Sixth International Baltic Conference on Data Bases and Information Systems

"Using Association Rules to Extract Regularities from Data"

Pēteris Grabusts
Riga Technical University

June 6-9, 2004

Outline

- **Overview of data mining association rules**
- **Algorithm Apriori**
- **Experiment**

Introduction

- **Data Mining**
  - Involves analyzing data in order to identify patterns and establish relationships or regularities
  - Also known as “knowledge discovery”
- **Association Rules**
  - Defines a pattern where one event is linked to another event
  - Mining association rules can be a complex, time-consuming task

Data Mining Association Rules

- Motivated by decision support problem
  - Large retail organizations collect sales data from every sales transaction (known as “basket data”
  - Each record consists of a transaction date and the store items bought in that transaction
  - Basket data can be analyzed to establish patterns such as bying trends
- Databases that store basket data are quite large, so we need “fast” algorithm to mine association rules

Association analysis

Association Analysis -- discovery of association relationships between attribute-value conditions.

Such relationships may be expressed in many ways. On common way is through association rules.

$X \Rightarrow Y$ $\qquad A_1 \wedge \ldots\ldots \wedge A_m \Rightarrow B_1 \wedge \ldots\ldots \wedge B_n$

Association Rules

- Describe relationships or regularities between events/items/attributes
- Form: Body => Head [support, confidence]
  - Example 1. Milk => Bread [8%, 70%]
    - The 8% support indicates that 8% of transactions in the database contained both milk and bread as purchased items
    - The 70% confidence means that 70% of the time, if a customer buys milk, they will also buy bread

Association Rules (cont.)

Example 2:

age (X, “20 .. 29”) ^ income (X, “50Ls..150Ls”) =>
buys (X, “CD player”)
[support = 50% confidence = 60%
]

% of data instances satisfying all three components of rule

% of data instances where hypothesis is satisfied and conclusion is predicted correctly

Formal Definition of Problem

- Formal statement of the problem [AIS93]:
  - Let $I = \{i_1, i_2, \ldots, i_m\}$ be a set of literals, called items. Let D be a set of transactions, where each transaction T is a set of items such that $T \subseteq I$.
  - 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 $X \subset I$, $Y \subset I$, and $X \cap Y = \emptyset$.
  - $X \Rightarrow Y$ holds in the transaction set D with *confidence* c if c% of transactions in D that contain X also contain Y.
  - $X \Rightarrow Y$ has *support* s in the transaction set D if s% of transactions in D contain $X \cup Y$.

Formal Definition of Problem (cont)

Given a set of transactions D, the problem of mining association rules is to generate all association rules that have support and confidence greater than the user-defined minimum support *(minsup)* and minimum confidence *(minconf).*

Discovering Association Rules

Two major steps:

- First, find all large itemsets, which are sets of items that have transaction support greater than minsup.
- Then, use the large itemsets to generate the desired rules which have a computed confidence greater than minconf.

Example: ABCD and AB are the large itemsets

- To determine if rule AB => CD is valid, compute the ratio conf=support(ABCD) / support(AB)
- If conf>=minconf, then the rule holds

Example of Confidence and Support

Computed based on number of transactions in D that include the itemset

TID	Items
100	M, N, O
105	M, O
102	M, P
212	N, O, P

Support ({M, O}) = 2/4= 50%

Support ({M}) = 3/4 = 75%

Support ({N}) = 2/4 = 75%

Support ({O}) = 3/4 = 75%

Suppport ({P}) = 2/4 = 50%

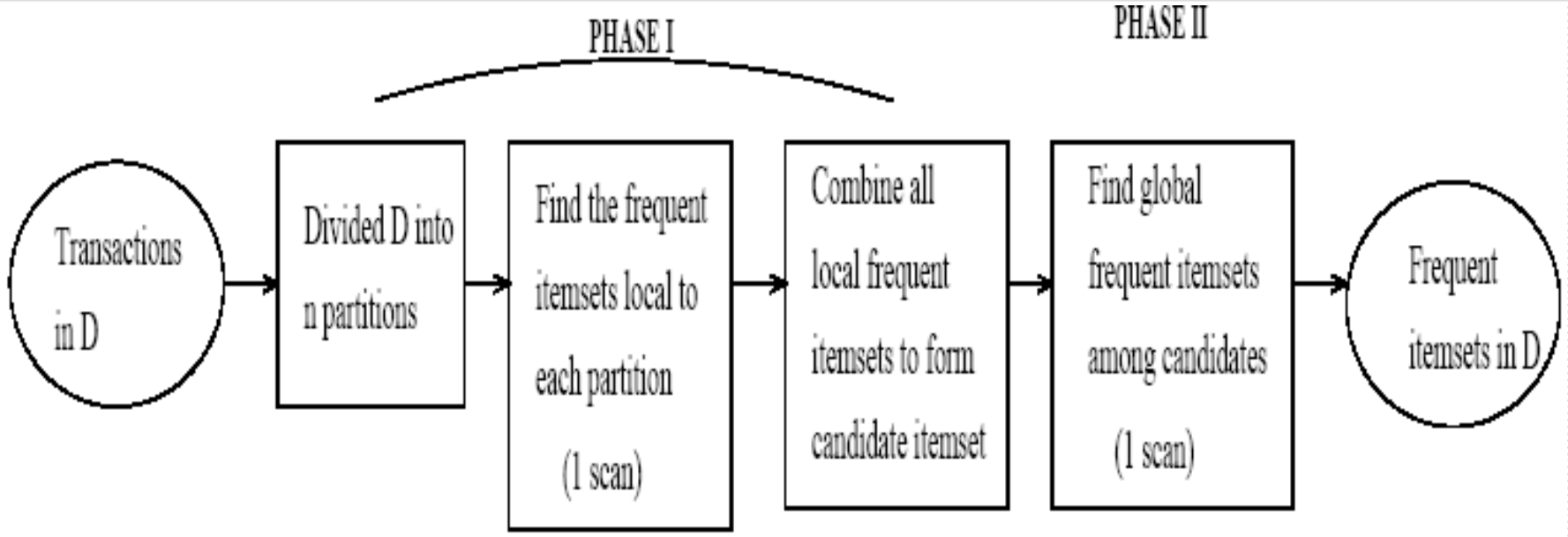
Confidence of Rule M $\Rightarrow$ O = Support ({M, O}) / Support ({M}) = 0.50/0.75 = 2/3 = 66.6%

Result: M $\Rightarrow$ O [50%, 66.6%]

Problem Decomposition

- Find all sets of items (*itemsets*) that have transaction support above minimum support. The *support* for an itemset is the number of transactions that contain the itemset. Itemsets with minimum support are called *large* itemsets, and all others *small* itemsets.
- Use large itemsets to generate the desired rules. If *ABCD* and *AB* are large itemsets, then we can determine if rule $AB \Rightarrow CD$ holds by computing the ratio *conf* = support(*ABCD*)/support(*AB*). If $conf \geq minconf$, then the rule holds. (The rule will surely have minimum support because *ABCD* is large.)

Problem Decomposition (cont.)



Discoverying Large Itemsets

- **Algorithms for discovering large itemsets make multiple passes over the data**
- **In the first pass, we count the support of individual items and determine which of them are large (with minimum support)**
- **In each subsequent pass, we start with a seed set of itemsets found to be large in the previous pass, then use this seed set for generating new potentially large itemsets, called *candidate* itemsets, and count the actual support for these candidate itemsets during the pass over the data**
- **At the end of the pass, we determine which of the candidate itemsets are actually large, and they become the seed for the next pass.**
- **This process continues until no new large itemsets are found**

Notations

- The number of items in an itemset is its *size* and call an itemset of size *k* a *k*-itemset
- Items within an itemset are kept in lexicographic order

***k*-itemset**	**An itemset having *k* items.**
L_k	**Set of large *k*-itemsets (those with minimum support). Each member of this set has two fields: i) itemset and ii) support count.**
C_k	**Set of candidate *k*-itemsets (potentially large itemsets). Each member of this set has two fields: i) itemset and ii) support count.**

Algorithm Apriori (I step)

1) L_1 = {large 1-itemsets};
2) for ($k = 2$; $L_{k-1} \neq \emptyset$; k++) do begin
3) C_k = apriori-gen(L_{k-1}); // New candidates
4) forall transactions $t \in D$ do begin
5) C_t = subset(C_k, t); Candidates contained in t
6) forall candidates $c \in C_t$ do
7) c.count++;
8) end
9) $L_k = \{c \in C_k \mid c.\text{count} \geq \text{minsup}\}$
10) end
11) Answer = $\cup_k L_k$;

Apriori Candidate Generation

The apriori-gen function takes as argument L_{k-1}, the set of all large (k – 1)-itemsets. It returns a superset of the a set of all large k-itemsets.

Apriori Candidate Generation (cont.)

- First, in the join step, we join L_{k-1} with L_{k-1}:

```
insert into Ck
select p.item1, p.item2, …, p.itemk-1, q.itemk-1
  from Lk-1 p, Lk-1 q
  where p.item1 = q.item1, …, p.itemk-2 = q.itemk-2, p.itemk-1 < q.itemk-1
```

- Next, in the prune step, we delete all itemsets c $\in C_k$ such that some $(k-1)$-subset of c is not in L_{k-1}:

```
forall itemsets c ∈ Ck do
  forall (k – 1)-subsets s of c do
    if (s ∉ Lk-1) then delete c from Ck;
```

Apriori-gen Example

- Let L_3 be {{1 2 3}, {1 2 4}, {1 3 4}, {1 3 5}, {2 3 4}}
- In the join step:
  - {1 2 3} joins with {1 2 4} to produce {1 2 3 4}
  - {1 3 4} joins with {1 3 5} to produce {1 3 4 5}
  - After the join step, C_4 will be {{1 2 3 4}, {1 3 4 5}}
- In the prune step:
  - {1 2 3 4} is tested for existence of 3-item subsets within L_3, thus for {1 2 3}, {1 3 4}, and {2 3 4}
  - {1 3 4 5} is tested for {1 3 4}, {1 4 5}, and {3 4 5}, with {1 4 5} not found, and thus this set is deleted
- We then will be left with only {1 2 3 4} in C_4

Generating Association Rules (II step)

- The second major step in mining association rules
- After finding large itemsets using previous algorithm, we can generate association rules
- For every non-empty subset *a of a large itemset I*, we output a rule of the form $a \Rightarrow (I - a)$ if the ratio of support(I) to support(*a*) is at least *minconf*
- Consider all subsets of I to generate rules with multiple consequents

Simple Algorithm

- We can improve the proceeding procedure by generating the subsets of large itemset
- For example, given an itemset *ABCD*, we first consider the subset *ABC*, then *AB*, etc
- Then if a subset *a* of a large itemset *I* does not generate a rule, the subsets of *a* need not to be considered for generating rules using I
- For example, if $ABC \Rightarrow D$ does not have enough confidence, we need not check whether $AB \Rightarrow CD$ holds

Simple Algorithm code

```
1)  forall large itemsets l_k, k ≥ 2 do
2)     call genrules(l_k, l_k)
```

Procedure genrules(l_k: large k-itemsets, a_m: large m-itemsets)

```
1)  A = {(m – 1)-itemsets a_m-1 | a_m-1 ⊂ a_m};
2)  forall a_m-1 ∈ A do again
3)     conf = support(l_k)/support(a_m-1)
4)     if (conf ≥ minconf) then begin
5)           output the rule a_m-1 ⇒ (l_k - a_m-1),
6)           with confidence = conf and support = support(l_k)
7)           if (m – 1 > 1) then
8)                 call genrules(l_k, a_m-1);
9)                 // to generate rules with subsets of a_m-1
10)                   as the antecedents
11)    end
12) end
```

Rule discovery example

- Large itemset: ABCDE
- Assume ACDE =>B and ABCE =>D are only one-item consequent rules with minimum confidence
- Simple algorithm:

  Recursive call to genrules(ABCDE,ACDE) with test if the two-item consequent rules ACD =>BE, ADE =>BC, CDE =>BA and ACE =>BD hold

  (First of the rules cannot hold since E⊂BE and ABCD =>E not have minconf. Second and third rules fail due to similar reasons)

Experimental part

- Main motivation –desire to find regularities in raw data
- Data
  - From Latvian Central Statistics Office
  - Inhabitant migration process
  - Questionary – 3044 respondents
- Environment - Matlab

Questionary

The respondents were asked the following questions:

1. In which country were you born? (with 11 possible reply options offered);
2. How long have you lived in this place? (with 4 reply options offered);
3. Where did you live before removing to this place? (with 3 reply options provided);
4. Please designate the type of the place you lived in before removing to the current place? (with 7 possible reply variants);
5. What was the reason for you to move to the current place? (with 6 possible replies);
6. Are you planning to move to another place within the next 3 years? (with 5 possible reply options provided).

Data

a) SPSS format

2	2	,	,	,	3
2	2	,	,	,	3
4	,	2	2	4	3
5	,	2	5	5	3
2	2	1	3	3	3
4	,	2	5	4	2
2	1	,	,	,	9
2	2	1	5	3	3
2	2	1	3	3	3

b) converted

12	45				83
12	46				83
14	45	52	62	74	83
15	46	52	65	75	83
12	45	51	63	73	83
14	45	52	65	74	82
12	45				88
12	43	51	65	73	83
12	45	51	63	73	83

Experiment I

Minconf=95 & minsup=95

(58 rules were derived)

12	45				83
12	46				83
14	45	52	62	74	83
15	46	52	65	75	83
12	45	51	63	73	83
14	45	52	65	74	82
12	45				88
12	43	51	65	73	83
12	45	51	63	73	83

→

12 74 => 51
46 => 83
12 74 83 => 51
12 46 => 83
45 51 65 => 83

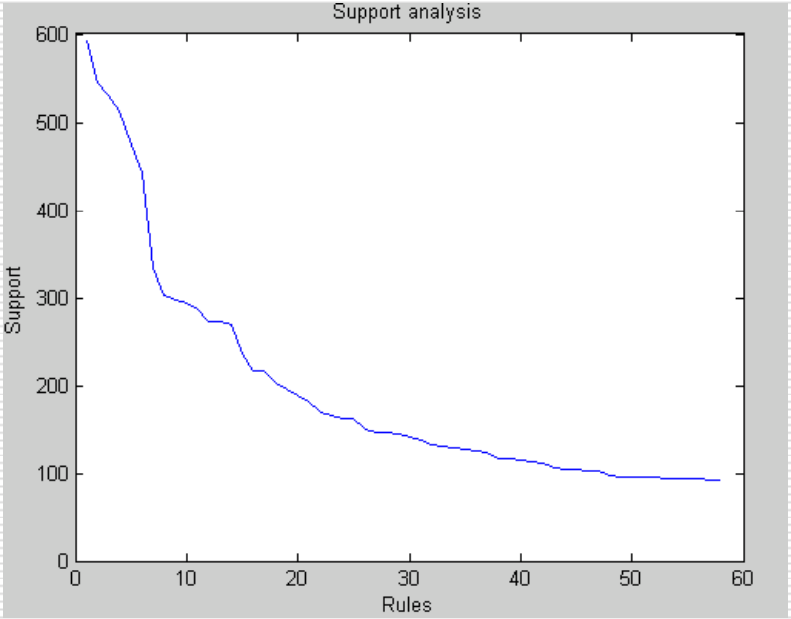
Experiment I (cont.)

Taking into account the decoding results one can conclude that:

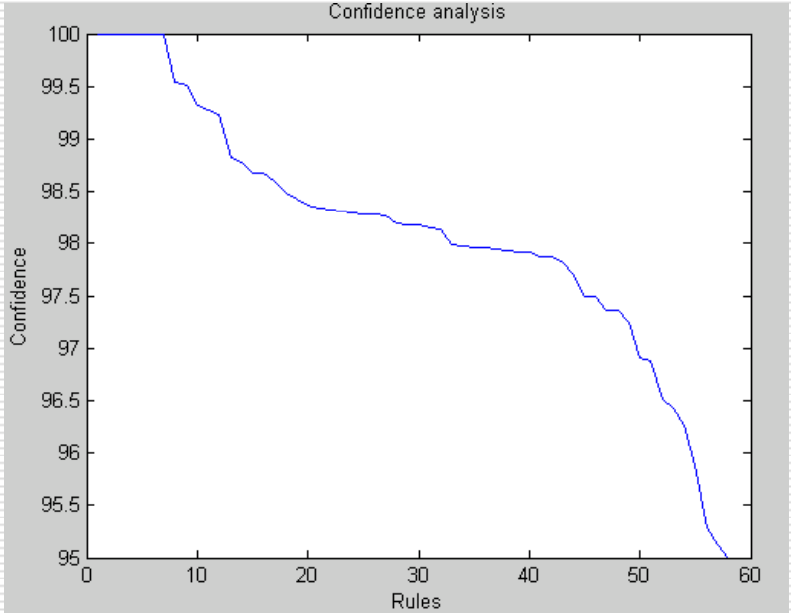
- the 1st rule determines: IF "you were born in Latvia" AND "moved to the current place due to family reasons" THEN "Before moving to the current place you were living in Latvia".
- the 2nd rule determines: IF "You have always lived in this place" THEN "In the next 3 years you are not planning to move to any other place".
- the 3rd rule determines: IF "You were born in Latvia" AND "You have moved to this place due to family reasons" AND "You are not planning to move to any other place within the next 3 years" THEN "Before moving to the current place you lived in Latvia"
- .....

Experiment I (cont.)

Determine the dependence of the count of regularities on the preliminarily assigned boundary values of Support and Confidence:



Dependence of rule count on support values



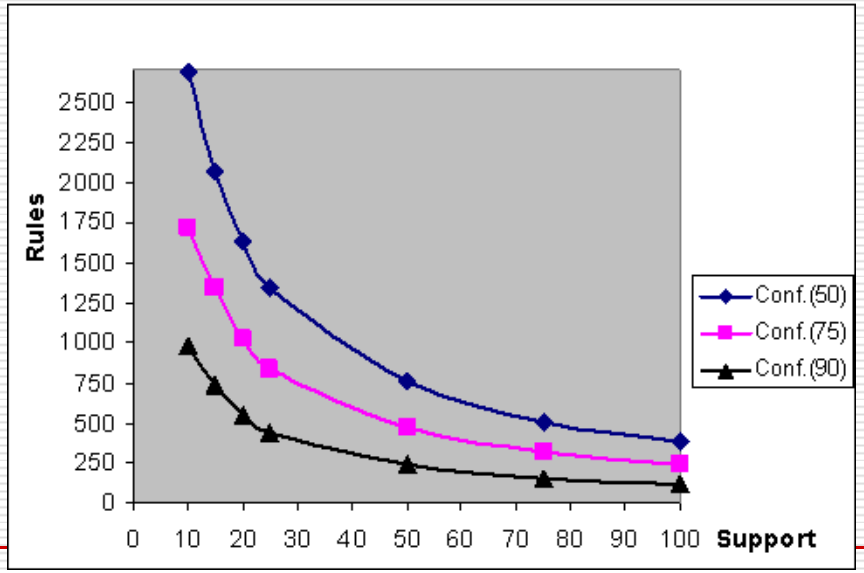
Dependence of rule count on confidence values

Experiment II

- Rule count's dependence on support values:

Support	10	15	20	25	50	75	100
Conf.-50	2690	2071	1628	1346	765	509	386
Conf.-75	1717	1343	1029	839	473	318	241
Conf.-90	983	736	553	436	246	159	121

- Graph of rule count's dependence on support values:



Conclusions

- Software implementation of association rules requires considerable time
- The analysed data should be possibly homogeneous
- Unfortunately, erroneous or strange data also participate in rule formation

Conclusions (cont.)

- Association rules describe relationships or regularities between items/events/attributes
- Mining association rules can be valuable for a variety of applications including marketing
- Two major steps:
  - **Discover large(frequent) itemsets that have minimum support:**
    - **Apriori, AprioriTID**
    - **Soft Apriori (alternative to Apriori)**
    - **FP-trees (alternative to Apriori)**
  - **Generate association rules that have minimum confidence**
    - **Simple & Faster algorithms discussed with Apriori**
- Research has continued to focus on improving the algorithms used to mine association rules

References

- R. Agrawal, T. Imielinski, and A. Swami. Mining Association Rules between Sets of Items in Large Databases. In Proceedings of the 1993 International Conference on Management of Data (SIGMOD 93), pages 207-216, May 1993
- R. Agrawal, R. Srikant. Fast Algorithms for Mining Association Rules (1994) Proc. 20th Int. Conf. Very Large Data Bases, VLDB
- Jiawei Han, Jian Pei, Yiwen Yin. Mining Frequent Patterns without Candidate Generation. SIGMOD Conference 2000: 1-12

Thanks !