INTRODUCTION

The mining of association rules is performed in two stages
The discovery of frequency sets of items from the data and the generation of association rules from frequency item set. The first stage is computationally intensive.

And the second stage has the drawback of possible generating thousands of rules, thus making the analysis of the results has to perform by a human analyst.

Our contribution relate to both stages of this process. We analyze stage one and introduce two new algorithm among which algorithm Closure improves upon the performance of the classic algorithm Apriori. We define the concept of an informative cover of the set of association rules and present algorithm cover rules for computing such a cover, which is in practice one or more orders of magnitude smaller than the set of association rules.

Next, we investigate how the mining of association rules should be performed when the data reside in several tables organized in a star scheme. We show how to modify the Apriori algorithm to compute the support of item sets by analysis the star scheme table or their outer join. Whereas association rules express the dependency between item sets, purity dependencies express the dependency between sets of attributes. We introduce the concept of purity dependency as a generalization of the concept of functional dependency and we show that these purity dependencies satisfy properties that are similar to the Armstrong rules for functional dependencies. As an application of our theory we
present an algorithm for discovering purity dependencies that can be used to predict the value of a target attribute.

Data Mining as Process

Figure 1.1 presents a generic model for the DM process, with the main logical steps involved. The colored-background boxes represent steps for which original contributions boxes will be presented throughout this dissertation.

Knowledge presentation employs visualization methods to display the extracted knowledge in an intuitive, accessible and easy to understand manner. Decisions on how to proceed with future iterations are made based on the conclusions reached at this point.

Knowledge Extraction Tasks

The purpose of a data mining system is defined through the desired employment mode. According to [Fay96], two main categories of goals can be distinguished: verification and discovery. Description tasks generate models and present them to the user in an easy-to-understand form. Summarization and visualization, clustering, link analysis and outlier analysis represent descriptive tasks. Prediction tasks generate models for predicting future behavior of certain entities. Classification, regression and time series prediction are the main tasks falling in this category. A dataset is a finite set of elements, named instances, or examples:

\[ T = \{ x_i \mid i = 1 \text{ to } m \} \] (2.1)

An instance \( x_i \) is a tuple of the form \( (x(1), \ldots, x(n), y_i) \), where \( x(1), \ldots, x(n) \) represent values for the predictive attributes (features), while \( y \) is the value of the target attribute, the class:

\[ x_i = (x(1), \ldots, x(n), y_i) \] (2.2)

Classification

Classification problems try to determine the characteristics which correctly identify the class to which each instance belongs to. Thus, the scope is to learn a model from the training set which describes the class \( Y \), i.e. predict \( y \) from the values of \( (x(1), \ldots, x(n)) \). The resulting model can be employed either for descriptive, or predictive tasks. Classification is similar to clustering, the main difference being that, in classification, the class to which each instance in the dataset belongs to is known a priori.

The main categories of classification methods

Association rules

The Association rules' problem has been introduced in [AIS93b] in the context of analyzing supermarket data. We begin our discussion of this problem with an analysis of possible organization of data.
An association rule is defined as the implication \( X \rightarrow Y \), described by two interestingness measures—support and confidence—where \( X \) and \( Y \) are the sets of items and \( X \cap Y = \emptyset \).

The following is a formal statement of the problem: Let \( I = \{i_1, i_2, \ldots, i_n\} \) be a set of literals, called items. Let \( D \) be set of transactions, where each transaction is set of items. Associated with each transaction is a unique identifier, called its TID. We say that a transaction \( T \) contains \( X \), a set of some items in \( I \), if \( X \subseteq T \).

An association rule is an implication of the form \( X \rightarrow Y \), where \( X \subseteq I \), \( Y \subseteq I \) and \( X \cap Y = \emptyset \).

- Association rule \( X \rightarrow Y \) has confidence \( c \) if \( c\% \) of transactions in \( D \) that contain \( X \) also contain \( Y \).
- Association rule \( X \rightarrow Y \) has support \( s \), if \( s\% \) of transactions in \( D \) contain \( X \cup Y \).

The problem of mining association is to generate all association rules that have support and confidence greater than the user-specified minimum support and minimum confidence in a given set of transactions \( D \).

### Database consideration

The supermarket data analyzed in \cite{2} has a simple formal model. Let \( I = \{i_j, i_k, \ldots, i_n\} \) be a set whose elements are called items and let \( T = \{t_1, t_2, \ldots, t_m\} \) be a multi set whose elements are transactions, where each transaction \( t_i \) represents a subset of \( I \). The multi set \( T \) can be stored in a special table with binary attributes \( I \) and an additional transaction identifier attribute \( t_i \). Such a table is called a binary table or binary database.

<table>
<thead>
<tr>
<th>( t_i )</th>
<th>( i_1 )</th>
<th>( i_2 )</th>
<th>( i_3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

In a binary table, the presence of an item in a transaction is indicated by a value of 1 for the attribute corresponding to the item, and its absence is indicated by a 0. Thus, the cardinality of the domain of the attributes corresponding to items is two. In most tables used in practice it is infrequent to have only binary attributes. Attributes that take values in a limits domain are called nominal or categorical attributes, and attributes whose domain is large are called numerical attributes. In the case of a nominal attribute, we will extend the use of the term item to designate a particular attributes-value pair \((A, a)\). Denoting \( A = a \) (e.g. Color = red). Numerical attributes are handled by transforming them to nominal attributes through the process of dividing their domain into intervals and assigning identifies to these intervals. For example, for an attribute age we could assign the identifier teenager for the age interval 10-19, resulting in item age = teenager. This transformation numerically attributes into nominal attributes is called discretization.

### Defining Table 1.2.

We can also consider the problem of data density. Intuitively, we consider one binary database to be denser than another when its attributes take the value 1 more often. We say that a database is sparse when its rows contain a relatively small number of 1’s, otherwise we call the database dense. These notions are informal, however, as there is no formal measure for quantifying the density of a database.

#### Problem definition

We begin by defining the central concepts of the AR mining problem. For historical reasons, the terminology that was introduced in \cite{1} in the context of analyzing supermarket data has been kept intact in all successive papers published on this subject, even when the paper discussed different application.

**Definition 1.1**

Let \( I = \{i_1, i_2, \ldots, i_n\} \) be a set whose elements are called items and let \( T = \{t_1, t_2, \ldots, t_m\} \) be a multi set whose elements are called transactions. Where each transaction \( t_i \) represents \( I \). A subset of \( I \) is called itemset. An item set of cardinality \( k \) is called \( k\)-itemset. The support of an itemset \( I \) is denoted by \( \text{SUPP}(I) \) and defined as:

\[
\text{SUPP}(I) = \frac{|\{T \in T | t \supseteq I\}|}{|T|}
\]
We define $\text{SUPP}(\varphi)=1$

An association rule is an implication of the form $A \rightarrow C$, where $A$ and $C$ are two disjoint itemsets called antecedent and consequent, respectively. We refer to the antecedent and consequent of a rule $r$ by using the notations $\text{antc}(r)$ and $\text{cons}(r)$. We refer to the items of rule $r$ by $\text{items}(r)=\text{antc}(r) \cup \text{cons}(r)$. The support of an association rule $r$ is defined as:

$$\text{Supp}(r) = \text{supp}(\text{items}(r)).$$

The confidence of an association rule $r$ is defined as:

$$\text{Conf}(r) = \frac{\text{Supp}(\text{items}(r))}{\text{supp}(\text{antc}(r))}.$$  

Note that the support of an itemset or rule, and the confidence of a rule, take value in the interval $[0,1]$. To simplify the notation, we will use $t_1$, $t_2$, $t_3$ to denote the itemsets $\{i_1, i_2, i_3\}$. In most algorithms, itemsets are actually implemented as lists where the items appear in lexicographic order, which allows them to be processed easily and efficiently.

In [AIS93b], the AR problem was divided into two subproblems:
1. Finding all frequent itemsets.
2. Generating all association rules starting from the frequent item sets found.

Finding all frequent itemsets is the most computationally intensive of the two subproblems and research efforts have focused on finding more efficient algorithms. Some methods search for only a subset of all frequent itemsets that has the property of summarizing or of allowing to infer the information on all frequent itemsets. For example, this subset can represent the set of large itemsets [Bay98] or the set of closed itemsets [2].

**Mining frequent itemsets**

Apriori is the first algorithm proposed in the association rule mining field and many other algorithms were derived from it. Starting from a database, it proposes to extract all association rules satisfying minimum thresholds of support and confidence. It is very well known that mining algorithms can discover a prohibitive amount of association rules; for instance, thousands of rules are extracted from a database of several dozens of attributes and several hundreds of transactions.

**Apriori algorithm**

The interesting feature of this algorithm is that it uses the database only once. The set $C^k$ is used for that purpose. Each member of set is in form $< \text{TID}, \{X_k\} >$, where $X_k$ are potentially large k-itemsets in transaction TID.

For $k=1$, $C^1$ is the database. $L1$ = {large 1-itemsets}

$C^k_{tk} =$ database $D$:

For ($k=2$; $L_{k-1} \subseteq \text{C}^k_{tk}$) do begin

$C^k_{tk}$ = apriori-gen($L_{k-1}$);

For all entries $t \in C^k_{tk}$ do begin

For $k=1$, $C^1$ is the database. $L1$ = {large 1-itemsets}

$C^k_{tk} =$ database $D$:

For ($k=2$; $L_{k-1} \subseteq \text{C}^k_{tk}$) do begin

$C^k_{tk}$ = apriori-gen($L_{k-1}$);

For all entries $t \in C^k_{tk}$ do begin
\[ C_k = \{c \in \text{set-of-items} \mid (c \cap c^{(k-1)}) \cap \text{set-of-items}\}; \]

\[ \text{For all candidates } c \in C_k \text{ do} \]

\[ c.\text{count}++; \]

\[ \text{If } (c^{(k)}) \text{ than } C^k \text{ } \cup \text{ID, } Ct; \]

\[ \text{End} \]

\[ \text{End} \]

\[ L_k = \{C^* \cap C^k \mid c.\text{count} \geq \text{minsup}\} \]

\[ \text{end} \]

\[ \text{ANS} = U_k L_k \]

**Advantages**

1) \( C^k \) could be smaller than the database. If a transaction does not contain \( k \)-itemset candidates, than it will be excluded from \( C^k \)

2) For large \( k \), each entry may be smaller than the transaction. The transaction might contain only few candidates

**Disadvantage**

For small \( k \), each entry may be larger than the corresponding transaction because an entry includes all \( k \)-item sets contained in the transaction.

**REFERENCES**
