UNIT - 3

ining Frq patterns, Associations and correlations

→ FP are patterns (such as itemsets, subsequences or substructures) that appear in a data set frequently.

Ex: a set of items, such as milk and bread, that appear frqly together in a transc data set is a frq itemset.

→ FP mining searches for recurring relationships in a given data set.

→ FP mining for the discovery of interesting associations and correlations b/w itemsets in transcational and relational databases.

Motivation: user want to know

(a) which ~~provid~~ products are often purchased together

(b) what are subsequent purchases.

Applications: (a) Basket data analysis, catlog design, web log analysis, sale campaign analysis and DNA sequence analysis.

Importance: opens the inner properties of data sets and its foundations for many datamining tasks.

They are (a) Association, correlation analysis

(b) Sequential, structural patterns

(c) pattern analysis

(d) Classification and clustering analysis

(e) semantic data compression and Broad app's,

... variable of 1..k

$$\text{support} = \frac{\text{min-supp} \times \text{no. of itemsets}}{100} = \frac{50 \times 6}{100} = 3 \text{ (threshold value)}$$

no. of itemsets = 6.

Confidence: $\text{confidence}(A \Rightarrow B) = P(B|A) = \frac{\text{support}(A \cup B)}{\text{support}(A)}$

$$= \frac{\text{support\_count}(A \cup B)}{\text{support\_count}(A)}$$

FP mining : A Road map.

FP mining can be categorized in many diff ways

1. Based on the completeness of patterns to be mined (complete set of freq itemsets, Closed freq itemsets, max
2. Based on the levels and dimensions of data involved in the rule
3. Based on the types of values handled in the rule,
4. Based on the kinds of rules to be mined.
5. Based on the kinds of patterns to be mined.

(2) → can find rules at diff levels of abstraction. for ex: a set of association rules to be mined includes the following rules, in which X is a variable representing customers who purchased items

buys(X, "computer") ⇒ buys(X, "HP-printer") —(1)

buys(X, "laptop-computer") ⇒ buys(X, "HP-printer") —(2)

Here computer is a higher-level abstraction of laptop computer

(3) buys(X, "comp") ⇒ buys(X, "antivirus-sw) —(3)

all (3) rules are single dimensional association rules because they each refer to only one dimension, buys.

(4) If a rule involves associations b/w the presence or absence of items, it is a Boolean association rule.

age(X, "30..39") ∧ income(X, "42K..48K") ⇒ buys(X, "high resolution TV")

(5) Generate various kinds of rules and also interesting relationships. Assoc.

Market-Basket Analysis :

→ Frq itemset mining leads to the discovery of Associations and correlations among items in large transactional or relational data sets.

→ The discovery of interesting correlation relationships among huge amounts of business transc records can help in many business decision-making processes, such as catalog design, cross-marketing and customer shopping behavior analysis.

$$\text{Support}\ (A \Rightarrow B) = P(A \cup B)$$

$$\text{confidence}\ (A \Rightarrow B) = P(B|A)$$

$$= \frac{\text{support}\ (A \cup B)}{\text{support}\ (A)} = \frac{\text{support\_count}(A \cup B)}{\text{support\_count}(A)}.$$

→ Rules that satisfy both a minimum support threshold (min_sup) and a minimum confidence threshold (min_conf) are called strong.

Generating Association rules

$l = \{I_1, I_2, I_5\}$. what are the assoc rules that can be generated from l? The nonempty subsets of l are $\{I_1, I_2\}$, $\{I_1, I_5\}$, $\{I_2, I_5\}$, $\{I_1\}$, $\{I_2\}$ and $\{I_5\}$. The resulting asso rules are as shown below, each listed with its confidence.

Alg: Apriori : find frq itemsets using an iterative level-wise approach based on candidate generation.

Input : → D, a database of transcations;
→ min_sup, the minimum support count threshold.

Output : L, freq itemsets in D.

Method :

1) L_1 = find_frq_1-itemsets(D);
2) for ($k = 2$; $L_{k-1} \neq \phi$; $k++$) {
   C_k = apriori_gen(L_{k-1});
   for each transc $t \in D$ { // scan D for counts
      C_t = subset(C_k, t); // get the subsets of t that are candidates
      for each candidate $c \in C_t$
         c.count++;
   }
   $L_k = \{ c \in C_k \mid c.count \geq min\_sup \}$
}
return $L = \cup_k L_k$;

Procedure apriori_gen (L_{k-1} : frq(k-1) - itemsets)
   for each itemset $l_1 \in L_{k-1}$
      for each itemset $l_2 \in L_{k-1}$
if $(l_1[1] = l_2[1] \wedge (l_2[2] = l_2[2]) \wedge \ldots \wedge (l_1[k-2] = l_2[k-2]) \wedge (l_1[k-1] < l_2[k-1])$ then
   {
   c = $l_1 \bowtie l_2$; // join step : generate candidates
   if has_infrequent_subset (c, L_{k-1}) then
      delete c; // prune step : Remove unfruitful candidate
   else add c to C_k;
   }
return C_k;

Procedure has_infrq_subset(c: candidate k-itemset;
   L_{k-1} : frq (k-1) - itemsets); // use prior knowledge
   for each (k-1)-subset s of c
      if $s \in L_{k-1}$ then
         return TRUE;
   return FALSE;

Q.

TID	List of Items
T100	$I_1, I_2 I_5$
T200	I_2, I_4
T300	I_2, I_3
T400	I_1, I_2, I_4
T500	I_1, I_3
T600	I_2, I_3
T700	I_1, I_5
T800	I_1, I_2, I_3, I_5
T900	I_1, I_2, I_3

$I_1 \wedge I_2 \Rightarrow I_5$ conf $= 2/4 = 50\%$ — (1)

$I_1 \wedge I_5 \Rightarrow I_2$ conf $= 2/2 = 100\%$ — (2)

$I_2 \wedge I_5 \Rightarrow I_1$ conf $= 2/2 = 100\%$ — (3)

$I_1 \Rightarrow I_2 \wedge I_5$ conf $= 2/6 = 33\%$ — (4)

$I_2 \Rightarrow I_1 \wedge I_5$ conf $= 2/7 = 29\%$ — (5)

$I_5 \Rightarrow I_1 \wedge I_2$ conf $= 2/2 = 100\%$ — (6)

So

Scan D for count of each candidate →

C1

Itemset	Sup.count
I_1	6
I_2	7
I_3	6
I_4	2
I_5	2

compare candidate support count with minimum support count = 2 →

L1

Itemset	Sup-Count
I_1	6
I_2	7
I_3	6
I_4	2
I_5	2

Generate C_2 candidate from L1 →

C2

I_1, I_2	4
I_1, I_3	4
I_1, I_4	1
I_1, I_5	2
I_2, I_3	4
I_2, I_4	2
I_2, I_5	2
I_3, I_4	0
I_3, I_5	1
I_4, I_5	0

L2

itemset	Sup-Count
I_1, I_2	4
I_1, I_3	4
I_1, I_5	2
I_2, I_3	4
I_2, I_4	2
I_2, I_5	2

Suppose threshold value = 70%. Then rules (2) (3) (6) satisfies association rules.

C3

I_1, I_2, I_3	2
I_1, I_2, I_5	2
I_1, I_2, I_4	1
I_1, I_3, I_5	1
I_2, I_3, I_4	0
I_2, I_3, I_5	1
I_2, I_4, I_5	0

I_1, I_2, I_3	2
I_1, I_2, I_5	2

3. min-sup $= 2/9 = 22\%$.

(or)

$= \frac{\text{min-sup} \times \text{no. of itemsets}}{100}$

$= \frac{22\% \times 9}{100} = 1.98$

Confidence $(A \Rightarrow B) = P(B/A)$

$= \frac{\text{support\_count}(A \cup B)}{\text{support\_count}(A)}$

$\forall x \in$ transaction, buys $(x, I_1) \wedge$ buys $(x, I_5) \Rightarrow$ buys (x, I_2)

$[s = 2, c = 100\%]$

Alg: FP-growth. Mine freq itemsets using an FP tree by pattern fragment growth.

INPUT: → D a transc db

→ min_sup, the min support count threshold.

Output: The complete set of frq patterns.

Method: (1) The FP-tree is constructed in the foll. steps:

(a) Scan the transcation db D once. collect F, the set of frq items and their support counts sort F in support count desc order as L, the list of frq items.

(b) Create the root of an FP-tree and label it as "null". For each transc Trans in D do the foll

Select and sort the frq items in Trans acc to the order of L. Let the sorted frq item list in Trans be [p|P], where p is the first element and P is the remaining list. in

call insert_tree([p|P], T), which is performed as follows.

If T has a child N such that N.item-name = p.item-name then incr N's count by 1; else create a new node N, and let its count be 1, its parent linked to be linked to T, and its node-link to the nodes with the same item-name via the node-link structure. If P is nonempty, call insert-tree(P, N) recursively.

(2) The FP-tree is mined by calling FP-growth(FP_tree, null), which is implemented as follows.

– cont –

Procedure FP_growth (Tree, α)

① if Tree contains a single path P then
② for each combination (denoted as β) of the nodes in the path P.
③ generate pattern β ∪ α with support_count = minimum support count of nodes in β;
④ else for each a_i in the header of Tree {
⑤ generate pattern β = a_i ∪ α with support_count = a_i . support_count;
⑥ construct β's conditional pattern base and then β's conditional FP_tree $Tree_\beta$;
⑦ if $Tree_\beta \neq \phi$ then
⑧ call FP_growth ($Tree_\beta$, β);
}

Fig: The FP-growth alg for discovering frq itemsets without candidate generation.

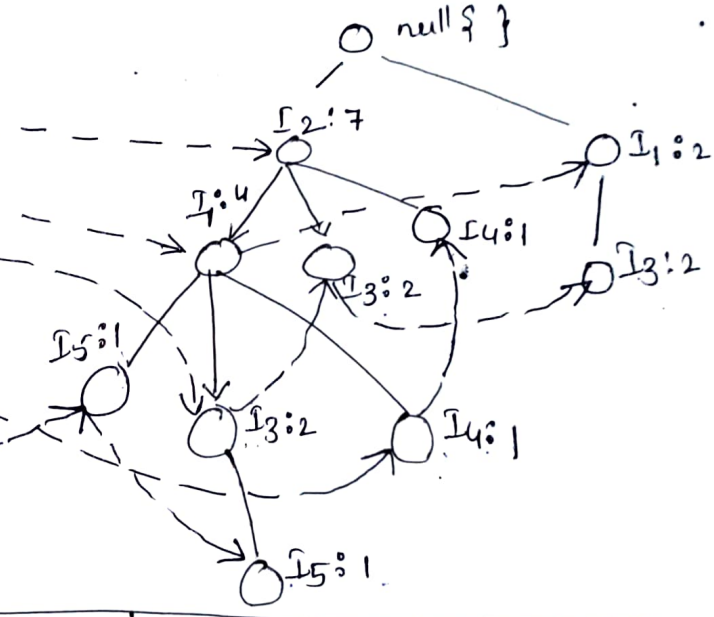
Ex:

TID	List of Item-IDs
T100	I_1, I_2, I_5
T200	I_2, I_4
T800	I_2, I_3
T400	I_1, I_2, I_4
T500	I_1, I_3
T600	I_2, I_3
T700	I_1, I_3
T800	I_1, I_2, I_3, I_5
T900	I_1, I_2, I_3

After descending order ⇒

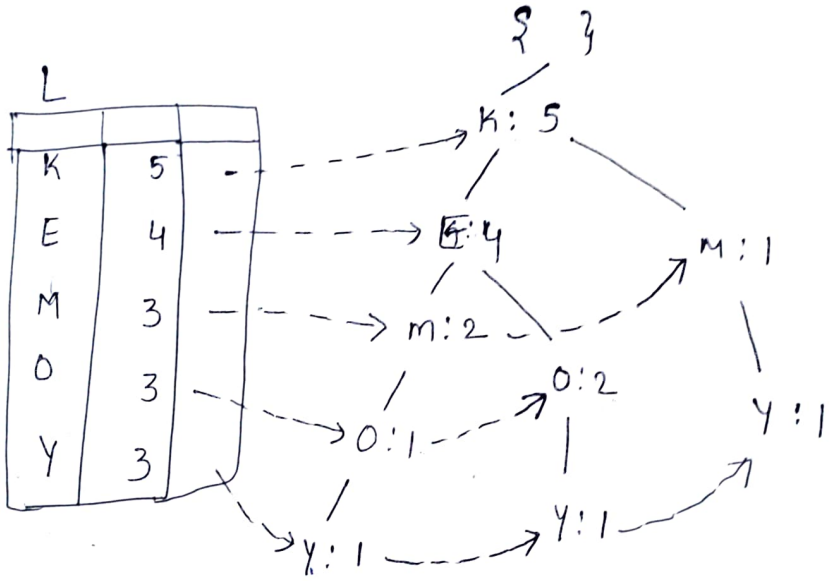
T100	I_2, I_1, I_5
T200	I_2, I_4
T300	I_2, I_3
T400	I_2, I_1, I_4
T500	I_1, I_3
T600	I_2, I_3
T700	I_1, I_3
T800	I_2, I_1, I_3, I_5
T900	I_2, I_1, I_3

Item ID	Support count	Node link
I_2	7	
I_1	6	
I_3	6	
I_4	2	
I_5	2	



Item	Conditional pattern Base	Conditional FP-tree	Freq. patterns Generated.
I_5	$\{\{I_2, I_1 : 1\}, \{I_2, I_1, I_3 : 1\}\}$	$\langle I_2 : 2, I_1 : 2\rangle$	$\{I_2, I_5 : 2\}, \{I_1, I_5 : 2\}$, $\{I_2, I_1, I_5 : 2\}$
I_4	$\{\{I_2, I_1 : 1\}, \{I_2 : 1\}\}$	$\langle I_2 : 2\rangle$	$\{I_2, I_4 : 2\}$
I_3	$\{I_2, I_1 : 2\}, \{I_2 : 2\}, \{I_1 : 2\}$	$\langle I_2 : 4, I_1 : 2\rangle \langle I_1 : 2\rangle$	$\{I_2, I_3 : 4\}$, $\{I_1, I_3 : 4\}$, $\{I_2, I_1, I_3 : 2\}$
I_1	$\{\{I_2 : 4\}\}$	$\langle I_2 : 4\rangle$	$\{I_2, I_1 : 4\}$

TID	Items	ordered itemset
T_{100}	{M, O, N, K, E, Y}	{K, E, M, O, Y}
T_{200}	{D, O, N, K, E, Y}	{K, E, O, Y}
T_{300}	{M, A, K, E}	{K, E, M}
T_{400}	{M, U, C, K, Y}	{K, M, Y}
T_{500}	{C, O, O, K, I, E}	{K, E, O}



Items	Conditional pattern base	Conditional FP tree	freq-pattern generated
Y	{{K, E, M, O : 1} {K, E, O : 1} {K : M : 1}}	K : 3	⟨K, Y : 3⟩
O	{{K, E, M : 1} {K, E : 2}}	K, E : 3	⟨K, O : 3⟩ ⟨E, O : 3⟩ ⟨O, K, E : 3⟩
M	{{K, O : 2} {K : 1}}	K : 3	⟨m, K : 3⟩
E	{K : 4}	K : 4	⟨E, K : 4⟩
K	—		

Improving the efficiency of Apriori

Hash-based tech — used to reduce the size of the candidate K-itemsets, C_K, for $K > 1$.

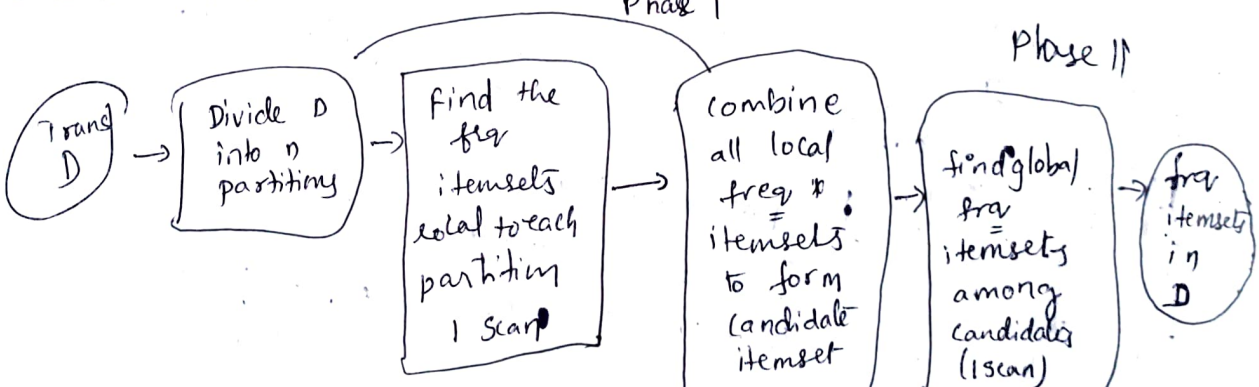
Create hash table H_2 using hash function $h(x,y) = ((\text{order of } x) \times 10 + (\text{order of } y)) \bmod 7$

H_2

bucket address	0	1	2	3	4	5	6
" count	2	2	4	2	2	4	4
" contents	$\{I_1, I_4\}$ I_3, I_5	I_1, I_5 I_1, I_5	I_2, I_3 I_2, I_3 I_2, I_3 I_2, I_3	$I_2 I_4$ I_2, I_4	I_2, I_5 I_2, I_5	I_1, I_2 I_1, I_3 I_1, I_2 I_1, I_2	I_1, I_3 I_1, I_3 I_1, I_3 I_1, I_3

(2) Transcation reduction (reducing the no. of transcations scanned in future iterations): A transc that does not contain any frequent K-itemsets cannot contain any freq (K+1) itemsets.

(3) partitioning (partitioning the data to find candidate itemsets).



(4) Sampling (mining on a subset of the given data): the basic idea of the sampling approach is to pick a random sample S of the given data D, and then search for frq itemsets in S instead of D. The sample size of S is such that the search for frq itemsets in S can be done in mainmemory, and so only one scan of the transc in S is reqd overall.

Becoz we are searching for frq itemsets in S rather than in D, it is possible that we will miss some of the global frq itemsets.

dynamic itemset counting (adding candidate itemsets at different points during a scan):

A dynamic itemset counting technique was proposed in which the database is partitioned into blocks marked by start points. In this variation, new candidate itemsets can be added at any start point, unlike in Apriori, which determine

1 - itemset

TID	Itemset
T100	I_1 I_2 I_5
T200	I_2 I_4
T300	I_2, I_3
T400	I_1, I_2, I_4
T500	I_1, I_3
T600	I_2, I_3
T700	I_1, I_3
T800	I_1, I_2, I_3, I_5
T900	I_1, I_2, I_3

→

items	support count
I_1	6
I_2	7
I_3	6
I_4	2
I_5	2

Hash fun

$h(x, y) = (\text{order of } x) * 10 + (\text{order of } y) \bmod 7$

$I_1 = 1$, $I_2 = 2$, $I_3 = 3$

$I_4 = 4$, $I_5 = 5$

2 - itemset

Itemset	sup count
I_1 I_2 (x y)	4
I_1 I_3 (x y)	4
I_1 I_4 (x y)	1
I_1 I_5 (x y)	2
I_2, I_3 (x y)	4
I_2, I_4	2
I_2, I_5	2
I_3, I_4	0
I_3, I_5	1
I_4, I_5	0

→ Size = 7. remainder

$(1 * 10 + 2) \bmod 7 = 5$
$(1 * 10 + 3) \bmod 7 = 6$
$(1 * 10 + 4) \bmod 7 = 0$
$(1 * 10 + 5) \bmod 7 = 1$
$(2 * 10 + 3) \bmod 7 = 2$
$(2 * 10 + 4) \bmod 7 = 3$
$(2 * 10 + 5) \bmod 7 = 4$
null
$(3 * 10 + 5) \bmod 7 = 0$
null

7)12(1 7 (5)

7)13(1 7 (6)

Now consider this as Bucket address

for example:

figure

Bucket '0' address contains

{ I_1, I_4, }

{ I_3, I_5 }

as count is (2)

From Association Analysis to Correlation Analysis

→ A correlation measure can be used to augment (increase) the support-confidence framework for association rules. This leads to correlation rules of the form (→ connection or relationship)

$$A \Rightarrow B [\text{support}, \text{confidence}, \text{correlation}]$$

→ That is, a correlation rule is measured not only by its support and confidence but also by the correlation between itemsets A & B.

→ Lift is a simple correlation measure that is given as follows.

→ The occurrence of itemset A is independent of the occurrence of itemset B if

$$P(A \cup B) = P(A)\,P(B)$$; otherwise, itemsets A & B are dependent & correlated as events.

→ This definition can easily be extended to more than two itemsets. The lift b/w the occurrence of A & B can be measured by computing

$$\text{lift}(A,B) = \frac{P(A \cup B)}{P(A)P(B)} \quad \text{— (1)}$$

If the resulting value of eqn (1) is less than 1, then the occurrence of A is

negatively correlated with the occurrence of B.

→ If the resulting value is greater than 1, then A & B are positively correlated, meaning that the occurrence of one implies the occurrence of the other.

→ If the resulting value is equal to 1, then A & B are independent & there is no correlation b/w them.

correlation Analysis using lift

→ Suppose we are interested in analyzing transcation at AllElectronics with respect to the purchase of computer games and videos. Let game refer to the transcations containing computer games and video refer to those containing video's.

→ Of the 10,000 trian's analyzed, the data show that 6,000 of the customer tran's included computer games, while 7,500 included video's, and 4000 included both computer games and video's.

→ Suppose that a data mining program for discovering association rules is run on the data, using a minimum support of, say 30% and a minimum confidence of 60%. The following association rule is discovered:

$$buys(X, \text{"computer games"}) \Rightarrow buys(X, \text{"videos"})$$

$$[support = 40\% \quad confidence = 66\%] \quad \text{—— (1)}$$

rule (1) is a strong association rule and would therefore be reported, since its support value of $\frac{4,000}{10,000} = 40\%$ & confidence value $= \frac{4000}{6000} = 66\%$ satisfy the min Sup & min_conf threshold resp.

→ In this, to help filter out misleading "strong" associations of the $A \Rightarrow B$ form the data, in above ex, we need to study how the two itemsets, A & B are correlated.

→ Let $\overline{game}$ refer to the trans of example that do not contain computer games, and $\overline{video}$ refer to those that do not contain videos.

negatively correlated with the occurrence

→ If the resulting value is greater tha
then A & B are positively correlated,
meaning that the occurrence of one im
the occurrence of the other.

→ If the resulting value is equal to
then A & B are independent & there
no correlation b/w them.

correlation Analysis using lift

→ suppose we are interested in analyzi
transcation at AllElectronics with respe
the purchase of computer games and
Let game refer to the transcations
Computer games and video refer to
containing video's.

→ Of the 10,000 trian's analyzed, the
show that 6,000 of the custom
included computer games, while 7,
included video's, and 4000 included
computer games and video's.