{11}f_{00} - f_{01}f_{10})^2}{f_{1+}f_{0+}f_{+1}f_{+0}}$$

If the computed χ^2 value is large, then we have more evidence to reject the independence hypothesis. This begs the question of how large should the χ^2 value be in order to conclude with high confidence that the variables are not independent? By looking up the standard probability tables for χ^2 distribution, a cutoff value can be obtained for which the independence assumption can be rejected at any significance level α.

However, χ^2 does not tell us the strength of correlation between items in an association pattern. Instead, it will only help us to decide whether items in the pattern are independent of each other. Thus, it cannot be used for ranking purposes. Another disadvantage is that the χ^2 statistic depends on the total number of transactions. But, the χ^2 cutoff value depends only on the degrees of freedom of the attributes, (which is one for binary attributes) and the significance level desired. For example, the rejection region for binary attributes at 0.05 significance level is 3.84. When the number of transactions is large, the cutoff value can be exceeded by a very large number of itemsets.

2.2.1.2 Correlation Coefficient

In statistics, a standard way to measure the degree of association between binary variables is to use Pearson's *ϕ-coefficient*, where

$$\phi = \frac{f_{11}f_{00} - f_{10}f_{01}}{\sqrt{f_{1+}f_{0+}f_{+1}f_{+0}}}$$

The value of this coefficient ranges from -1 to +1. *ϕ-coefficient* is desirable because it relates to statistical correlation, a widely accepted measure in statistics. However, it is also symmetric in terms of exchanging $f_{11} \leftrightarrow f_{00}$ and $f_{10} \leftrightarrow f_{01}$. This could be a problem, because it can not distinguish between

large f_{00} from large f_{11}. For example, Table 2.2 should correspond to a more interesting pattern compared to the one in Table 2.3, yet their *ϕ-coefficients* stay the same, since Table 2.2 has a larger support.

	B	¬B	
A	50	20	70
¬A	20	10	30
	70	30	100

Table 2.2: A 2 x 2 contingency table of a more interesting pattern

	B	¬B	
A	10	20	30
¬A	20	50	70
	30	70	100

Table 2.3: A 2 x 2 contingency table of a less interesting pattern

Other measures from statistical literature include Goodman and Kruskal's λ and τ coefficients, Pearson's coefficient of contingency, uncertainty coefficients, etc [67].

2.2.2 Objective Measures From Data Mining

2.2.2.1 Support and Confidence

A rule $A \rightarrow B$ has support s if $s\%$ of all the transactions contains both A and B. The rule has a confidence c if $c\%$ of all transactions that contain A also contains B. Those rules that exceed a predetermined minimum threshold for support and confidence are considered to be interesting. Since the choice of an appropriate support threshold is often ad-hoc, it should be ensured that support-based pruning would not remove many of the interesting patterns. Kumar et al. [67] indicate that placing a maximum support threshold will lead to pruning uncorrelated, positively correlated and negatively correlated itempairs in equal proportions to their initial distribution. In contrast, if a minimum support threshold is specified, most of the itempairs removed are either uncorrelated or negatively correlated [67]. But, if the minimum support threshold is

too high, it tends to produce patterns that are obvious to most analysts. On the other hand, a low minimum support threshold may result in an unmanageable number of patterns. Furthermore, some of the most interesting patterns may have very low support, e.g. patterns involving expensive jewelries. Currently, there have been some attempts to rectify this problem such as assigning weights to different items [8], using a relative interestingness measure [32] or employing a combination of random sampling and hashing techniques [12] to mine exception rules. Hussain et al. argue that objective measures are always reliable due to their unbiased nature, but those measures are sometimes completely unable to justify a rule's interestingness as they can not handle knowledge from common sense rules [32]. The example below illustrates the situation:

$A \rightarrow X$	Common Sense Rule (Strong Pattern)
	(High support, high confidence)
$A, B \rightarrow \neg X$	Exception Rule (Weak Pattern)
	(Low support, high confidence)
$B \rightarrow \neg X$	Reference Rule
	(Low support and/or low confidence)

Hussain et al. search for reliable exceptions starting from the common sense rules [32]. They find exception rule from the two common sense rules $A \rightarrow X$ and $B \rightarrow X$ ($B \rightarrow X$ as common sense infers $B \rightarrow \neg X$ to be reference for its obvious low support and/or low confidence). By doing this, they can estimate the amount of surprise the exception rule brings from the knowledge of extracted rules. If only the above exception rule were given, it would not be objectively interesting. However, if the other two common sense rules ($A \rightarrow X$ and $B \rightarrow X$) are known then the exception rule should be interesting. This suggests a need to have a relative interestingness measure. That is, Hussain et al. mine rules that are objectively interesting, but in the meantime they measure interestingness with respect to already mined rules.

Cohen et al. state that Apriori algorithm is only effective when the only rules of interest are relationships that occur very frequently [12]. However, there are some applications, such as identification of similar web documents, where the rules of interest have comparatively few instances in the data. In these cases, we must look for highly correlated items, or possibly even causal

relationships between infrequent items. Cohen et al. develop a family of algorithms for solving this problem, employing a combination of random sampling and hashing techniques [12].

Since support is an objective measure, it does not involve domain knowledge. Incorporating domain knowledge may require having an subjective interestingness measure. Sarawagi et al. propose using temporal description length to approximate the role of domain knowledge in the search for interesting patterns [10]. They concentrate on the problem of boolean market basket data and states that a set of k items is interesting, not necessarily because its support exceeds a user-defined threshold, but because the relationship between the items changes over time and these changes are not totaly explained by the changes in the support of smaller subsets of items. Sarawagi et al. propose a precise characterization of surprise based on the number of bits in which a basket sequence can be encoded under a carefully chosen coding scheme [10]. In this scheme, it is inexpensive to encode sequences of itemsets that have steady, hence likely to be well known, correlation between items. Conversely, a sequence with large code length hints at a possibly surprising correlation.

The Confidence measure can be misleading in some situations, as explained in the following example.

	Windows	¬Windows	
Linux	20	10	30
¬Linux	60	10	70
	80	20	100

Table 2.4: A 2 x 2 contingency table example pointing out a situation where confidence is misleading

Suppose the support and confidence thresholds were set at 5% and 50%, respectively. The association rule Linux → Windows would have a 20% support and 67% confidence. Thus, it will pass both threshold conditions and eventually declared to be interesting. However, this information is misleading. The prior probability that a customer buys Windows is 80%. Once it is known that the customer had bought Linux, the conditional probability that he or she would buy Windows reduces to 67%. In other words, the discovered rule does not

make sense. This is why confidence may not be an appropriate measure.

If it is assumed that the overall confidence of an itemset is represented by the maximum confidence among the rules that can be generated from this itemset, it is observed that there is a linear relationship between Pearson's ϕ *-coefficient* and the overall confidence in the range of typically encountered support values. In fact, the most interesting rule according to any objective measure must reside along a support border [5].

2.2.2.2 Interest Factor

The Interest factor is another objective measure of association between items [67]. It is defined as the ratio between the joint probability of two items to their marginal probabilities:

$$I(X,Y) = \frac{P(X,Y)}{P(X)P(Y)} = \frac{f_{11}N}{f_{1+}f_{+1}}$$

This ratio can range anywhere between 0 and $+\infty$. An interest factor of 1 corresponds to total independence between the two items. Positively (or negatively) correlated items will have an interest factor greater (less) than 1. Unfortunately, this measure can be close to one (independence) even though the two items are highly dependent on each other. For example, let's consider the following situation: Let P(X,Y) = 0.1, P(X) = 0.1 and P(Y) = 0.1. The interest factor is large i.e., 1 / 0.1 = 10, while its ϕ*-coefficient* is equal to 1 (perfect positive correlation). Now, if P(X,Y) = 0.9, P(X) = 0.9 and P(Y) = 0.9, once again the ϕ*-coefficient* is equal to 1. However, the interest factor has dropped dramatically from 10 to 1 / 0.9 = 1.11, which is very close to the independence situation.

This objective measure also has a linear relationship (strong correlation) with ϕ*-coefficient* in the range of typically encountered support values [67].

2.2.2.3 Conviction

The Conviction measure is proposed as an alternative to interest factor because they needed an asymmetric objective measure for implication rules [67]. Conviction is defined to be:

$$conviction = \frac{P(X)P(\neg Y)}{P(X, \neg Y)}$$

This measure is derived from interest factor in the following way. A rule $X \rightarrow Y$ is logically equivalent to $\neg(X \cap \neg Y)$. Thus, above equation is an asymmetric way for testing independence between X and Y. The ratio between $P(X, \neg Y)$ and $P(X)P(\neg Y)$ is inverted due to the negation symbol in the logical expression $\neg(X \cap \neg Y)$.

Conviction is different from confidence because it does not suffer from the problem of producing misleading rules. Unlike interest factor, conviction will assign the value $+\infty$ if the confidence of the rule is 1 (regardless of what $P(X, Y)$ is). If two items are independent, their conviction value will be equal to 1.

2.2.2.4 Gini Index

For an association rule of the form $X \rightarrow Y$, Gini [67] is defined as follows:

$$Gini = P(X)(P(Y|X)^2 + P(\neg Y|X)^2) + P(\neg X)(P(Y|\neg X)^2 + P(\neg Y|\neg X)^2)$$
$$- P(Y)^2 - P(\neg Y)^2$$

The values for the Gini index range between 0 (when the two items are independent) and 0.5 (when the two items are perfectly correlated). Gini index treats both positively and negatively correlated itemsets in the same way. However, as the range of support values is restricted, the negatively correlated pairs are eliminated.

2.2.2.5 Piatetsky-Shapiro's rule-interest

Piatetsky-Shapiro's rule-interest is defined as:

$$RI = P(X,Y) - P(X)P(Y)$$

The range of this measure is between -0.25 and 0.25. If X and Y are independent, $RI = 0$. RI is maximum when $P(X,Y) = P(X) = P(Y) = 0.5$. This measure also has a strong correlation with *ϕ-coefficient* in the range of typically encountered support values.

2.2.2.6 Entropy (Information Gain Ratio)

Entropy is developed from information theory and is defined for a rule $X \rightarrow Y$ as follows [67]:

$$Entropy = \frac{H_X + H_Y - H_{XY}}{H_X}$$

where

$H_X = -P(X)\log P(X) - P(\neg X)\log P(\neg X)$,

$H_Y = -P(Y)\log P(Y) - P(\neg Y)\log P(\neg Y)$ and

$H_{XY} = \sum_i \sum_j P(X=i, Y=j)\log P(X=i, Y=j)$.

The range of this measure is between 0 (for absolute independence) and 1 (for perfect correlation). This measure treats positive and negatively correlated items in the same way.

2.2.2.7 Neighborhood-based Unexpectedness

Dong and Li introduce neighborhood-based interestingness by considering unexpectedness in terms of neighborhood-based parameters [14]. They first

present some novel notions of distance between rules and of neighborhoods of rules. The neighborhood-based interestingness of a rule is then defined in terms of the pattern of the fluctuation of confidences or the density of the mined rules in some of its neighborhoods. They rank the interesting rules by combining some neighborhood-based characteristics, the support and confidence of the rules, and user's feedback.

Taking the association rules into account, we might think that it is possible to handle the post-mining rule analysis problem by increasing the threshold values. This way, the number of induced association rules will decrease. However, using the world map analogy, only those global peaks will be induced and the useful information conveyed by those local peaks over vast plains will be missed [14]. For example, suppose we have two different geographical regions A and B, where A is a mountainous area with an average altitude of 5000 meters and B is a vast plain with an average altitude of 50 meters. A mountain with a height of 6000 meters in A is not as interesting as a mountain with a height of 1000 meters in B. That is, the interestingness concept should also be related to the position among neighborhoods.

Interestingness is used to evalute the importance of an association rule by considering its unexpectedness in terms of other association rules in its neighborhood. The neighborhood of an association rule consists of all association rules within a given distance. The distance metric is given by:

$$D(R_1, R_2) = \delta_1|(X_1 \cup Y_1)\Theta(X_2 \cup Y_2)| + \delta_2|X_1\Theta X_2| + \delta_3|Y_1\Theta Y_2|$$

where $R_1 = X_1 \rightarrow Y_1$, $R_2 = X_2 \rightarrow Y_2$, δ_1, δ_2 and δ_3 are parameters to weight the relative importance of all three terms, and Θ is an operator denoting the symmetric difference between X and Y (i.e.,$(X - Y) \cup (Y - X)$). An r-neighborhood of a rule is given by the set:

$$N(R_o, r) = \{R|D(R, R_o) \leq r,\ R \text{ a potential rule}\}$$

and is used to define the interestingness of a rule. Two types of interestingness are unexpected confidence and isolated interestingness. *Unexpected Confidence*

Interestingness is given by:

$$UCI = \begin{cases} 1 & \text{if } ||c(R_o) - ac(R_o, r)| - sc(R_o, r)| > t_1 \\ 0 & \text{otherwise} \end{cases}$$

where $c(R_o)$ is the confidence of R_o, $ac(R_o, r)$ and $sc(R_o, r)$ are the average confidence and standard deviation of the confidences of the rules in the set $M \cap N(R_o, r)$ - $\{R_o\}$ (M is the set of rules satisfying the minimum support and confidence), and t_1 is a threshold.

$$II = \begin{cases} 1 & \text{if } |N(R_o, r)| - |M \cap N(R_o, r)| > t_2 \\ 0 & \text{otherwise} \end{cases}$$

where $|N(R_o, r)|$ is the number of potential rules in an r-neighborhood, $|M \cap N(R_o, r)|$ is the number of rules generated from the neighborhood, and t_2 is a threshold.

2.2.3 Subjective Measures

One approach to defining interestingness of a pattern is to define it in objective terms, where interestingness of a pattern is measured in terms of its structure and the underlying data used in the discovery process. However, that objective measures of interestingness, although useful in many respects, usually do not capture all the complexities of the pattern discovery process, and are not sufficient in many data mining applications because one can still generate a large number of strong rules that are interesting "objectively" but of little interest to the user. Thus, subjective measures of interestingness are needed to define interestingness of a pattern.

2.2.3.1 Silberschatz's Belief System

The subjective measures do not depend only on the structure of a rule and on the data used in the discovery process, but also on the user who examines the pattern [64]. These measures recognize that a pattern that is of interest to one

user, may be of no interest to another user. There are two major reasons why a pattern is interesting from the subjective (user-oriented) point of view:

- **Actionability Measure**
  A pattern is interesting if the user can do some action after he/she sees the pattern. The user can use the outcomes of the interesting pattern to his/her advantage. Actionability is an important subjective measure of interestingness because users usually prefer to see the knowledge that makes their lives easy, by taking proper actions in response to the newly discovered and learned patterns.

- **Unexpectedness Measure**
  Unexpectedness measure can be analyzed both objectively and subjectively. A newly discovered pattern can be surprising to the user, which automatically leads it to be an interesting pattern. Surprising or unexpected patterns are interesting since they contradict expectations of the human beings, and the expectations naturally depend on the belief system. It is therefore convenient to define the unexpected measure of interestingness in terms of the belief system that the user owns. Silberschatz et al. express interestingness of a pattern in terms of how it affects the belief system [64].

  Unexpectedness and actionability are two different types of interestingness. There are patterns that are unexpected but non-actionable. There are patterns that are actionable but expected. There are also patterns that are both unexpected and actionable. It is worthwhile to note that most of the actionable patterns are also unexpected; and most of the unexpected patterns are also actionable [64]. Therefore, unexpectedness and actionability are good substitutes for each other.

  Although both actionability and unexpectedness are important, people are usually interested in actionability since they prefer to react to the patterns to make their lives easy. However, actionability is very difficult to capture formally [64] because the space of all patterns should be partitioned into a finite number of equivalence classes and a proper set of actions should be associated with each equivalence class. The space of all patterns is usually unknown in many situations. Even if the space

is known, it is a very hard task to partition the space into equivalence classes and to associate a proper set of actions with each equivalence class. Even if the space partitioning and the action associating steps succeed, there is no guarantee that the actions and the association of the actions to the equivalence classes will never change. These difficulties make actionability difficult to capture formally.

Unexpectedness and actionability are good substitutes for each other. Most of the unexpected patterns are actionable and most of the actionable patterns are unexpected. Therefore, we can handle actionability through unexpectedness [64]. Unexpectedness is related to beliefs and beliefs are classified into two categories:

- Hard beliefs are beliefs that can never be changed despite contradictory evidence.
- Soft beliefs are beliefs that a user is willing to change if compelling new evidence are found.

Interestingness determines the extent to which a soft belief is changed as a result of encountering new evidence (i.e., discovered knowledge). A pattern is interesting relative to some belief system if it affects this system, and the more it affects it, the more interesting the pattern is. Interestingness within the context of soft beliefs is given by:

$$I = \sum_{\alpha} \frac{P(\alpha|E,\epsilon) - P(\alpha|\epsilon)}{P(\alpha|\epsilon)}$$

where α is a belief, E is new evidence, ϵ is the previous evidence supporting belief α, $P(\alpha|\epsilon)$ is the confidence in belief α, and $P(\alpha|E,\epsilon)$ is the new confidence in belief α given the new evidence E. Summation is over all beliefs. The Bayes theorem is used to determine the new confidence and is given by:

$$P(\alpha|E,\epsilon) = \frac{P(E|\alpha,\epsilon)P(\alpha|\epsilon)}{P(E|\alpha,\epsilon)P(\alpha|\epsilon) + P(E|\neg\alpha,\epsilon)P(\neg\alpha|\epsilon)}$$

The Bayesian approach can be applied to arbitrary beliefs not only for beliefs expressed as rules [64].

2.2.3.2 Liu's Fuzzy Matching Technique for Classification Rules

Rule induction research implicitly assumes that after producing the rules from a data set, these rules will be used directly by an expert system or a human user. In real-life applications, the situation may not be as simple as that, particularly, when the user of the rules is a human being. The human user almost always has some previous concepts or knowledge about the domain represented by the data set. Naturally, he/she wishes to know how the new rules compare with his/her existing knowledge [41]. With the increasing use of machine learning techniques in practical applications such as data mining, this issue of post-analysis of rules warrants greater emphasis and attention. Liu and Hsu perform the post-analysis of classification rules generated by systems such as C4.5. They propose a fuzzy matching technique to perform the post-analysis of classification rules [41].

Most of the work on machine learning focuses on the generation of rules from various types of data sets as well as pruning of the generated rules [59, 7]. Some systems also use existing domain knowledge in the induction process [53, 11, 57]. However, their purpose is mainly for helping the induction process so as to increase learning efficiency and/or improve prediction accuracy of the generated rules. Clearly, the focus of their research is quite different from the one presented by Liu and Hsu, which is primarily a post-analysis method that aims to help the user analyze the rules generated.

In the fuzzy matching technique developed to perform the post-analysis of rules, existing rules, E, (from previous knowledge) are regarded as fuzzy rules and are represented using fuzzy set theory. The newly generated rules, B, are matched against the existing fuzzy rules using the fuzzy matching technique. The matching process results in identifying conforming and unexpected rules.

They present a high level view of the fuzzy matching method. It consists of two main steps:

- The user converts each rule in E to a fuzzy rule. The fuzzy rule has the same syntax as the original rule, but its attribute values must be described using some fuzzy linguistic variables.

- The system matches each new rule $B_i \in B$ against each fuzzy rule $E_j \in E$ in order to obtain the degree of match for each new rule B_i against the set E. The new rules in B are then ranked according to their degrees of match with E.

The rules in E and B have the same syntax and semantics as the rules produced by C4.5. The syntax of the rules generated by C4.5 has the following form:

$$P_1, P_2, ..., P_n \rightarrow C$$

where "," means "and", and P_i is a proposition of the form: *attr OP value*, where *attr* is the name of an attribute in the data set, *value* is a possible value for *attr*, and $OP \in \{=, \neq, <, >, \leq, \geq\}$ is the operator. C is the consequent of the form: $Class = value$.

2.2.3.3 Liu's Tuple-level Fuzzy Matching Technique for Classification Rules

The technique proposed by Liu et al. asks the user to provide his/her expected patterns according to his/her past knowledge and/or intuitive feelings [43]. Given these expectations, the system uses a tuple-level fuzzy matching technique to analyze and rank the discovered patterns according to a number of interestingness measures. In this technique, a number of rankings can be performed for different purposes. Two main types of ranking are conformity ranking and unexpectedness ranking.

The conformity can be measured at the pattern-level and at the tuple-level. Liu and Hsu propose a pattern-level fuzzy matching technique [41]. For example;

Discovered pattern (or rule): If X > 9, Y < 5, Q=2 Then R = TRUE

User-expected pattern (or rule): If X > 9, Y < 6, P=4 Then R =TRUE

On the surface, the two rules seem similar (or conforming). The pattern-level match method in [41] will give a high conforming match value. However, it may be the case that tuples in the database suggest that (X>9, Y<5, Q=2) implies (X>9, Y<5, P=4) -the two patterns are conforming, or that (X>9, Y<5, P=4) implies R = FALSE -the user expected pattern is actually false, or other possible situations. Without consulting the actual database, it cannot be decided which one of these cases is true. However, it is obvious that pattern level matching is efficient.

2.2.3.4 Liu's General Impressions for Classification Rules

Liu et al. propose a technique that analyzes the discovered classification rules against a specific type of existing knowledge, which they call general impressions, to help the user identify interesting rules [42]. They first propose a representation language to allow general impressions to be specified. They then present some algorithms to analyze the discovered classification rules against a set of general impressions. The results of the analysis tell us which rules conform to the general impressions and which rules are unexpected. Although both unexpected and conforming rules are considered to be interesting, unexpected rules are more interesting.

Liu and Hsu report a fuzzy matching approach to analyze the discovered rules against the user's existing concepts [41]. One limitation of this technique is that too much reliance is being placed on the user's ability to supply the set of fuzzy expectations. In many situations, users do not know enough about their domains to supply the expected rules. Instead, Liu and Hsu find that even if the users cannot supply the set of fuzzy expectations, they do have certain general impressions (GI) about their domains [42]. Silberschatz et al. propose to use a belief system to describe unexpectedness [64]. However, this approach requires the user to provide complex belief information, such as conditional probabilities, which are difficult to obtain in practice. It does not handle GIs.

Assume a human user has some previous concepts about the domain represented by the database D. These concepts can be correct, partially correct or entirely wrong. Two types of existing concepts exist:

- **General Impressions (GI):** The user does not have detailed concepts about the domain, but does have some vague feelings. For example, in a housing loan domain, the user may feel that having a high monthly salary increases one's chance of obtaining a loan.
- **Reasonably Precise Knowledge (RPK):** The user has more definite idea. For example, in the same loan domain, the user may believe that if one's monthly salary is over $5000; one will be granted a loan. Of course, the user may not be so sure that it is exactly $5000. There is a fuzziness surrounding the value $5000 in his/her mind.

Liu and Hsu study the rule analysis against RPK [41], where as Liu et al. focus on GIs [42]. In the situation where one has some RPK about certain aspects of the domain, but only GIs about the others, a combined approach may be used.

Liu et al. analyze classification rules produced by C4.5 [42], as in the work described in [41]. A general impression is used to evaluate the importance of classification rules by comparing discovered rules to an approximate or vague description of what is considered to be interesting. So, a general impression is a kind of specification language. There are two types of general impressions that can be specified: Type 1 and Type 2.

A Type 1 general impression is a rule of the form $A_1OP_1, A_2OP_2...A_xOP_x \rightarrow C_j$, where each A_iOP_i is called an impression term, each A_i is an attribute, each OP_i is an impression descriptor from the set $\{<, >, <<, |, \square\}$, and C_j is a class. The $<$ ($>$) impression descriptor means smaller (larger) attribute values are more likely to lead to inclusion in class C_j, $<<$ means some range of attribute values are more likely to lead to inclusion in class C_j, $|$ means some relationship exists between an attribute and class C_j but the nature of this relationship is not exactly known, and $\square$ means that some subset of the possible values for an attribute are more likely to lead to inclusion in class C_j.

A Type 2 general impression is specified when there is more confidence that the combination of impression terms will lead to inclusion in class C_j. A Type 2 general impression is a rule of the form $A_1OP_1, A_2OP_2...A_kOP_k$ &

$A_mOP_m, A_nOP_n...A_xOP_x \rightarrow C_j$, where the part to the left (right) of the & symbol is called the core (supplement). The core must always exist, otherwise the general impression should be specified as Type 1. If the supplement exists, then the rule is called a maximal impression. In a maximal impression, the general impression is that the impression terms in the core and any subset of those in the supplement are more likely to lead to inclusion in class C_j. If the supplement does not exist, then the rule is called an exact impression. In an exact impression, the general impression is that the impression terms in the core are more likely to lead to inclusion in class C_j. The specified general impressions are matched against the rules generated, and ranked to identify those that are most valid [42].

2.2.3.5 Rule Templates for Association Rules

Klemetinen et al. show how a formalism of rule templates makes it possible to easily describe the structure of interesting rules [37]. They also give examples of visualization of rules, and show how a visualization tool interfaces with rule templates. Templates, which are closer to regular expressions, can be used to describe the form of interesting rules, and also to specify which rules are not interesting.

For a rule to be presented to the user, a rule must be interesting; i.e., it should match one of the inclusive templates and it must not be uninteresting; i.e., it should not match with any of the restrictive templates [37]. That is, to be interesting, a rule has to match an inclusive template. If a rule, however, matches a restrictive template, it is considered as uninteresting. Rule pruning can be done by setting support, confidence, and rule size thresholds. But, although rule pruning based on support and confidence thresholds is effective, it fails to take into account special interests or domain knowledge. So, subjective measures such as rule templates provide effective solutions. The simple idea of classifying the attributes of the original data set to an inheritance hierarchy, and using templates defined in terms of that hierarchy, can be used to prune the rule sets effectively and according to the user's intuitions. The drawback of rule templates is that the degree of interestingness is not specified [37]. To

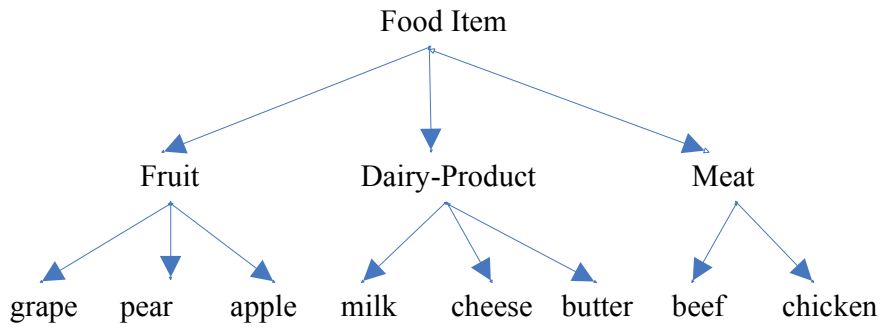

Figure 2.1: A Taxonomy Example

give an interestingness value to the discovered rules, inclusive templates could be given weights.

2.2.3.6 Liu's General Impressions, Reasonably Precise Concepts and Precise Knowledge for Association Rules

Liu et al. propose a new approach to assist the user in finding the interesting rules (in particular, unexpected rules) from a set of discovered association rules [44]. This technique is characterized by analyzing the discovered association rules using the user's existing knowledge about the domain and then ranking the discovered rules according to various interestingness criteria, e.g., conformity and various types of unexpectedness.

Before discussing the proposed technique, they first introduce the concept of association rules, in particular, generalized association rules. The generalized association rule model is more general than the original association rule model.

The (generalized) association rule mining is defined as follows: Let $I = \{i_1, ..., i_w\}$ be a set of items. Let G be a directed acyclic graph on the items. An edge in G represents an is-a relationship. Then, G is a set of taxonomies. A taxonomy example is shown in Fig. 2.1 (taken from [44]). Let T be a set of transactions, where each transaction t is a set of items such that $t \subseteq I$. A (generalized) association rule is an implication of the form $X \rightarrow Y$, where $X \subset I$, $Y \subset I$, and $X \cap Y = \emptyset$. The rule $X \rightarrow Y$ holds in the transaction set T with confidence c if $c\%$ of transactions in T that support X also support Y. The rule has support s in T if $s\%$ of the transactions in T contains $X \cup Y$.

For example; an association rule in the transaction set T could be:

grape→ apple [support = 10%, confidence = 60%]

which says that 10% of people buy grape and apple together, and 60% of the people who buy grape also buy apple. This rule only involves items at the bottom level of the taxonomy. Rules involving items of more than one level also exist:

Fruit → Dairy-Product

Fruit, milk → Meat

The proposed technique consists of 3 components [44]:

- A specification language: it allows the user to specify his/her various types of existing knowledge.
- An interestingness analysis system: It analyzes the discovered association rules using the user's specifications, and through such analysis, to identify: conforming rules, unexpected consequent rules, unexpected condition rules and both side unexpected rules.
- A visualization system: It enables the user to visually detect interesting rules easily.

Srikant et al. propose an association rule-mining algorithm that can take item constraints specified by the user in the rule mining process so that only those rules that satisfy the constraints are generated [66].

This association rule mining algorithm and rule templates view the process of finding subjectively interesting rules as a query-based process, although the queries may be considered during the rule generation or after all rules have been considered. It is hard to find the truly unexpected rules. Many rules that do not satisfy the user's queries may also be of interest. It is just that the user has never thought of them or has forgotten about them. Technique by Liu et al. not only identifies those conforming rules as query-based methods, but also provides three types of unexpected rules [44].

Liu et al. reported two techniques for analyzing the subjective interestingness of classification rules [41, 42]. However, these techniques cannot be applied to analyzing association rules. Association rules require a different specification language and different ways of analyzing and ranking the rules. Tuzhilin et al. propose a method of discovering unexpected patterns that takes into consideration a set of expectations or beliefs about the problem domain [54]. The method discovers unexpected patterns using these expectations to seed the search for patterns in data that contradict the beliefs. However, this method is not an efficient post-analysis method unless the user is able to specify his/her beliefs about the domain completely beforehand, which is very difficult. It does not handle user's rough or vague feelings, but only precise knowledge. Silberschatz et al. propose to use belief systems to describe unexpectedness [64]. These approaches require the user to provide complex belief information, such as conditional probabilities, which are difficult to obtain in practice.

Liu et al. present Interestingness Analysis System [44]. It is an interactive and iterative technique. In each iteration, it first asks the user to specify his/her existing knowledge about the domain. It then uses this knowledge to analyze the discovered rules according to some interestingness criteria, conformity and various types of unexpectedness, and through such analysis to identify those potentially interesting rules.

The specification language allows three types of specifications. Each represents knowledge of a different degree of preciseness. They are general impressions (GI), reasonably precise concepts (RPC) and precise knowledge (PK).

GI is of the form:

$GI(< S_1 \dots S_m >)$([support, confidence] OPTIONAL)

Each S_i is either an item, a class, or an expression $C+$ or $C*$, where C is a class. A discovered rule: $a_1 \dots a_n \rightarrow b_1 \dots b_k$, conforms to the GI if $< a_1 \dots a_n, b_1 \dots b_k >$ can be considered to be an instance of $< S_1 \dots S_m >$, otherwise it is unexpected with respect to the GI.

For example, $GI(< a, \{b, c\}+ >)$ can be expanded into:

$(a \rightarrow b) \cup (a \rightarrow c) \cup (b \rightarrow a) \cup (c \rightarrow a) \cup ((a \cap b) \rightarrow c) \cup ((a \cap c) \rightarrow b) \cup ((b \cap c) \rightarrow a) \cup (a \rightarrow (b \cap c)) \cup (b \rightarrow (a \cap c)) \cup (c \rightarrow (a \cap b)) \ldots$

RPC is of the form:

$RPC(< S_1 \ldots S_m \rightarrow V_1 \ldots V_g >)|$([support, confidence] OPTIONAL)

and *PK* is of the form:

$PK(< S_1 \ldots S_m \rightarrow V_1 \ldots V_g >)$[support, confidence].

2.2.3.7 Interestingness via What is Not Interesting

To determine what is subjectively interesting, the user's domain knowledge is needed to be incorporated into the system. The KDD community provides the following approach: They require a domain expert or an advanced user to formally (even if vaguely) express, using a predefined grammar, what he finds (not) interesting or what a domain user already knows (Rule templates, General impressions). The success of these strategies is conditioned upon the availability of a domain expert willing to go through the significant effort of completing this task. Unfortunately, acquiring such an expert for the duration of the process is a costly procedure if such an expert is available.

To overcome these difficulties, Sahar et al. present a simple and short process of eliminating a substantial portion of uninteresting association rules in a list outputted by a data-mining algorithm [61]. Instead of trying to establish what is interesting, they look for rules that are not interesting; more specifically, the simple rules whose elimination implies the automatic elimination of many other rules in the list.

They ask a user to classify only a few rules, specifically chosen so that their elimination can bring about the automatic elimination of many other rules [61]. This approach has several benefits. (1) They circumvent the major difficulty of defining why a certain rule is interesting to a user. (2) They ask only classification questions. The questions are not descriptive in nature and therefore easier and quicker for a user to answer. (3) They make the

classification process simpler by having a user classify only very simple rules: $a \rightarrow b$ where both a and b are literals. (4) Every classification a user makes can potentially eliminate an entire family of rules, not only a single rule, meaning that we need to ask fewer questions. (5) They do not require a domain expert to classify the rules; a naive user can successfully classify most of the rules due to their simplicity.

Chapter 3

A New Benefit-Maximizing, Feature Projections Based Classification Learning Algorithm

In this thesis, we propose a method that has the ability to formulate the interestingness issue of association rules as a benefit-maximizing classification problem. Therefore, it makes sense to choose an appropriate classifier to be used in the proposed method. This chapter is devoted to the choice of an appropriate benefit-maximizing classifier. Upon selection of an appropriate classifier, modeling interestingness of streaming association rules as a benefit-maximizing classification problem will be explained in Chapter 4.

In this chapter, we propose a new benefit-maximizing classifier, namely "Benefit-Maximizing Classifier by Voting Feature Projections" (*BMCVFP*). It is a benefit-maximizing, feature projections based classification learning algorithm. The training phase of *BMCVFP* not only works incrementally, but also preserves order-independency among the training instances. The classical classification learning algorithms generally work on data sets having *linear* and/or *nominal* features. On the other hand, *BMCVFP* can also work on data sets having *ordered pair of sets* type features. This new feature type, whose

definition is given below, is introduced. Because, in interesting modeling of association rules as a classification problem, some features of the model will be *ordered pair of sets* type.

> **Definition 1:** *An ordered pair of sets type feature f is a feature whose values are of the form (set1, set2) where (set1, set2) ≠ (set2, set1).*

The chapter is organized as follows. Section 3.1 reviews knowledge representation by feature projections. Section 3.2 explains the basic concepts for benefit-maximizing classification by voting feature segments. Sections 3.3 and 3.4 are devoted to the training and the classification phases of the *BMCVFP* classifier, respectively.

3.1 Knowledge Representation by Feature Projections

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 [13, 21, 22, 23, 24, 25, 26, 65]. 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 [36]. 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 [36].

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 Naive Bayesian classifier, which also assumes the features are independent [56]. Note that votes are summed in feature projections approach, while probabilities are multiplied in the Naive Bayesian classifier. Naive 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 [56]. Valev proposed the parallelized version of feature projections based approach, which processes each feature in parallel [69]. Ko and Seo applied feature projections to the text categorization problem [38].

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

$I =< lbv, ubv, N_1, N_2, \ldots, N_s, V_1, V_2, \ldots, V_s >$

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 [25, 26, 65], a segment represents examples from a single class, whereas the authors in [13, 22, 23] 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 $lbv = ubv$.*

$I =< lbv, ubv, N_1, N_2, \ldots, N_s, V_1, V_2, \ldots, V_s >$

The existing feature projections based classifiers can be trained incrementally. However, they do not preserve order-independency [70]. 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, that are constructed on nominal features. This fact motivated us to construct only point segments on the linear features. In the case of ordered pair of sets type features, each ordered pair (set1, set2) is assumed to be a point segment where $lbv = (set1, set2)$ and $ubv = lbv$.

On a nominal and ordered pair of sets type feature, the number of feature values is limited, so it is possible to save each observed feature value (each observed ordered pair of sets, in the case of ordered pair of sets type feature) 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 ordered pair of sets type features. Therefore, 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 to use 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 [3, 4]. 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:

$$N_c = classcount[c]\,(lim_{\Delta x \to 0}(gpdf_{f,c}(x)\Delta x)) \tag{3.1}$$

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.

$$gpdf_{f,c}(x) = \frac{1}{\sigma[f,c]\sqrt{2\pi}} e^{-\frac{(x-\mu[f,c])^2}{2(\sigma[f,c])^2}} \tag{3.2}$$

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.

$$\sigma[f,c] = \sqrt{\mu^2[f,c] - (\mu[f,c])^2} \tag{3.3}$$

where $\mu^2[f,c]$ is the mean of the squares of the f values of training instances of class c.

3.2 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 [68]. Margineantu showed that cost based approaches are equivalent to benefit based approaches if the amount of benefit achieved after classification is not relevant [46]. 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 x 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 [20, 24, 34, 35]. 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 making it possible to realize 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* $I =< lbv, ubv, N_1, N_2, \ldots, N_s, V_1, V_2, \ldots, V_s >$ *is given as:*

$$MinBenefit(I) = \sum_{c}(N_c(B[\underset{i}{argmin}B[i,c],c]))$$

Definition 6: *Given a benefit matrix B, the maximum benefit attainable on a segment* $I =< lbv, ubv, N_1, N_2, \ldots, N_s, V_1, V_2, \ldots, V_s >$ *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* $I =< lbv, ubv, N_1, N_2, \ldots, N_s, V_1, V_2, \ldots, V_s >$ *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 [34]. 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. 3.4. This formulation ensures that the smaller the probability of a class is, the more the benefit of correctly classifying that class is.

$$B[i,j] = \begin{cases} 0 & \text{if } i \neq j \\ \frac{1}{prob(j)} = \frac{\sum_c classcount[c]}{classcount[j]} & \text{else} \end{cases} \tag{3.4}$$

Using Eq. 3.4, $MinBenefit$, $MaxBenefit$, $SegmentBenefit$ and finally $SegmentClassVote$ definitions simplify to the following:

$$\begin{aligned} MinBenefit(I) &= \sum_c (N_c(B[\underset{i}{argmin} B[i,c], c])) \\ &= \sum_c (N_c(0)) = 0 \end{aligned} \tag{3.5}$$

$$\begin{aligned} MaxBenefit(I) &= \sum_c (N_c(B[c,c])) \\ &= \sum_c \left(N_c \frac{\sum_i classcount[i]}{classcount[c]} \right) \end{aligned} \tag{3.6}$$

$$\begin{aligned} SegmentBenefit(I,k) &= \sum_c (N_c(B[k,c])) \\ &= N_k(B[k,k]) \\ &= N_k \frac{\sum_i classcount[i]}{classcount[k]} \end{aligned} \tag{3.7}$$

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

$$= \frac{\frac{N_k}{classcount[k]}}{\sum_c \frac{N_c}{classcount[c]}} \tag{3.8}$$

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. 3.1 in Eq. 3.8, $SegmentClassVote$ definition can be rewritten in a generic form for linear features as:

$$\begin{aligned}
SegmentClassVote(I,k) &= \frac{\frac{N_k}{classcount[k]}}{\sum_c \frac{N_c}{classcount[c]}} \\
&= \frac{\frac{classcount[k] lim_{\Delta x \to 0}\left(gpdf_{f,k}(x)\Delta x\right)}{classcount[k]}}{\sum_c \frac{classcount[c] lim_{\Delta x \to 0}\left(gpdf_{f,c}(x)\Delta x\right)}{classcount[c]}} \\
&= \frac{lim_{\Delta x \to 0}\left(gpdf_{f,k}(x)\Delta x\right)}{\sum_c lim_{\Delta x \to 0}\left(gpdf_{f,c}(x)\Delta x\right)} \\
&= \frac{lim_{\Delta x \to 0}\left(gpdf_{f,k}(x)\Delta x\right)}{lim_{\Delta x \to 0}\left(\sum_c gpdf_{f,c}(x)\Delta x\right)} \\
&= lim_{\Delta x \to 0}\left(\frac{gpdf_{f,k}(x)\Delta x}{\sum_c gpdf_{f,c}(x)\Delta x}\right) \\
&= \frac{gpdf_{f,k}(x)}{\sum_c gpdf_{f,c}(x)}
\end{aligned} \tag{3.9}$$

3.3 Training in the BMCVFP Algorithm

There are various types of benefit-maximizing classifiers in the literature [20, 24, 34, 35]. However, they do not learn the concept description incrementally

in the training phase and none of them are suitable for data sets including ordered pair of sets type features. Therefore, we designed 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 ordered pair of sets type features. 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. 3.1. 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 works incrementally. We keep the number of training instances of class c in $classcount[c]$. Let t, with class t_c, be the incoming training instance. If it is the first training instance, we perform the $classcount[c]$ initialization task for each class c at lines $1-5$. Following this, $classcount[t_c]$ is incremented and benefit matrix is updated by the *UpdateBenefitMatrix* algorithm given in Fig. 3.2. The rest of the training phase differs according to the type of the features.

For a nominal and ordered pair of sets type feature f (lines $11-22$), 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.

For a linear feature f (lines $23-25$), we update the Gaussian probability distribution function of f values of training instances of class t_c by using the *UpdateGaussianProbDistributionFunction* algorithm given in Fig. 3.3. This algorithm updates $\mu[f, t_c]$, $\mu^2[f, t_c]$, $\sigma[f, t_c]$ and finally $gpdf_{f,t_c}(x)$.

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
[3]        initialize classcount[c] to 0
[4]     end for
[5]  end if
[6]  let t_c be the class of t
[7]  increment classcount[t_c]
[8]  UpdateBenefitMatrix(classcount)
[9]  for each feature f do
[10]    let t_f be the feature value of t on f
[11]    if f is a nominal or ordered pair of sets type feature then
[12]       if t_f exists as a point segment then
[13]          increment segment_class_count[f, t_f, t_c]
[14]       end if
[15]       else
[16]          add a new point segment t_f
[17]          for each class c do
[18]             initialize segment_class_count[f, t_f, c] to 0
[19]          end for
[20]          set segment_class_count[f, t_f, t_c] to 1
[21]       end else
[22]    end if
[23]    else // f is a linear feature
[24]       UpdateGaussianProbDistributionFunction(f, t_c, t_f)
[25]    end else
[26]    UpdateSegmentsClassVotes(f)
[27] end for
[28] return point segments' class votes on each feature
```

Figure 3.1: 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] $B[i,j] \leftarrow \frac{1}{prob(j)} = \frac{\sum_c classcount[c]}{classcount[j]}$
[5] end if
[6] else
[7] $B[i,j] = 0$
[8] end else
[9] end for
[10] end for
[11] return B

Figure 3.2: The UpdateBenefitMatrix Algorithm

Algorithm: UpdateGaussianProbDistributionFunction(f, c, t_f)
Input:
Feature, f; Class, c; f value of the new training instance t, t_f
Output:
Updated Gaussian Probability Distribution Function of f values of training instances of class c, $gpdf_{f,c}(x)$
Method:
[1] $\mu[f,c] \leftarrow \frac{(classcount[c]-1)\mu[f,c]+t_f}{classcount[c]}$

[2] $\mu^2[f,c] \leftarrow \frac{(classcount[c]-1)\mu^2[f,c]+(t_f)^2}{classcount[c]}$

[3] $\sigma[f,c] \leftarrow \sqrt{\mu^2[f,c] - (\mu[f,c])^2}$

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

[5] return $gpdf_{f,c}(x)$

Figure 3.3: UpdateGaussianProbDistributionFunction(f,c,t_f) Algorithm

Algorithm: UpdateSegmentsClassVotes(f)
Input:
Feature, f
Output:
Updated class votes of point segments of f
Method:
[1] if f is a nominal or ordered pair of sets type feature then
[2] for each point segment p on feature projection f do
[3] for each class c do
[4] $$segment\_class\_vote[f,p,c] \longleftarrow \frac{\frac{segment\_class\_count[f,p,c]}{classcount[c]}}{\sum_i \frac{segment\_class\_count[f,p,i]}{classcount[i]}}$$
[5] end for
[6] end for
[7] return $segment\_class\_vote[f,p,c]$ $(\forall p, c)$
[8] end if
[9] else // f is a linear feature
[10] for each class c do
[11] $$segment\_class\_vote[f,x,c] \longleftarrow \frac{gpdf_{f,c}(x)}{\sum_i gpdf_{f,i}(x)}$$ $(\forall x \in \Re)$ //Generic Computation
[12] end for
[13] return $segment\_class\_vote[f,x,c]$ $(\forall c)$
[14] end else

Figure 3.4: UpdateSegmentsClassVotes(f) Algorithm

There is no need to maintain the training instances in the *BMCVFP* algorithm. *BMCVFP* is an incremental classification learning 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. $\mu[f,c]$ and $\mu^2[f,c]$ values 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 nominal and ordered pair of sets type feature f, we need to store segments and class distribution of training instances on each segment s, $segment\_class\_count[f,s,c]$.

The training phase concludes by updating the class votes of point segments on each feature projection by using the *UpdateSegmentsClassVotes* algorithm given in Fig. 3.4.

3.4 Classification in the BMCVFP Algorithm

The classification phase of *BMCVFP* is shown in Fig. 3.5. In this phase, the query instance q is projected on each feature dimension f, and each feature calculates its class votes (lines 6-13). For each class c, votes of features for c are summed up to get the aggregate class vote for c (lines 18 and 25). The class x taking the highest aggregate vote is predicted as the class of q. For the predicted class x, the certainty value of the prediction is also given (line 29).

Predicting the class of a query instance q is explained in Fig. 3.7. If all the classes have a zero vote, the predicted class and its associated certainty are taken as "-1". Otherwise, the class x taking the highest aggregate vote is predicted as the class of q. The certainty value of the prediction is taken as the ratio of the aggregate vote of c to the sum of the aggregate votes of all classes. *BMCVFP* has the flexibility to select the features to be involved in the voting process for a query instance q. This flexibility is realized by setting the *use_certainty_on_single_feature_prediction* parameter as "true" and by setting a threshold value for the minimum certainty value, $MinC_v$, parameter.

The class x taking the highest vote from feature f is called the favored class of feature f for the query instance q. If the *use_certainty_on_single_feature_prediction* parameter is set as "true", f can involve in the voting process for q only if $feature\_vote[f, x] \geq MinC_v$ (lines 14-15). If a feature does not participate in the voting process, its class votes are simply taken as "0".

In the classification phase, the class votes of each feature are multiplied by the weight of the corresponding feature (lines 17 and 24). Using feature weighting ensures that some features become more effective in the voting process. If the features are treated equally, feature weights can be selected as "1" for each feature f.

Class vote calculation differs among feature types. On a linear and nominal feature f, query instance has a value of q_f. This value, which is a real number in the case of linear features, constitutes a point segment on feature projection f. Segment class vote calculation taking benefit maximization into account has

been defined for nominal and linear features in Eq.3.8 and Eq.3.9, respectively. The class votes of f for the query instance q falling into the point segment q_f is the segment class votes of q_f.

Class vote calculation of the ordered pair of sets type features is shown in Fig. 3.6. Query instance has a value of q_f on feature dimension f. However, $q_f = (set1, set2)$. 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 among 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 8-10). If q_f does not exist in the saved point segments, then we first multiply the 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 V_1 and V_2 is as follows:

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

Algorithm: BMCVFP$_{query}(q, use_certainty_on_single_feature_prediction, MinC_v)$
Input:
The query instance, q;
Whether minimum certainty requirement will be met on the predictions of each feature, *use_certainty_on_single_feature_prediction*;
Minimum Certainty Value, $MinC_v$
Output:
Predicted class and the certainty value of this prediction, $(prediction, certainty)$
Method:

```
[1]  for each class c do
[2]     initialize final_vote[c] to 0
[3]  end for
[4]  for each feature f do
[5]     let q_f be the feature value of q on f
[6]     if f is an ordered pair of sets type feature then
[7]        CalculateOrderedPairofSetsTypeFeatureVotes(f, q_f)
[8]     end if
[9]     else // f is a linear or nominal feature
[10]       for each class c do
[11]          feature_vote[f, c] ← segment_class_vote[f, q_f, c]
[12]       end for
[13]    end else
[14]    if use_certainty_on_single_feature_prediction = "true" then
[15]       if feature_vote[f, argmax_c(feature_vote[f, c])] ≥ MinC_v then
[16]          for each class c do
[17]             feature_vote[f, c] ← feature_vote[f, c] * feature_weight[f]
[18]             final_vote[c] ← final_vote[c] + feature_vote[f, c]
[19]          end for
[20]       end if
[21]    end if
[22]    else
[23]       for each class c do
[24]          feature_vote[f, c] ← feature_vote[f, c] * feature_weight[f]
[25]          final_vote[c] ← final_vote[c] + feature_vote[f, c]
[26]       end for
[27]    end else
[28] end for
[29] PredictClassAndCalculatePredictionCertainty(finalvote)
[30] return (prediction, certainty)
```

Figure 3.5: The BMCVFP$_{query}$ Algorithm

V_1 and V_2 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 $V2$:

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

Finally, the similarity between the ordered pairs $V1$ and $V2$ is $Similarity(V1, V2) = \frac{2}{5}.\frac{1}{3} = 0.13$

Another example showing the computation of similarity between two ordered pairs of sets V_3 and V_4 is as follows:

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

The similarity between V_3 and V_4 is computed as the multiplication of left and right hand side set similarities.

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

$$set1 \cap set3 = \{item2, item4\} \text{ and } set2 \cap set4 = \emptyset$$
$$Set\_similarity(set1, set3) = min(\tfrac{2}{4}, \tfrac{2}{5}) = \tfrac{2}{5}$$
$$Set\_similarity(set2, set4) = 1$$

Again, the similarity between the ordered pairs $V3$ and $V4$ is $Similarity(V3, V4) = \frac{2}{5}.1 = 0.40$

Algorithm: CalculateOrderedPairofSetsTypeFeatureVotes(f, q_f)
Input:
Feature, f; the query instance q's value on feature f, q_f
Output:
Vote distribution of feature f among classes for query instance q, $feature\_vote[f, c]$ $(\forall c)$
Method:

```
[1]  for each class c do
[2]      feature_vote[f, c] ← 0
[3]  end for
[4]  sum_sim ← 0
[5]  for each point segment p on feature f do
[6]      sim ← Similarity(q_f, p)
[7]      if sim = 1 then
[8]          for each class c do
[9]              feature_vote[f, c] ← segment_class_vote[f, p, c]
[10]         end for
[11]         return feature_vote[f, c] (∀c)
[12]     end if
[13]     else
[14]         for each class c do
[15]             feature_vote[f, c] ← feature_vote[f, c] + segment_class_vote[f, p, c] * sim
[16]             sum_sim ← sum_sim + sim
[17]         end for
[18]     end else
[19] end for
[20] for each class c do
[21]     feature_vote[f, c] ⟵ feature_vote[f,c] / sum_sim
[22] end for
[23] return feature_vote[f, c] (∀c)
```

Figure 3.6: CalculateOrderedPairofSetsTypeFeatureVotes(f,q_f) Algorithm

Algorithm: PredictClassAndCalculatePredictionCertainty($classvote$)
Input:
Array of the class votes, $classvote$
Output:
Class that takes the highest vote and the certainty value of this prediction, $(prediction, certainty)$
Method:

[1] $prediction \leftarrow \underset{c}{argmax}(classvote[c])$
[2] if $classvote[prediction] = 0$ then
    // All classes had a vote of 0, *prediction* and its *certainty* are taken as -1
[3] $prediction \leftarrow -1$
[4] $certainty \leftarrow -1$
[5] end if
[6] else
[7] $certainty \leftarrow \frac{classvote[prediction]}{\sum_c classvote[c]}$
[8] end else
[9] return $(prediction, certainty)$

Figure 3.7: PredictClassAndCalculatePredictionCertainty Algorithm

Chapter 4

A Benefit-Maximizing, Interactive Rule Interestingness Learning Algorithm

This chapter is devoted to modeling interestingness of streaming association rules as a benefit-maximizing classification problem. "Benefit-Maximizing, Interactive Rule Interestingness Learning" *(BM_IRIL)* algorithm is proposed for this purpose.

The chapter is organized as follows. Section 4.1 investigates the reasons that motivate us to model interestingness concept as a classification problem. Section 4.2 gives information about factors influencing the interestingness of an association rule and explains feature representations of these factors in the classification problem. Section 4.3 introduces *BM_IRIL* algorithm in the big picture. Section 4.4 shows the algorithmic details of *BM_IRIL*, comprehensively.

4.1 Motivation for Modeling Interestingness Concept as a Classification Problem

The interestingness issue has been an important problem ever since the beginning of data mining research [17]. There are many factors contributing to the interestingness of a discovered pattern [17, 45, 62]. These factors are usually grouped as objective factors and subjective factors. 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 [37, 41, 42, 44, 55, 63] or objective [2, 6, 14, 18, 19, 32]. 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 another objective interestingness measure xy^2. An objective interestingness factor can also be used as an objective interestingness measure alone (e.g., confidence) [47, 67]. 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 by Zhao et al. in [72] 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 [41] or general impressions [42, 44, 63]. 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 [37] 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 [39, 49, 60] or after [2, 6, 14, 18, 19, 32, 37, 41, 42, 44, 55, 63] 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 thesis, 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 [41]. It may not be suitable on some domains. Authors in [33] 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, a 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 of induced rules 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, the proposed system should learn a subjective model for the interestingness concept description of the induced rules. The interaction with the user ensures this subjectivity.

In this study, we primarily work with association rules and think of a domain from which transactions and association rules induced from these transactions are gathered at varying periods. Learning a subjective model for the interestingness concept description of these so-called streaming association rules is important. The proposed system, "Benefit-Maximizing Interactive Rule Interestingness Learning" *(BM_IRIL)* algorithm, formulates the interestingness concept of these streaming association rules as a classification problem and learns a different interestingness model for each user. *BM_IRIL* is a post-processing system that works in an incremental manner and employs user interactivity at a certain level.

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. 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 one of the values of "interesting" or "uninteresting", and these values are initially unknown for each rule.

4.2 Modeling Interestingness Concept of Association Rules 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 $T \subseteq I$. An association rule R is an implication of the form $A \rightarrow B$, where $A \subseteq I$,

$B \subseteq I$ and $A \cap B = \emptyset$, 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 [29] 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 [47, 67]. For instance, support and confidence can alone be used as objective interestingness measures [47, 67].

We use confidence, coverage, strength, and size of the rules among the determining features in modeling the interestingness of streaming association

Linear Feature	Short Description or Formula
Confidence	$\frac{m(A \cup B)}{m(A)}$
Coverage	$\frac{m(A)}{N}$
Strength	$\frac{m(A \cup B) * N}{m(A) * m(B)}$
Size	$\|A\| + \|B\|$

Table 4.1: Linear features and formulas

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 4.1, 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 in order to ensure that all objective determining features are independent of each other ($support = confidence * coverage$).

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. Confidence, coverage, strength, and size features are linear valued features, whereas the three new features are ordered pair of sets type features.

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). Let us explain this by an example:

Let $R1 : item1, item2, item3, item4 \rightarrow item6, item7, item9$ be an association rule induced in a domain.

The *both sides* feature value corresponding to $R1$ is:

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

The *left hand side* feature value corresponding to $R1$ is:

$$V2 = (\{item1, item2, item3, item4\}, \emptyset)$$

The *right hand side* feature value corresponding to $R1$ is:

$$V3 = (\emptyset, \{item6, item7, item9\})$$

These three ordered pair of sets type features 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 representation of the association rule $R : A \rightarrow B$ is actually the vector $< confidence_R, coverage_R, strength_R, size_R, (A, \emptyset), (\emptyset, B), (A, B), interestingness\_label_R >$. The vector consists of

Feature Name	Feature Type	Feature Value
Confidence	Linear	$confidence_R$
Coverage	Linear	$coverage_R$
Strength	Linear	$strength_R$
Size	Linear	$size_R$
Left hand side	Ordered pair of sets	$(A, \emptyset)$
Right hand side	Ordered pair of sets	$(\emptyset, B)$
Both sides	Ordered pair of sets	(A, B)

Table 4.2: Feature name, type and values for the query instance representation of a particular association rule $R : A \rightarrow B$

seven determining features and one target feature, the *interestingness_label*. Determining feature name, type, and values for the query instance representation of the association rule $R : A \rightarrow B$ are summarized in Table 4.2. The interestingness of R, $interestingness\_label_R$, is predicted by the determining features.

4.3 BM_IRIL in the Big Picture

BM_IRIL, whose schematic form is shown in Fig.1.1, aims to classify the streaming association rules automatically with sufficient certainty. It consults the expert of the domain in case a rule cannot be classified with sufficient certainty. The number of times the expert is consulted should be minimized to achieve a limited user interactivity.

In situations where unlabeled data is abundant but labeling data is expensive, the learning algorithm can actively query the user/expert for labels. In the literature, this type of supervised learning is called active learning [40]. In this respect, *BM_IRIL* approach can also be considered as an active learning

approach.

BM_IRIL takes the association rule set stream and the certainty threshold value ($MinCv$) 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 [29]. The output of the *BM_IRIL* algorithm is the association rules classified as "interesting" or "uninteresting", with sufficient certainty at each period. The user can easily filter the rules classified as interesting among the rules output. The classification process continues as long as the transaction set stream is supplied to the system.

BM_IRIL employs a core classification algorithm inside. The unexpectedness and actionability interestingness factors are represented by ordered pair of sets type features. Therefore, the core classification algorithm should also handle this type of features. Consequently, we designed a suitable classifier, namely "Benefit-Maximizing Classifier by Voting Feature Projections" (*BMCVFP*) to handle this type of features, too. *BMCVFP* is an incremental, feature projections based, benefit-maximizing classification algorithm. Chapter 3 is devoted to this algorithm.

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 employed a benefit-maximizing classification for learning the interestingness classification of the rules.

In *BM_IRIL* algorithm, 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, *BM_IRIL* consults 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 for *BMCVFP* and the interestingness concept description (interestingness model) is updated incrementally.

We assume that the interest of a human expert depends on a selective subset of the objective interestingness factors, including the rule's content itself. But,

in the literature, they seek to find a correlation between real human interest and objective interestingness measures [9, 50, 51, 52].

We proposed to model interestingness of patterns as a classification problem in [3]. To the best of our knowledge, none of the existing approaches in the literature had tried to model interestingness concept as a classification problem. The "Feature Projection Based Rule Classification" *(FPRC)* algorithm in [3] used a non-incremental classifier. In order to handle the case of streaming rules to be classified, the "Interactive Rule Interestingness Learning" *(IRIL)* algorithm, that used an incremental classifier, has been developed [4]. Both *FPRC* and *IRIL* are applicable to learning the interestingness of classification rules, while they are not suitable for association rules.

The classification rules used in the study explained in [4] 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 ($Class_1$: $vote_1$, $Class_2$: $vote_2$, ..., $Class_k$: $vote_k$)

where A_i's are the features, $Class_i$'s are the classes and op $\in \{=, \neq, <, >, \leq, \geq\}$.

Determining feature name, type, and short descriptions for the query instance representation of a classification rule R are summarized in Table 4.3.

The *BM_IRIL* proposed here is designed for learning the interestingness concept of association rules. Furthermore, it takes into account the benefit concept while classifying the association rules. Subjective interestingness factors such as unexpectedness and actionability are incorporated into the vector representation of query rules, for the first time. The core classifier is an incremental one as in *IRIL*.

BM_IRIL also proposes a new feature weighting technique that takes benefit maximization issues into account. Feature weights are not externally supplied to the algorithm. They are updated upon arrival of each training rule.

Feature Name	Feature Type	Feature Description
Major Class	Nominal	Class taking the highest vote
Confidence	Linear	$confidence_R$
Coverage	Linear	$coverage_R$
Completeness	Linear	$completeness_R$
Size	Linear	$size_R$
Zero Voted Class Count	Linear	Number of classes with zero vote
Vote Std. Dev.	Linear	Standard deviation of the votes given to the classes

Table 4.3: Feature name, type, and short descriptions for the query instance representation of a particular classification rule R

To summarize, *BM_IRIL* is a benefit-maximizing, interactive and incremental rule interestingness learning algorithm. It models the interestingness of streaming 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.

The explained contributions of the proposed interestingness concept learning system make *BM_IRIL* a novel approach in the literature.

4.4 BM_IRIL Algorithm

BM_IRIL algorithm is shown in Fig. 4.1. In a particular period p, it takes the input parameters $MinC_v$, R_p (Association rules induced from the transactions gathered at period p), *use_instant_concept_update*, *use_feature_weighting* and *use_certainty_on_single_feature_prediction* to work.

Algorithm: BM_IRIL($R_p, use\_certainty\_on\_single\_feature\_prediction, MinC_v$, $use\_feature\_weighting, use\_instant\_concept\_update$)
Input:
Rules induced at period p, R_p;
Whether minimum certainty requirement will be met on the predictions of each feature, *use_certainty_on_single_feature_prediction*;
Minimum Certainty Value, $MinC_v$;
Whether feature weighting will be used in the querying process, *use_feature_weighting*;
Whether instant interestingness concept update will be employed upon each classification of a rule by the user, *use_instant_concept_update*
Output:
Set of rules classified as interesting with sufficient certainty
Method:

```
[1]   R_up ← Ø // Set of rules classified by the user at period p
[2]   R_sp ← Ø // Set of rules classified with sufficient certainty by BMCVFP at period p
[3]   for each rule r ∈ R_p do
[4]     BMCVFP_query(r, use_certainty_on_single_feature_prediction, MinC_v)
[5]     if certainty ≥ MinC_v then
[6]       insert r into R_sp
[7]     end if
[8]     else
[9]       ask the user to classify r and set certainty of this classification to 100%
[10]      insert r into R_up
[11]      if use_instant_concept_update = "true" then
[12]        insert r into R_t and BMCVFP_train(r)
[13]        if use_feature_weighting = "true" then
[14]          for each feature f do
[15]            UpdateFeatureWeight(f, r)
[16]          end for
[17]        end if
[18]      end if
[19]    end else
[20]  end for
[21]  if use_instant_concept_update = "false" then
[22]    for each rule r ∈ R_up do
[23]      insert r into R_t and BMCVFP_train(r)
[24]      if use_feature_weighting = "true" then
[25]        for each feature f do
[26]          UpdateFeatureWeight(f, r)
[27]        end for
[28]      end if
[29]    end for
[30]  end if
[31]  return the rules classified as interesting among R_sp
```

Figure 4.1: The BM_IRIL Algorithm

BM_IRIL algorithm regards each input association rule having the form $R : A \rightarrow B$ as a query instance and represents the query instance by a vector $< confidence_R, coverage_R, strength_R, size_R, (A, \emptyset), (\emptyset, B), (A, B), ? >$. 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 has been explained in Section 4.2. 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 run 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 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 set the values of *use_instant_concept_update*, *use_feature_weighting* and *use_certainty_on_single_feature_prediction* parameters as "true" by default, and of course, have the flexibility to adjust them before the execution of the *BM_IRIL* algorithm.

If *use_instant_concept_update* parameter is "true", a query rule r classified manually by the user is inserted into R_{up} and R_t at the same time. 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 "false", the rules classified manually by the user are only inserted 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 *use_feature_weighting* parameter is "true", feature weights are updated each time an association rule r is inserted into R_t and the interestingness model is updated. *UpdateFeatureWeight* algorithm, shown in Fig. 4.2, tells us how to update the weight of a feature f. *UpdateFeatureWeight* algorithm makes use of Eq. 4.1 (line 15).

$$feature\_weight[f] = \frac{\sum_c (corr\_pred\_tr\_cnt[f,c]B[c,c])}{\sum_c (classcount[c]B[c,c])} \tag{4.1}$$

In Eq. 4.1, $B[c,c]$ 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 learned interestingness model on feature projection f with $certainty \geq MinC_v$ so far.

Algorithm: UpdateFeatureWeight(f, t)
Input:
Feature, f; training instance, t
Output:
Updated weight of f, $feature\_weight[f]$
Method:
[1] let t_f be the feature value of t on f
[2] let t_c be the class of t
[3] if f is an ordered pair of sets type feature then
[4] $CalculateOrderedPairofSetsTypeFeatureVotes(f, t_f)$
[5] end if
[6] else // f is a linear or nominal feature
[7] for each class c do
[8] $feature\_vote[f, c] \leftarrow segment\_class\_vote[f, t_f, c]$
[9] end for
[10] end else
[11] $prediction \leftarrow \underset{c}{argmax}(feature\_vote[f, c])$
[12] if $(feature\_vote[f, prediction] \geq MinC_v)$ and $(prediction = t_c)$ then
[13] increment $corr\_pred\_tr\_cnt[f, prediction]$
[14] end if
[15] $feature\_weight[f] = \frac{\sum_c (corr\_pred\_tr\_cnt[f,c]B[c,c])}{\sum_c (classcount[c]B[c,c])}$
[16] return $feature\_weight[f]$

Figure 4.2: The UpdateFeatureWeight Algorithm

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:

- To classify most of the input association rules automatically with sufficient certainty and to keep the user participation low
- To keep the benefit accuracy of the classifications high

Experimental results in Chapter 5 show that we achieve these goals.

Chapter 5

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 [29] 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, for our experiments.

Table 5.1 gives the classification distribution statistics of the association rules between the domain expert and the *BM_IRIL* system for the $MinC_v$ value of 70%. Columns 3 and 4 of Table 5.1 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:

$$BenefitAccuracy_p = \frac{\sum_c (corr\_pred\_cnt[p,c]B[c,c])}{\sum_c (pred\_cnt[p,c]B[c,c])} \tag{5.1}$$

At each period p, all the induced association rules are regarded as query rules and are tried to be classified by *BMCVFP*. In Eq. 5.1, $B[c,c]$ 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 $certainty \geq MinC_v$.

BM_IRIL attempted to classify a total of 1263 association rules, presented along with objective interestingness factor values, with sufficient certainty. These 1263 rules were induced in 25 weeks. The distribution of these rules among 25 weeks, the number of interesting and uninteresting rules in each week and the classification distribution statistics of rules between user and the *BM_IRIL* system at $MinC_v = 70\%$ for each week is given in Table 5.1. It is clear 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 a phony high benefit accuracy on the remaining small number of rules. On the other hand, it is also 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 these two success criteria.

$$Performance = BenefitAccuracy * (1 - UserParticipation) \tag{5.2}$$

where, user participation is the proportion of examples in the period that the user has labeled. Table 5.2 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 accuracy values 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 thesis, *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 will be willing to classify each rule by brute force. Even if the number of rules is small, the user should not be expected to label each rule one by one. Otherwise, there would not be a need for a system modeling the interestingness concept. The user should be consulted for only a small percentage of rules. This is what *BM_IRIL* actually achieves. User participation is kept at very low levels.

Table 5.3 displays 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 the *Naive Bayesian* classifier as the core classifier in *BM_IRIL* and compared it against the *BM_IRIL* system employing *BMCVFP* as the core classifier inside. The *Naive 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. 3.4. The comparison results provided in Table 5.4 show that the *BMCVFP* classifier performs better than the classical *Naive Bayesian* classifier.

In this work, we also defined and analyzed the *use_feature_weighting*, *use_certainty_on_single_feature_prediction*, and *use_instant_concept_update* parameters. We took them as "true" by default in our experiments. Results in Tables 5.5, 5.6, and 5.7 empirically prove that *use_instant_concept_update* and *use_certainty_on_single_feature_prediction* parameters should be set as "true" whereas *use_feature_weighting* parameter is free to set for the optimum performance of the *BM_IRIL* system.

We used *Wilcoxon Signed Ranks test*, at $\alpha = 0.05$ significance level, to show the differences were actually significant in Tables 5.4, 5.5, 5.6, and 5.7 for the comparison criteria. *The Wilcoxon Signed Ranks test* is designed to test a hypothesis about the location (median) of a population distribution. It often involves the use of matched pairs, for example, before and after data, in which case it tests for a median difference of zero. In our case, if we think of *Benefit Accuracy* criterion, the null hypothesis tested is that the differences between the benefit accuracy values of two algorithms (*BM_IRIL* and the modified version) have a median value of zero (That is, the differences are not significant).

Asymp. Sig. values given in the corresponding tables are all less than 0.05 (except for Table 5.7), implying that the differences are statistically significant (except for Table 5.7). We found out that 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.

In the comparisons, unless indicated otherwise, *BM_IRIL* has the following default properties:

1. uses *BMCVFP* as the core classifier
2. meets minimum certainty requirement on the predictions of each feature
3. employs feature weighting in the querying process and
4. employs instant concept update upon each classification of a rule by the user

Period No	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 5.1: Classification distribution statistics of rules between user and the BM_IRIL system at $MinC_v = 70\%$

Period No	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%

Table 5.2: User Participation, Recall, Benefit Accuracy, and Performance values at $MinC_v = 70\%$

	Minimum Certainty Value ($MinC_v$)						
Period No	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%**

Table 5.3: Performance comparison at various minimum certainty values. *Friedman Test* employed for significance test. $Asymp.Sig. = 2.015e^{-18}$ at $\alpha = 0.05$

Period No	Comparison Criterion					
	Benefit Accuracy		User Participation		Performance	
	BM_IRIL	BM_IRIL using Naive Bayesian as the core classifier	BM_IRIL	BM_IRIL using Naive Bayesian as the core classifier	BM_IRIL	BM_IRIL using Naive 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%

Table 5.4: Comparison of *BM_IRIL* against *BM_IRIL* using the *Naive Bayesian* classifier as the core classifier at $MinC_v = 70\%$. *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 $\alpha = 0.05$

	Comparison Criterion					
	Benefit Accuracy		User Participation		Performance	
Period No	BM_IRIL	BM_IRIL that does not use certainty on single feature prediction	BM_IRIL	BM_IRIL that does not use certainty on single feature prediction	BM_IRIL	BM_IRIL that does not use certainty on single feature prediction
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%

Table 5.5: Comparison of *BM_IRIL* against *BM_IRIL* that does not use certainty on single feature prediction. *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 $\alpha = 0.05$

	Comparison Criterion					
	Benefit Accuracy		User Participation		Performance	
Period No	BM_IRIL	BM_IRIL that does not use instant concept update	BM_IRIL	BM_IRIL that does not use instant concept update	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%

Table 5.6: Comparison of *BM_IRIL* against *BM_IRIL* that does not use instant concept update. *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 $\alpha = 0.05$

	Comparison Criterion					
	Benefit Accuracy		User Participation		Performance	
Period No	BM_IRIL	BM_IRIL that does not use feature weighting	BM_IRIL	BM_IRIL that does not use feature weighting	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%

Table 5.7: Comparison of *BM_IRIL* against *BM_IRIL* that does not use feature weighting. *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 $\alpha = 0.05$

Chapter 6

Conclusions

Association rules are among the important pattern types and employed today in many application areas including web usage mining, intrusion detection, filtering, screening, and bioinformatics.

A common application domain of association rules is sales data, known as basket data. 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 previous month. For this reason, it was worthwhile to consider the problem where transaction sets are input to rule learning algorithms as a stream of packages.

In this thesis, we dealt with the interestingness issue of association rules discovered in domains from which information in the form of transactions is gathered at different time intervals.

The sets of transactions may come in varying sizes and in varying periods. Once a set of transactions arrives, the association rule learning algorithm is run on the last set of transactions, resulting in a new set of 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.

There are objective and subjective interestingness measures in the literature. A particular objective interestingness measure is not sufficient by itself [41]. It may not be suitable on some domains. It 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 of induced rules 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 of rules on a given domain usually differs from one expert to the other and also over time for a given expert. Therefore, the proposed system should have learned a subjective model for the interestingness concept description of the induced rules.

In this thesis, we worked on streaming association rules and proposed "Benefit-Maximizing Interactive Rule Interestingness Learning" *(BM_IRIL)* algorithm to deal with the interestingness issue of these 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. Unexpectedness and actionability interestingness factors of association rules are handled by taking the rule's content into account. We assume that the interest of a human expert depends on these selected features.

We defined a new feature type, namely ordered pair of sets, to represent the unexpectedness and actionability interestingness factors. Therefore, the core classification algorithm that *BM_IRIL* employed inside should have also handled this type of features. Consequently, we designed a suitable classifier, namely "Benefit-Maximizing Classifier by Voting Feature Projections" (*BM-CVFP*) to handle this type of features, too.

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 employed a benefit-maximizing core classification learning algorithm.

The proposed method, *BM_IRIL*, can work on association rules induced by any association rule-mining algorithm. It works incrementally and employs user interactivity at a certain level. Therefore, it can also be considered as an active learning approach.

BM_IRIL also proposes a new feature weighting technique that takes benefit maximization issues into account. Feature weights are not externally supplied to the algorithm. They are updated upon arrival of each training rule.

Besides *BMCVFP*, we also used the *Naive Bayesian* classifier as the core classifier in *BM_IRIL* and compared two core classifiers. The *Naive 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 *Naive Bayesian* classifier.

In this work, we also defined and analyzed the *use_feature_weighting*, *use_certainty_on_single_feature_prediction*, and *use_instant_concept_update* parameters. We took them as "true" by default in our experiments. Results empirically prove that *use_certainty_on_single_feature_prediction*, and *use_instant_concept_update* parameters should be set as "true" whereas *use_feature_weighting* parameter is free to set for the optimum performance of the *BM_IRIL* system.

BM_IRIL was evaluated on a real supermarket dataset. We recorded the customer transactions for 25 weeks. Each week is taken as the unit of period and has its own set of transactions. Following this, we induced association rules from the set of transactions for each period. *BM_IRIL* tried to classify the association rules with high accuracy and low user percentage. 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 one of the most interesting rules for the user analyzing the domain.

The contributions of the proposed interestingness concept learning system make *BM_IRIL* a novel approach in the literature. As a future work, unexpectedness and actionability of classification rules may somehow be modeled and *BM_IRIL* may be adopted to also work with classification rules, rather than just association rules. As another future work, a similar approach can be implemented on clusters. Factors related to the interestingness of clusters can be analyzed and used as determining features in a similar interestingness-learning algorithm. *BM_IRIL* was evaluated on a supermarket domain where transactions consist of the sales of the customers. The proposed framework

can also be tested on different application areas, such as web usage mining, intrusion detection, filtering, screening, and bioinformatics.

Appendix A

Sample Association Rules Induced from a Supermarket Transaction Set

1) Lipton Yellow Label Tea AND Balküpü Sugar → Bread AND Pınar Cheese

2) Tomato AND Cucumber AND Potato → Lemon AND Onion

3) Akmina Soda AND Roasted Chickbea AND Coca Cola → Hazelnut

4) Pınar Milk AND Egg → Bread AND Milliyet Newspaper

5) Meat AND Bağdat Spices → Tat Canned Food

6) Whiskas 85gr Lamb Kidney → Whiskas 85gr Beef Lamb

7) Nestle Crunch AND Balparmak Honey AND Pınar Cheese → Pınar Butter

8) Knorr Ready Soup → Bread AND Lemon

9) Nestle Coffee → Nestle Coffee Mate

Bibliography

[1] R. Agrawal, H. Mannila, R. Srikant, R. Toivonen, and A.I. Verkamo. Fast discovery of association rules. *Advances in Knowledge Discovery and Data Mining*, pages 307–328, 1996.

[2] A.S. Al-Hegami, V. Bhatnagar, and N. Kumar. Novelty framework for knowledge discovery in databases. In Y. Kambayashi et al., editor, *Proceedings of DaWak 2004*, volume 3181 of *LNCS*, pages 48–57. Springer-Verlag, 2004.

[3] T. Aydin and H.A. Güvenir. Feature projection based rule classification. In *Proceedings of Twelfth Turkish Symposium on Artificial Intelligence and Neural Networks (TAINN'2003), Çanakkale*, pages 652–661, July 2–4 2003.

[4] T. Aydin and H.A. Güvenir. Learning interestingness of streaming classification rules. In Cevdet Aykanat, Tugrul Dayar, and Ibrahim Korpeoglu, editors, *Proceedings of ISCIS 2004*, volume 3280 of *LNCS*, pages 62–71. Springer-Verlag, 2004.

[5] R. Bayardo and R. Agrawal. Mining the most interesting rules. In *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 145–154, 1999.

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

[7] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and regression trees*. Belmont: Wadsworth, 1984.

[8] C.H. Cai, W.C. Fu, C.H. Cheng, and W.W. Kwong. Mining association rules with weighted items. In *Proceedings of IEEE International Database Engineering and Applications Symposium*, pages 68–77, 1998.

[9] D.R. Carvalho, A.A. Freitas, and N. Ebecken. Evaluating the correlation between objective rule interestingness measures and real human interest. In A. Jorge et al., editor, *Proceedings of PKDD 2005*, volume 3721 of *LNAI*, pages 453–461. Springer-Verlag, 2005.

[10] S. Chakrabarti, S. Sarawagi, and B. Dom. Mining surprising patterns using temporal description length. In *Proceedings of the 24th International Conference on VLDB*, pages 606–617, 1998.

[11] P. Clark and S. Matwin. Using qualitative models to guide induction learning. In *Proceedings of the International Conference on Machine Learning*, pages 49–56, 1993.

[12] E. Cohen, M. Datar, S. Fujiwara, A. Gionis, P. Indyk, R. Motwani, J.D. Ullman, and C. Yang. Finding interesting associations without support pruning. Technical report, Department of Computer Science, Stanford University, 2001.

[13] G. Demiröz and H.A. Güvenir. Classification by voting feature intervals. In Maarten van Someren and Gerhard Widmer, editors, *Proceedings of 9th European Conference on Machine Learning*, volume 1224 of *LNAI*, pages 85–92. Springer-Verlag, April 23–25 1997.

[14] G. Dong and J. Li. Interestingness of discovered association rules in terms of neighborhood-based unexpectedness. In *Proceedings of the 2nd Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 72–86, 1998.

[15] M. Ester, H.P. Kriegel, J. Sander, and X. Xu. A density based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, pages 226–231, 1996.

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

[17] W.J. Frawely, G. Piatetsky-Shapiro, and C.J. Matheus. Knowledge discovery in databases: an overview. *Knowledge Discovery in Databases*, pages 1–27, 1991.

[18] A.A. Freitas. On objective measures of rule surprisingness. In *Proceedings of the Second European Conference on the Principles of Data Mining and Knowledge Discovery (PKDD'98)*, pages 1–9, 1998.

[19] A.A. Freitas. On rule interestingness measures. *Knowledge Based Systems*, 12:309–315, 1999.

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

[21] H.A. Güvenir, S. Altıngövde, I. Uysal, and E. Erel. Bankruptcy prediction using feature projection based classification. In *Proceedings of SCI/ISAS'99, Orlando, Florida*, pages 108–113, July 31–August 4 1999.

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

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

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

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

[26] H.A. Güvenir and I. Sirin. Classification by feature partitioning. *Machine Learning*, 23(1):47–67, 1996.

[27] R.J. Hilderman and H.J. Hamilton. Heuristic measures of interestingness. In *Proceedings of the 3rd European Conference on the Principles of Data Mining and Knowledge Discovery*, pages 232–241, 1999.

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

[29] http://fuzzy.cs.uni-magdeburg.de/ borgelt/apriori.html.

[30] http://kiew.cs.uni-dortmund.de:8001/mlnet/instances/81d91eaa-da13fe98e4.

[31] http://mathworld.wolfram.com/k-meansclusteringalgorithm.html.

[32] F. Hussain, H. Liu, E. Suzuki, and H. Lu. Exception rule mining with a relative interestingness measure. In *Proceedings of Pacific-Asia Conf. on Knowledge Discovery and Data Mining*, pages 86–97, 2000.

[33] X. Huynh, F. Guillet, and H. Briad. A data analysis approach for evaluating the behaviour of interestingness measures. In A. Hoffman, H. Motoda, and T. Scheffer, editors, *Proceedings of DS 2005*, volume 3735 of *LNAI*, pages 330–337. Springer-Verlag, 2005.

[34] N. Ikizler. Benefit maximizing classification using feature intervals. Master's thesis, Department of Computer Engineering, Bilkent University, 2002.

[35] N. Ikizler and H.A. Güvenir. Maximizing benefit of classifications using feature intervals. In Vasile Palade, Robert J. Howlett, and Lakhmi Jain, editors, *Proceedings of 7th International Conference on Knowledge-Based Intelligent Information & Engineering Systems (KES'2003), Oxford, United Kingdom*, volume 2773 of *LNAI*, pages 339–345. Springer-Verlag, Sept. 3-5 2003.

[36] M. Kawakita, M. Minami, S. Eguchi, and C.E. Lennert-Cody. An introduction to the predictive technique adaboost with a comparison to generalized additive models. *Fisheries Research*, 76:328–343, 2005.

[37] M. Klemettinen, H. Mannila, P. Ronkainen, H. Toivonen, and A.I. Verkamo. Finding interesting rules from large sets of discovered association rules. In *Proceedings of the 3rd International Conference on Information and Knowledge Management*, pages 401–407, 1994.

[38] Y. Ko and J. Seo. Text categorization using feature projections. In *Proceedings of the 19th International Conference on Computational Linguistics, Taipei, Taiwan*, pages 1–7, 2002.

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

[40] D.D. Lewis and W.A. Gale. A sequential algorithm for training text classifiers. In *Proceedings of the 17th annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3–12, 1994.

[41] B. Liu and W. Hsu. Post-analysis of learned rules. In *Proceedings of AAAI 1996*, pages 828–834, 1996.

[42] B. Liu, W. Hsu, and S. Chen. Using general impressions to analyze discovered classification rules. In *Proceedings of the 3rd International Conference on KDD*, pages 31–36, 1997.

[43] B. Liu, W. Hsu, H. Lee, and L. Mun. Tuple-level analysis for identification of interesting patterns. Technical report, National University of Singapore, 1996.

[44] B. Liu, H. Wynne, S. Chen, and Y. Ma. Analyzing the subjective interestingness of association rules. *IEEE Expert Intelligent Systems & Their Applications*, 15(5):184–192, 2000.

[45] J.A. Major and J.J. Mangano. Selecting among rules induced from a hurricane database. In *Proceedings of AAAI Workshop on Knowledge Discovery in Databases*, pages 30–31, 1993.

[46] D.D. Margineantu. *Methods for cost-sensitive learning.* PhD thesis, Oregon State University, 2002.

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

[48] R.S. Michalski, I. Mozetic, J. Hong, and N. Lavrac. The multi-purpose incremental learning system aq15 and its testing application to three medical domains. In *Proceedings of the 5th National Conference On Artificial Intelligence*, pages 1041–1045, 1986.

[49] E. Noda, A.A. Freitas, and H.S. Lopes. Discovering interesting prediction rules with a genetic algorithm. In *Proceedings of CEC-99*, 1999.

[50] M. Ohsaki, S. Kitaguchi, K. Okamoto, H. Yokoi, and T. Yamaguchi. Evaluation of rule interestingness measures with a clinical dataset on hepatitis. In J.F. Boulicaut et al., editor, *Proceedings of PKDD 2004*, volume 3202 of *LNAI*, pages 362–373. Springer-Verlag, 2004.

[51] M. Ohsaki, S. Kitaguchi, H. Yokoi, and T. Yamaguchi. Investigation of rule interestingness in medical data mining. In S. Tsumoto et al., editor, *Proceedings of AM 2003*, volume 3430 of *LNAI*, pages 174–189. Springer-Verlag, 2005.

[52] M. Ohsaki, Y. Sato, S. Kitaguchi, H. Yokoi, and T. Yamaguchi. Comparison between objective interestingness measures and real human interest in medical data mining. In R. Orchard et al., editor, *Proceedings of IEA/AIE 2004*, volume 3029 of *LNAI*, pages 1072–1081. Springer-Verlag, 2004.

[53] J. Ortega and D. Fisher. Flexibly exploiting prior knowledge in empirical learning. In *Proceeding of International Conference on Machine Learning*, 1995.

[54] B. Padmanabhan and A. Tuzhilin. A belief-driven method for discovering unexpected patterns. In *Proceedings of KDD 1998 Workshop*, pages 74–100, 1998.

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

[56] C. Pateritsas and A. Stafylopatis. A nearest features classifier using a self-organizing map for memory base evaluation. In S. Kollias et al., editor, *Proceedings of ICANN 2006*, volume 4132 of *LNCS*, pages 391–400. Springer-Verlag, 2006.

[57] M. Pazzani and D. Kibler. The utility of knowledge in inductive learning. *Machine Learning*, 9:57–94, 1992.

[58] J.R. Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986.

[59] J.R. Quinlan. *C4.5: program for machine learning*. Morgan Kaufmann, 1992.

[60] I. Rahal, D. Ren, A. Perera, H. Najadat, W. Perrizo, R. Rahhal, and W. Valdivia. Incremental interactive mining of constrained association rules from biological annotation data with nominal fetaures. In *Proceedings of the 2005 ACM Symposium on Applied Computing*, pages 123–127, 2005.

[61] S. Sahar. Interestingness via what is not interesting. In *Proceedings of the 5th International Conference on Knowledge Discovery and Data Mining*, pages 332–336, 1999.

[62] G.P. Shapiro and C.J. Matheus. The interestingness of deviations. In *Proceedings of AAAI Workshop on Knowledge Discovery in Databases*, pages 25–36, 1994.

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

[64] A. Silberschatz and A. Tuzhilin. What makes patterns interesting in knowledge discovery systems. *IEEE Transactions on Knowledge and Data Engineering*, 8(6):375–398, 1996.

[65] I. Sirin and H.A. Güvenir. Empirical evaluation of the cfp algorithm. In C. Rowles, H. Liu, and N. Foo, editors, *Proceedings of the Sixth Australian Joint Conference on Artificial Intelligence, Melbourne, Australia*, pages 311–315, November 1993.

[66] R. Srikant, Q. Vu, and R. Agrawal. Mining association rules with item constraints. In *Proceedings of KDD 1997 Workshop*, pages 67–73, 1997.

[67] P.N. Tan and V. Kumar. Interestingness measures for association patterns: A perspective. In *Proceedings of KDD 2000 Workshop on Post-processing in Machine Learning and Data Mining*, 2000.

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

[69] V. Valev. Supervised pattern recognition by parallel feature partitioning. *Pattern Recognition*, 37:463–467, 2004.

[70] H. Watanabe, M. Arai, and K. Okuda. Batch mode algorithms of classification by feature partitioning. *IEICE Trans. Inf. & Systems*, E81-D(1), 1998.

[71] J. Zhang and R.S. Michalski. An integration of rule induction and exemplar-based learning for graded concepts. *Machine Learning*, 21:235–267, 1995.

[72] Y. Zhao, C. Zhang, and S. Zhang. Discovering interesting association rules by clustering. In G.I. Webb and Xinghuo Yu, editors, *Proceedings of AI 2004*, volume 3339 of *LNAI*, pages 1055–1061. Springer-Verlag, 2004.

Publications Done During This Thesis Work

- T. Aydin and H.A. Güvenir. Feature projection based rule classification. In *Proceedings of Twelfth Turkish Symposium on Artificial Intelligence and Neural Networks (TAINN'2003), Çanakkale, Turkey*, pages 652-661, July 2-4 2003.

- T. Aydin and H.A. Güvenir. Learning interestingness of streaming classification rules. In Cevdet Aykanat, Tugrul Dayar, and Ibrahim Korpeoglu, editors, *Proceedings of ISCIS 2004,* volume 3280 of *LNCS*, pages 62-71. Springer-Verlag, 2004.

- T. Aydin and H.A. Güvenir. Modeling interestingness of streaming classification rules as a classification problem. In F.A. Savaci, editor, *Proceedings of 14th Turkish Symposium on Artificial Intelligence and Neural Networks (TAINN'05), Izmir, Turkey*, volume 3949 of *LNAI*, pages 168-176. Springer-Verlag, 2006.

- T. Aydin and H.A. Güvenir. A novel hybrid approach for interestingness analysis of classification rules. In Akito Sakurai, Koti Hasida, and Katsumu Nitta, editors, *Proceedings of JSAI 2004, Kanazawa, Japan,* volume 3609 of *LNAI*, pages 485-496. Springer-Verlag, 2007.

- T. Aydin and H.A. Güvenir. Modeling interestingness of streaming association rules as a benefit-maximizing classification problem. *Knowledge-Based Systems,* 22(1): 85-99, 2009. (In SCI)