

Fast Algorithms for mining Association Rules

Rakesh Agarwal , Ramakrishnan Srikant IBM Almaden Research Center

Presenter : Jun Ma

11/19/09EECS 584, Fall 20081

Motivation

- People buy these items together!
 - Tires, auto accessories and auto services
 - Diapers and beer
- Is there a way to discover these kinds of relations?
- The answer is **DATA MINING**.

11/19/09EECS 584, Fall 20082

Data Mining - Introduction

- What is data mining?
Process of extracting *hidden interesting patterns* representing *knowledge* from large volumes of data.
- Why is it important?
Explosive growth of data (www, barcode readers, etc.)

↓

Patterns useful for marketing, scientific discovery, fraud detection, etc.

11/19/09EECS 584, Fall 20083

What kind of patterns can be mined?

1. Associations
2. Correlations
3. Models (describing classes/concepts)
4. Anomalies
5. Clusters

11/19/09EECS 584, Fall 20084

Association Rule

- **Definition:**
 - I - $\{I_1, I_2, I_3, \dots, I_m\}$
 - D - set of transactions
 - T - transaction with a set of items
 - X, Y - Itemset

$X \Rightarrow Y \quad X \subset I, Y \subset I, X \cap Y = \emptyset$

Support($X \Rightarrow Y$) = $P(X \cup Y)$

i.e., Probability that a transaction contains both X and Y

Confidence($X \Rightarrow Y$) = $P(Y | X)$

i.e., Probability that a transaction containing X (given X) also contains Y

11/19/09EECS 584, Fall 20085

Association Rule

TID	Itemset
1	{a,b}
2	{c,d}
3	{b}
4	{b,c,d}

Support:
{a,b}=1/4

Confidence:
a=>b 1/1
b=>a 1/3

Example :
computer => antivirus_software
[support = 2%, confidence = 60%]
2% of all transactions show that computer and antivirus_ software are purchased together.
60% of customers who purchased a computer also purchased antivirus_ software .

11/19/09EECS 584, Fall 20086

Association Rule Mining

- **Problem definition**
Generate all association rules which satisfy pre-defined threshold support (minsup) and confidence values (minconf).
- **Related previous work**
 - AIS algorithm
 - SETM algorithm
 - KID3 algorithm (*Machine learning*)
 - Functional dependencies

11/19/09

EECS 584, Fall 2008

7

Problem Decomposition

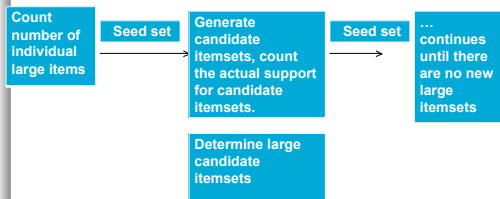
- Two steps to solve the problem
 - Find all the large itemsets (itemsets with minimum support)
 - Generate rules using the large itemsets

11/19/09

EECS 584, Fall 2008

8

Discovering Large Itemsets



Large Itemsets : Itemsets satisfying threshold support value

Candidate Itemsets : Potentially large Itemsets

11/19/09

EECS 584, Fall 2008

9

Apriori Algorithm

Generate candidate itemsets by using only the itemsets found large in previous pass.

Basic Intuition

Any subset of a large itemset is large.

The candidate itemsets having k items can be generated by joining large itemsets having $k-1$ items, and deleting those that contain any subset that is not large.

Apriori algorithm

Assumption : Items are sorted in lexicographic order
 L_k , set of large k -itemsets
 C_k , set of candidate k -itemsets

```

L1 = {large 1-itemsets};
for (k = 2; Lk != ∅; k++) do begin
  Ck = apriori-gen(Lk-1);
  for all transactions t ∈ D do begin
    Ct = subset(Ck, t);
    forall candidates c ∈ Ct do
      c.count++;
  End
  Lk = {c ∈ Ck | c.count ≥ minsup}
End
Answer = ∪k Lk;
  
```

11/19/09

EECS 584, Fall 2008

11

Apriori Candidate Generation

Join step

```

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
  
```

Prune step

```

forall itemsets c ∈ Ck do
  forall (k-1) subsets s of c do
    if (s is not in Lk-1) then
      delete c from Ck;
  
```

L3: {(1,2,3),(1,2,4),
(1,3,4),(1,3,5),(2,3,4)}



C4: {(1,2,3,4),(1,3,4,5)}

11/19/09

EECS 584, Fall 2008

12

Apriori Algorithm - Example Pass 1

Database

Tid	Items
10	A, C, D
20	B, C, E
30	A, B, C, E
40	B, E

C1

Itemset	sup
{A}	2
{B}	3
{C}	3
{D}	1
{E}	3

L1

Itemset	sup
{A}	2
{B}	3
{C}	3
{E}	3

Seed Set

Minimum support_count = 2
Minimum confidence = 65%

11/19/09 EECS 584, Fall 2008 13

Apriori Algorithm - Example Pass 2

L1

Itemset	sup
{A}	2
{B}	3
{C}	3
{E}	3

C2

Itemset	sup
{A, B}	2
{A, C}	2
{A, E}	1
{B, C}	2
{B, E}	3
{C, E}	2

L2

Itemset	sup
{A, C}	2
{B, C}	2
{B, E}	3
{C, E}	2

Seed Set

Database

Tid	Items
10	A, C, D
20	B, C, E
30	A, B, C, E
40	B, E

11/19/09 EECS 584, Fall 2008 14

Apriori Algorithm - Example Pass 3

L2

Itemset	sup
{A, C}	2
{B, C}	2
{B, E}	3
{C, E}	2

C3

Itemset	sup
{B, C, E}	2

L3

Itemset	sup
{B, C, E}	2

Seed Set

{A, B, C}
{A, B, E}
{B, C, E}

Database

Tid	Items
10	A, C, D
20	B, C, E
30	A, B, C, E
40	B, E

11/19/09 EECS 584, Fall 2008 15

Apriori Subset Function

- Candidate itemsets are stored in a hash-tree.
- Leaf node: itemsets.
Interior node: hash table.
- Add an itemset: start from the root and go down the tree until reach a leaf.
- Find all candidates contained in a transaction.
At a leaf: find itemsets in t, add reference
At an interior node: reach by hashing i, hash on each item comes after i in t.
At root: has on every item in t.

AprioriTid Algorithm

- Variation of Apriori algorithm
- TIDs of transactions are associated with the corresponding candidate itemsets.
- Support is calculated using the set of candidate itemsets with TIDs in \bar{C}_k

$\bar{C}_k <TID, \{X_k\}>$
 X_k : Potentially large k-itemset present in the transaction with identifier TID.

11/19/09 EECS 584, Fall 2008 17

Algorithm AprioriTid

```

1)  $L_1 = \{ \text{large 1-itemsets} \};$ 
2)  $\bar{C}_1 = \text{database } D;$ 
3) for (  $k = 2; L_{k-1} \neq \emptyset; k++$  ) do begin
4)    $C_k = \text{apriori-gen}(L_{k-1});$  // New candidates
5)    $\bar{C}_k = \emptyset;$ 
6)   for all entries  $t \in \bar{C}_{k-1}$  do begin
7)     // determine candidate itemsets in  $C_k$  contained
       // in the transaction with identifier  $L_1$ TID
        $C_t = \{ c \in C_k \mid (c - c[k]) \in L_{k-1}\text{-itemsets} \wedge$ 
          $(c - c[k-1]) \in L_{k-2}\text{-itemsets} \};$ 
8)     for all candidates  $c \in C_t$  do
9)        $c.\text{count}++;$ 
10)    if ( $C_t \neq \emptyset$ ) then  $\bar{C}_k += \langle t, \text{TID}, C_t \rangle;$ 
11)  end
12)   $L_k = \{ c \in C_k \mid c.\text{count} \geq \text{minsup} \}$ 
13) end
14) Answer =  $\bigcup_k L_k;$ 

```

Figure 2: Algorithm AprioriTid

AprioriTid Algorithm Example

Database

TID	Items
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

C_1

TID	Set-of-Items
100	{1}, {3}, {5}
200	{2}, {3}, {5}
300	{1}, {2}, {3}, {5}
400	{2}, {5}

I_1

Itemset	Support
{1}	2
{2}	3
{3}	3
{5}	3

C_2

Itemset	Support
{1, 2}	2
{1, 3}	2
{1, 5}	1
{2, 3}	2
{2, 5}	3
{3, 5}	2

I_2

TID	Set-of-Items
100	{1, 3}
200	{2, 3}, {2, 5}, {3, 5}
300	{1, 2}, {1, 3}, {1, 5}
400	{2, 3}, {2, 5}, {3, 5}

C_3

Itemset	Support
{2, 3, 5}	2

I_3

Itemset	Support
{2, 3, 5}	2

11/19/09 EECS 584, Fall 2008 19

Relative performance AIS, SETM, Apriori and AprioriTid

11/19/09 EECS 584, Fall 2008 20

Relative performance Apriori and AprioriTid

11/19/09 EECS 584, Fall 2008 21

AprioriHybrid Algorithm

- Uses Apriori in initial passes.
- Switches to AprioriTid when size of \bar{C}_k gradually decreases.
- Performs better than Apriori in most cases.
- When size of \bar{C}_k is large for many passes and suddenly drops there is no much gain.

11/19/09 EECS 584, Fall 2008 22

Conclusion

- We can use Apriori and AprioriTid algorithms for faster generation of association rules.
- Extending the algorithm further
 - hierarchy of items
 - quantities of items

11/19/09 EECS 584, Fall 2008 23