TUTORIAL 1

Cx4032-Data Analytics and Mining
(Data Mining)

Q1

- The following is a pseudocode of Apriori algorithm

1. Generate frequent itemsets of length 1
2. Repeat until no new frequent itemsets are identified
   a. Generate length (k+1) candidate itemsets from length k frequent itemsets
   b. Prune candidate itemsets containing subsets of length k that are infrequent
   c. Count the support of each candidate by scanning the DB
   d. Eliminate candidates that are infrequent, leaving only those that are frequent

Lecture slide: Anti-Monotone Property

- Any subset of a frequent itemset must be also frequent — an anti-monotone property
  - Any transaction containing {beer, diaper, milk} also contains {beer, diaper}
  - {beer, diaper, milk} is frequent → {beer, diaper} must also be frequent
- In other words, any superset of an infrequent itemset must also be infrequent
  - **No superset of any infrequent itemset should be generated or tested**
  - Many item combinations can be pruned!

Answer

- Suppose minsup =4 we have the following frequent items:
  - frequent items a,b,c,d,e,f
  - Frequent 2-itemsets: ab 4, ad 5, ae 4, bd 4, bf 4, de 4, df 4
  - If we generate 3 itemsets: abd ?
  - If we generate 3 itemsets: abe ?

Q2

- we introduce how to generate length (k+1) candidate itemsets from length k frequent itemsets. Explain this with an example. Can give another way of generating candidates?

Candidate Generation: $F_{k-1} \times F_{k-1}$ Method

- To generate C_{k+1} from F_k: Merge two frequent (k)-itemsets if their first (k-1) items are identical

- F_3 = {ABC,ABD,ABE,ACD,BCD,BDE,CDE}
  - Merge(**AB**C, **AB**D) = **AB**CD
  - Merge(**AB**C, **AB**E) = **AB**CE
  - Merge(**AB**D, **AB**E) = **AB**DE

Any k itemset subsets must be frequent. So the two k-itemsets with the identical k-1 items should be frequent.

  - Do not merge(**A**BD,**A**CD) because they share only prefix of length 1 instead of length 2

- Different ways of generating candidates.
  - They may generate different sets of candidate.
  - The real frequent itemsets must be included.
  - but after pruning, they will be the same.

Candidate Generation: Merge Fk-1 and F1 itemsets

Frequent 2-itemset

Itemset
{Beer, Diapers}
{Bread, Diapers}
{Bread, Milk}
{Diapers, Milk}

Frequent 1-itemset

Item
Beer
Bread
Diapers
Milk

Candidate Generation

Itemset
{Beer, Bread, Diapers}
{Beer, Diapers, Milk}
{Bread, Diapers, Milk}
{Beer, Bread, Milk}

Candidate Pruning

Itemset
{Bread, Diapers, Milk}

Figure 5.7. Generating and pruning candidate k-itemsets by merging a frequent $(k-1)$-itemset with a frequent item. Note that some of the candidates are unnecessary because their subsets are infrequent.

More example

- F_3 = {ABC,ABD,ABE,ACD,BCD,BDE,CDE}
- F1= {A, B, C, D, E}
- We generate
  - ABC: ABCD, ABCE
  - ABD:ABDE
  - ABE:
  - ACD: ACDE
  - BCD: BCDE
  - BDE:
  - CDE:

After prunning
C_4 = {ABCD}

Q3 finding frequent itemsets

- 1: a, c,d
- 2, b, c,e
- 3, a, b, c, e
- 4, b, e,

- Minsup =2

Example: Apriori-based Mining

Database D

TID	Items
10	a, c, d
20	b, c, e
30	a, b, c, e
40	b, e

Min_sup=2 (0.5)

Scan D →

1-candidates

Itemset	Sup
a	2
b	3
c	3
d	1
e	3

→

Freq 1-itemsets

Itemset	Sup
a	2
b	3
c	3
e	3

→

2-candidates

Itemset
ab
ac
ae
bc
be
ce

Scan D →

Counting

Itemset	Sup
ab	1
ac	2
ae	1
bc	2
be	3
ce	2

→

Freq 2-itemsets

Itemset	Sup
ac	2
bc	2
be	3
ce	2

→

3-candidates

Itemset
bce

Scan D →

Freq 3-itemsets

Itemset	Sup
bce	2

Q4

- Suppose {B,C,D} is a frequent itemset. Enumerate the candidate rules:

Step2: Rule generation

- **Step 1:** Find all frequent itemsets I
  - Generate all itemsets whose support $\geq$ *minsup*
- **Step 2: Rule generation**
  - Given a frequent itemset I, find all non-empty subsets $A \subset I$ such that $A \rightarrow I - A$ satisfies the **minimum confidence requirement *minconf***
    - If {A,B,C,D} is a frequent itemset, candidate rules:

      ABC →D, ABD →C, ACD →B, BCD →A,
      A →BCD, B →ACD, C →ABD, D →ABC
      AB →CD, AC → BD, AD → BC, BC →AD,
      BD →AC, CD →AB

    - An example to compute the rule confidence
      - confidence(***A,B→C,D***) = support(**A,B,C,D**) / support(**A,B**)
  - Do the above for every frequent itemset, and **output all the rules above the confidence threshold**

- If {B,C,D} is a frequent itemset, candidate rules:

  $BC \rightarrow D$, $BD \rightarrow C$, $CD \rightarrow B$

  $B \rightarrow CD$, $C \rightarrow BD$, $D \rightarrow BC$

Q5

- Consider the observation: If A,B,C→D is below confidence, so is A,B→C,D. Can we design an efficient order of generating rules based on the observation?

Pruning in Rule Generation

- Confidence does not have an anti-monotone property

  $c(ABC \rightarrow D)$ can be larger or smaller than $c(AB \rightarrow D)$

- How to prune?
  - Confidence of rules generated from the same itemset has an anti-monotone property
  - E.g., Suppose {A,B,C,D} is a frequent 4-itemset:

$$c(ABC \rightarrow D) \geq c(AB \rightarrow CD) \geq c(A \rightarrow BCD)$$

  Confidence is anti-monotone w.r.t. number of items on the RHS of the rule

  - **Observation:** If **A,B,C→D** is below confidence, so is **A,B→C,D**
  - Can generate "bigger" rules from smaller ones (RHS)!

We check $c(ABC \rightarrow D)$, next $c(AB \rightarrow CD)$, followed by $c(A \rightarrow BCD)$

Rule Generation for Apriori Algorithm

Lattice of rules