Parul® University

Data Mining and Warehousing (03105430)

Prof. Dheeraj Kumar Singh, Assistant Professor
Information Technology Department

Parul®
University

DIGITAL LEARNING CONTENT

CHAPTER- 5

Mining Frequent Patterns, Associations, and Correlations

Market Basket Analysis

- Analyzes customer buying habits by finding associations between different items that customers place in their “shopping baskets”.
- Help retailers to develop marketing strategies.

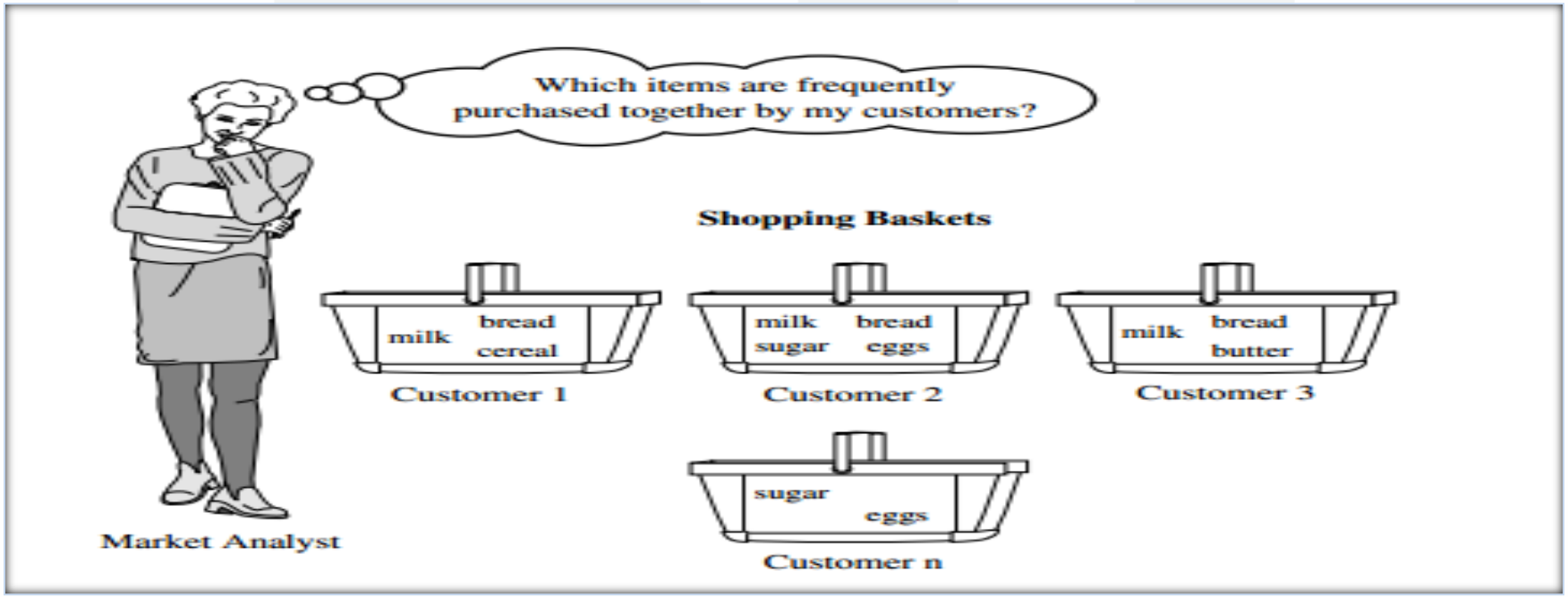


Image source : Data Mining Concepts and Techniques Book

Frequent Pattern Mining

- Frequent patterns are patterns, such as itemsets, subsequences, or substructures that appear in a data set frequently.
- A subsequence, such as buying a PC, followed by a digital camera, if it occurs frequently in a shopping history database, is a (frequent) sequential pattern.
- A substructure can refer to different structural forms, such as subgraphs, subtrees, or sublattices, which may be combined with itemsets or subsequences. If a substructure occurs frequently, it is called a (frequent) structured pattern.

Measures of rule interestingness

- Association rules are considered interesting if they satisfy both a minimum support and a minimum confidence threshold.
- The rule A ⇒ B holds in the transaction set D with support s, where s (relative support) is percentage of transactions in D that contain A ∪ B .
- Given as the probability, P(A ∪ B) : support(A⇒B) = P(A ∪ B)
- The rule A ⇒ B has confidence c in the transaction set D, where c is the percentage of transactions in D containing A that also contain B.
- Given as the conditional probability, P(B|A):

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

Basic Terminology of Association rule mining

- A set of items is referred to as an itemset. An itemset that contains k items is a k-itemset.
  - Example: The set {computer, antivirus software} is a 2-itemset.
- The occurrence frequency of an itemset is number of transactions that contain the itemset. This is also known as the frequency, support count, absolute support or count of the itemset.
- If relative support of an itemset I satisfies a pre specified minimum support threshold then I is a frequent itemset.

Basic Terminology of Association rule mining

- A set of items is referred to as an itemset. An itemset that contains k items is a k-itemset.
  - Example: The set {computer, antivirus software} is a 2-itemset.
- The occurrence frequency of an itemset is number of transactions that contain the itemset. This is also known as the frequency, support count, absolute support or count of the itemset.
- If relative support of an itemset I satisfies a pre specified minimum support threshold then I is a frequent itemset.

Association Rule Mining

- **Can be viewed as a two-step process:**

  1. Find all frequent itemsets:

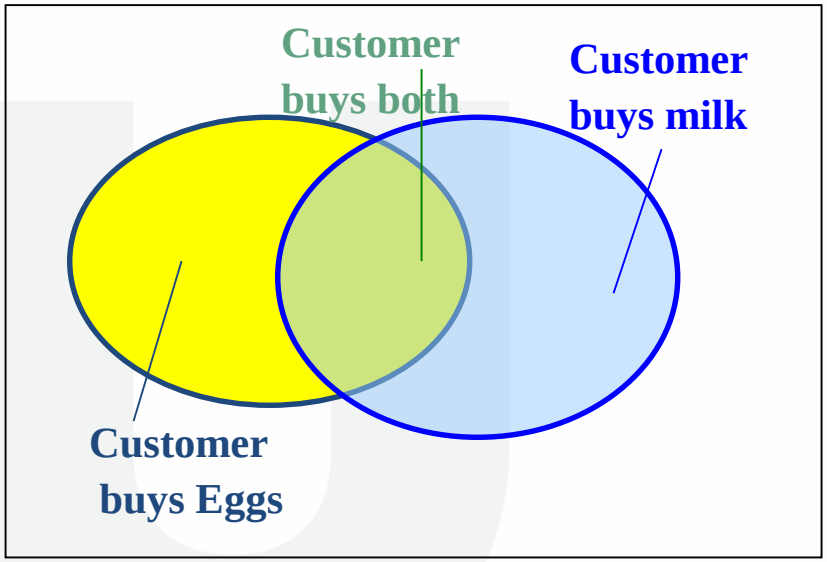
  – **By definition, each of these itemsets will occur at least as frequently as a predetermined minimum support count, min_sup.**

  2. Generate strong association rules from the frequent itemsets:

  – **By definition, these rules must satisfy minimum support and minimum confidence.**

Association Rule Mining

- Given Itemset X = $\{x_1, \ldots, x_k\}$
- Find all the rules *X* → *Y* with minimum support and confidence.
- Where
  - support *s*, is probability that a transaction contains X ∪ Y.
  - Confidence *c, is* conditional probability that a transaction having X also contains *Y*



Customer transaction example

Association Rule Mining: Example

Let sup_{min} = 50%, $conf_{min}$ = 50%

Then Frequent Pattern will be :

{A:3, B:3, D:4, E:3, AD:3}

Association rules:

A → D (60%, 100%)`

D → A (60%, 75%)

…

Transaction-id	Items bought
10	A, B, D
20	A, C, D
30	A, D, E
40	B, E, F
50	B, C, D, E, F

Transaction Database Example

Frequent pattern mining Classification

- Based on the completeness of patterns to be mined
- Based on the levels of abstraction involved in the rule set
- Based on the number of data dimensions involved in the rule
- Based on the types of values handled in the rule
- Based on the kinds of rules to be mined
- Based on the kinds of patterns to be mined

Classification based on the completeness of patterns

- Complete set of frequent itemsets
- Closed frequent itemsets
- Maximal frequent itemsets
- Constrained frequent itemsets
- Approximate frequent itemsets
- Near-match frequent itemsets
- Top-k frequent itemsets

Classification based on the levels of abstraction

- **Multilevel association rules: When a rule references different levels of abstraction**
  - *buys(X , "computer") ⇒ buys(X , "HP printer")*

    buys(X , "laptop") ⇒ buys(X , "HP printer")
- **In above rules items bought by customer X are referenced at different levels of abstraction, in which computer is a higher-level abstraction of laptop.**
- **Single-level association rules: When rules within a given set do not reference items or attributes at different levels of abstraction.**

Classification based on types of values handled in rule

- **Boolean association rule: When a rule involves associations between the presence or absence of items.**
  - *computer ⇒ antivirus software [support = 2%, confidence = 60%]*
- **Quantitative association rule: When a rule describes associations between quantitative items or attributes.**
- **In quantitative association rules, quantitative values for items or attributes are partitioned into intervals.**
  - *age(X , "30 . . . 39") ∧ income(X , "42K . . . 48K")*

    ⇒buys(X , "high_ resolution_TV")

Classification based on the kinds of rules to be mined

- **Association rules: Rules generated from frequent patterns**
- **Correlation rules: Uncover statistical correlations in discovered associations.**
- **Strong gradient relationships: Based on gradient.**
  - **Gradient is the ratio of measure of an item when compared with that of its parent (a generalized itemset), its child (a specialized itemset), or its sibling (a comparable itemset).**
  - **"The average sales from Sony Digital Camera increase over 16% when sold together with Sony Laptop Computer"**

Both Sony Digital Camera and Sony Laptop Computer are siblings, where the parent itemset is Sony.

Classification based on kinds of patterns to be mined

- **Frequent itemset mining: Mining of frequent itemsets from transactional or relational data sets.**
- **Sequential pattern mining: Searches for frequent subsequences in a sequence data set, where a sequence records an ordering of events.**
- **Structured pattern mining: Searches for frequent substructures in a structured data set such as graphs, lattices, trees, sequences, sets, single items, or combinations of such structures.**

Frequent Itemset Mining Methods: Apriori algorithm

- **Find Frequent Itemsets Using Candidate Generation.**
- **Apriori algorithm proposed by R. Agrawal and R. Srikant in 1994 for mining frequent itemsets for Boolean association rules.**
- **Apriori property is used to reduce the search space.**
- **This property belongs to a special category of properties called antimonotone where, if a set cannot pass a test, all of its supersets will fail the test as well.**
- **Apriori property:**

  "All nonempty subsets of a frequent itemset must also be frequent".

Apriori: A Candidate Generation-and-Test Approach

- **Initially, scan given database once to get frequent 1-itemset.**
- **Generate length (k+1) candidate itemsets from length k frequent itemsets.**
  - **The join step**: L_k, a set of candidate k-itemsets is generated by joining L_{k-1} with itself. This set of candidates is denoted C_k. C_k is a superset of L_k.
  - To reduce the size of Ck, the **Apriori property** is used.
- **Test the candidates against database.**
  - **The prune step:** To find L_k, candidates having a count less than minimum support are pruned.
- **Terminate when no frequent or candidate set can be generated.**

Generation of candidate itemsets and frequent itemset

If the minimum *confidence* is 50%, then the only two rules generated from this 2-itemset, {Shoes, Jacket} that have confidence greater than 50%, are:

Shoes ⇒ Jacket
Support=50%, Confidence=66%
Jacket ⇒ Shoes
Support=50%, Confidence=100%

Transaction ID	Items Bought
1	Shoes, Shirt, Jacket
2	Shoes,Jacket
3	Shoes, Jeans
4	Shirt, Sweatshirt

Frequent Itemset	Support
{Shoes}	75%
{Shirt}	50%
{Jacket}	50%
{Shoes, Jacket}	50%

Generation of candidate itemsets and frequent

Database D

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

Scan D

C_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{4}	1
{5}	3

L_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{5}	3

C_2

itemset
{1 2}
{1 3}
{1 5}
{2 3}
{2 5}
{3 5}

Scan D

C_2

itemset	sup
{1 2}	1
{1 3}	2
{1 5}	1
{2 3}	2
{2 5}	3
{3 5}	2

L_2

itemset	sup
{1 3}	2
{2 3}	2
{2 5}	3
{3 5}	2

Min support =50%

C_3

itemset
{2 3 5}

Scan D

L_3

itemset	sup
{2 3 5}	2

Apriori Algorithm

Input: D, a database of transactions; min sup, minimum support count threshold.

Output: L, frequent itemsets in D.

Method:

(1) L_1 = find frequent 1-itemsets(D);
(2) for (k = 2;L_{k-1} != ϕ; k++) {
(3) C_k = apriori_gen(L_{k-1});
(4) for each transaction t ∈ D {
(5) C_t = subset(C_k , t)
(6) for each candidate c ∈ C_t
(7) c.count++;
(8) }
(9) L_k= {c ∈ C_k |c.count ≥ min sup}
(10) }
(11) return L = $\cup_k L_k$

Apriori Algorithm

procedure apriori_gen (L_{k-1} :frequent (k − 1)-itemsets)

(1)for each itemset $l_1 \in L_{k-1}$

(2) for each itemset $l_2 \in L_{k-1}$

(3) if ($l_1[1] = l_2[1]$) ∧ ($l_1[2] = l_2[2]$) ∧ ... ∧ ($l_1[k-2] = l_2[k-2]$) ∧ ($l_1[k-1] < l_2[k-1]$) then {

(4) $c = l_1 \bowtie l_2$; // join step

(5) if has_infrequent_subset (c, L_{k-1}) then

(6) delete c; // prune step

(7) else add c to C_k ;

(8) }

(9) return C_k ;

Apriori Algorithm

procedure has_infrequent_subset (c: candidate k-itemset;

L_{k-1} : frequent (k – 1)-itemsets);

(1) for each (k – 1)-subset s of c

(2) if s !∈ L_{k-1} then

(3) return TRUE;

(4) return FALSE;

Generating Association Rules from Frequent Itemsets

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

Based on this equation, association rules can be generated as follows:

- For each frequent itemset *l*, generate all nonempty subsets of *l*.
- For every nonempty subset *s* of *l*, output the rule "$s \Rightarrow (l - s)$"
- if $\frac{support\_count(l)}{support\_count(s)} \geq min\ conf$,

where *min conf* is the minimum confidence threshold.

- **Because the rules are generated from frequent itemsets, it automatically satisfies minimum support.**

Improved Apriori algorithm: Hash-based technique

- **Hash-based technique can be used to reduce the size of the candidate k-itemsets, C_k , for k > 1.**
  - For example, when scanning each transaction in the database to generate the frequent 1-itemsets, L_1 , from the candidate 1-itemsets in C_1 , we can generate all of the 2-itemsets for each transaction, hash (i.e., map) them into the different buckets of a hash table structure, and increase the corresponding bucket counts.
  - Following hash table shows H2 using hash function h(x, y) = ((order of x) X 10 + (order of y)) mod 7

bucket Address	0	1	2	3	4	5	6
bucket count	2	2	4	2	2	4	4
bucket Contents	{I1, I4} {I3, I5}	{I1, I5} {I1, I5}	{I2, I3} {I2, I3} {I2, I3} {I2, I3}	{I2, I4} {I2, I4}	{I2, I5} {I2, I5}	{I1, I2} {I1, I2} {I1, I2} {I1, I2}	{I1, I3} {I1, I3} {I1, I3} {I1, I3}

Improved Apriori algorithm: Transaction reduction

- A transaction that does not contain any frequent k-itemsets cannot contain any frequent (k+1)-itemsets.
- Therefore, such a transaction can be marked or removed from further consideration.

Improved Apriori algorithm: Partitioning

- **Partitioning technique consists of two phases.**

  Phase I: Subdivides transactions of D into n non overlapping partitions.
  - For each partition, find all frequent itemsets called as **local frequent itemsets based** on minimum support.

  Phase II: Second scan of D is conducted to determine the global frequent itemsets.
  - The collection of frequent itemsets from all partitions forms the **global candidate itemsets** with respect to D.

- **Any itemset that is potentially frequent with respect to D must occur as a frequent itemset in at least one of the partitions.**

Improved Apriori algorithm: Partitioning

- Requires only two database scans to mine the frequent itemsets.
- Any itemset that is potentially frequent with respect to D must occur as a frequent itemset in at least one of the partitions.
- Partition size and number of partitions are set so that each partition can fit into main memory and therefore read operation requires only once in each phase.

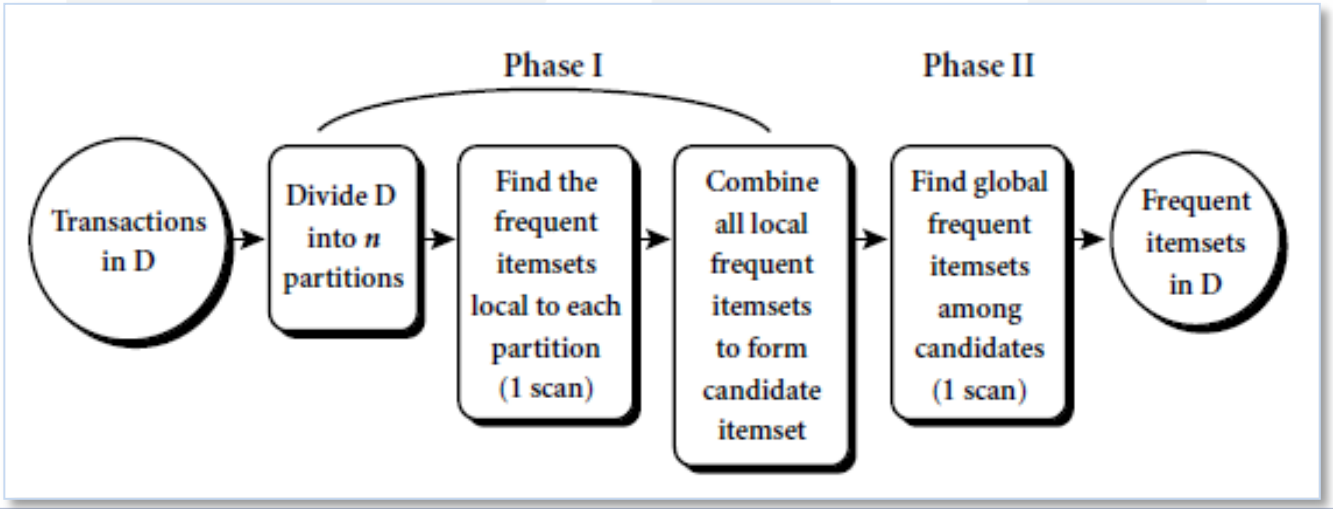


Image source : Data Mining Concepts and Techniques Book

Improved Apriori algorithm: Sampling

- Choose a random sample S from given data D, and then search for frequent itemsets in S instead of D.
- The sample size of S is such that search for frequent itemsets in S can be done in main memory.
- Use a lower support threshold than overall minimum support to find the frequent itemsets local to S denoted by L^S.
- The rest of the database is then used to compute actual frequencies of each itemset in L^S.
- If L^S contains all frequent itemsets in *D*, then only one scan of *D* is required.
- Otherwise, a second pass can be done, to find frequent itemsets that were missed in the first pass.

Improved Apriori algorithm: Dynamic itemset counting

- The database is partitioned into blocks marked by start points.
- In this variation, new candidate itemsets can be added at any start point
- This technique is dynamic, because it estimates support of all of itemsets that have been counted so far, add new candidate itemsets if all of their subsets are estimated to be frequent.
- The resulting algorithm requires fewer database scans than Apriori.

Mining Frequent Itemsets without Candidate Generation

- **Adopts a divide-and-conquer strategy.**
- **Solves the problems of Apriori candidate generate-and-test method:**
  - **It may need to generate a huge number of candidate sets.**
  - **It may need to repeatedly scan the database and check a large set of candidates by pattern matching.**

FP-Growth Algorithm

Input: D, a database of transactions; min sup, minimum support count threshold.

Output: The complete set of frequent patterns

- **Build a compact data structure called the FP-tree using 2 passes over data-set.**

Pass1:

- **Scan data and find support for each item.**
- **Discard infrequent items.**
- **Sort frequent items in decreasing order based on their support.**

Pass2: Construct the Frequent Pattern tree.

- **Create root of an FP-tree, and label it as "null."**
- **Let the sorted frequent item list in transaction be [pIP], where p is the first element and P is the remaining list.**
- **Call insert tree([pIP], T)**

FP-Growth Algorithm

- **insert tree([pIP], T) Procedure:**
  - If T has a child N such that N.item-name=p.item-name,
  - Then increment N's count by 1,
  - Else create a new node N, and let its count be 1, its parent link be linked to T, and its node-link to nodes with same item-name via the node-link structure.
  - If P is nonempty, call insert tree(P, N) recursively.
- **Next step is to extracts frequent itemsets directly from FP-tree through tree traversal.**
  - The FP-tree is mined by calling FP-growth(FP tree, null).

FP-Growth Algorithm

procedure FP-growth(Tree, α)

(1) if Tree contains a single path P then

(2) for each combination (denoted as β) of nodes in 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 $Tree_\beta$!= ϕ then

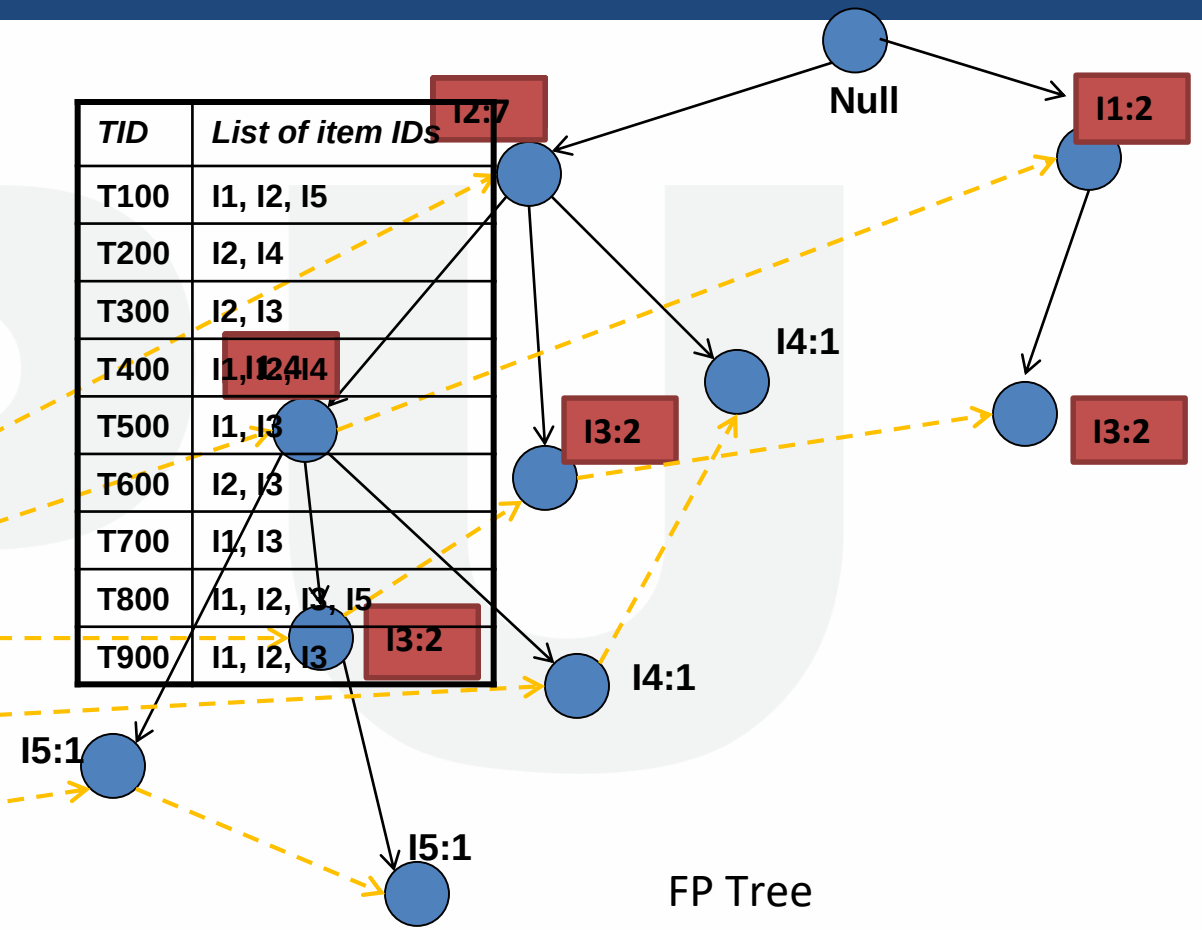
(8) call FP growth($Tree_\beta$, β); }

FP-Growth or frequent-pattern growth Algorithm

Item ID	Support count	Node-link
I2	7	
I1	6	
I3	6	
I4	2	
I5	2	

Item Header Table

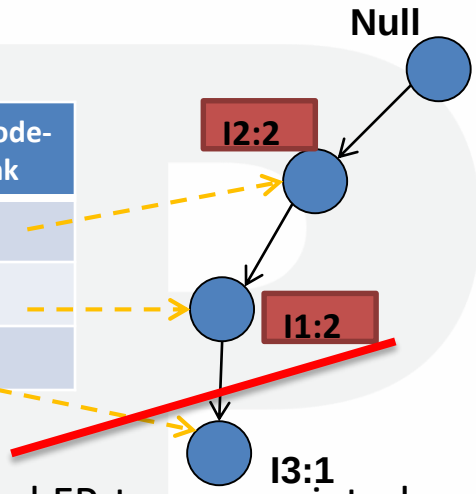
TID	List of item IDs
T100	I1, I2, I5
T200	I2, I4
T300	I2, I3
T400	I1, I2, I4
T500	I1, I3
T600	I2, I3
T700	I1, I3
T800	I1, I2, I3, I5
T900	I1, I2, I3



FP Tree

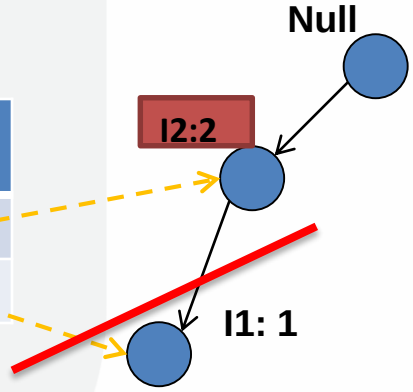
FP-Growth or frequent-pattern growth Algorithm

Item ID	Support count	Node-link
I2	2	
I1	2	
I3	1	



The Conditional FP-tree associated with the conditional node I5.

Item ID	Support count	Node-link
I2	2	
I1	1	

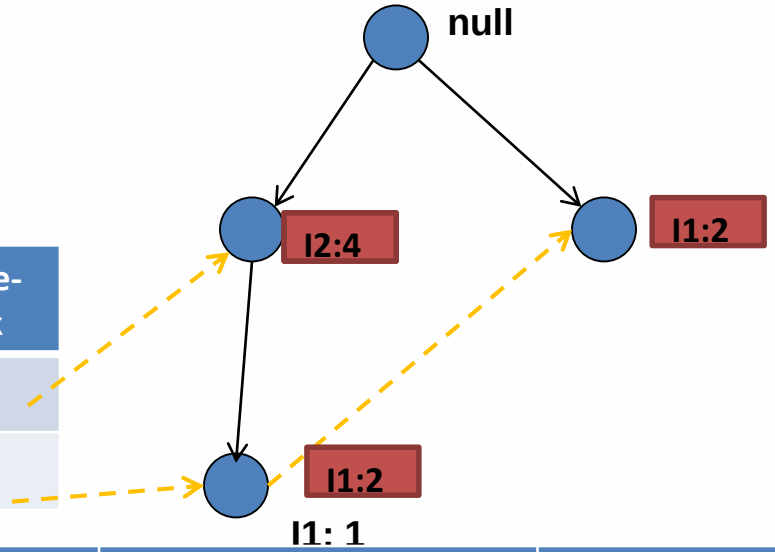


Conditional FP-tree associated with the conditional node I4.

Item	Conditional Pattern Base	Conditional FP-tree	Frequent Patterns Generated
I5	{{I2, I1: 1}, {I2, I1, I3: 1}}	<I2: 2, I1: 2>	{I2, I5: 2}, {I1, I5: 2}, {I2, I1, I5: 2}
I4	{{I2, I1: 1}, {I2: 1}}	<I2: 2>	{I2, I4: 2}

FP-Growth or frequent-pattern growth Algorithm

Item ID	Support count	Node-link
I2	4	
I1	4	



Item	Conditional Pattern Base	Conditional FP-tree	Frequent Patterns Generated
I5	{{I2, I1: 1}, {I2, I1, I3: 1}}	<I2: 2, I1: 2>	{I2, I5: 2}, {I1, I5: 2}, {I2, I1, I5: 2}
I4	{{I2, I1: 1}, {I2: 1}}	<I2: 2>	{I2, I4: 2}
I3	{{I2, I1: 2}, {I2: 2}, {I1: 2}}	<I2: 4, I1: 2>, <I1: 2>	{I2, I3: 4}, {I1, I3: 4}, {I2, I1, I3: 2}

FP-Growth or frequent-pattern growth Algorithm

Final Table of Mining the FP-tree by creating conditional pattern bases.

Item	Conditional Pattern Base	Conditional FP-tree	Frequent Patterns Generated
I5	{{I2, I1: 1}, {I2, I1, I3: 1}}	<I2: 2, I1: 2>	{I2, I5: 2}, {I1, I5: 2}, {I2, I1, I5: 2}
I4	{{I2, I1: 1}, {I2: 1}}	<I2: 2>	{I2, I4: 2}
I3	{{I2, I1: 2}, {I2: 2}, {I1: 2}}	<I2: 4, I1: 2>, <I1: 2>	{I2, I3: 4}, {I1, I3: 4}, {I2, I1, I3: 2}
I1	{{I2: 4}}	<I2: 4>	{I2, I1: 4}

Characteristics of FP-Growth

- Only 2 passes over data set
- Compresses data-set
- No candidate generation
- Much faster than Apriori algorithm
- FP-Tree may not fit in memory
- FP-Tree is expensive to build

Correlation analysis

- The support and confidence measures are insufficient at filtering out uninteresting association rules.
  - Strong Rules are not necessarily interesting.
  - Many of rules are redundant as well.
- To tackle this weakness, a correlation measure can be used to augment the support-confidence framework for association rules.
- This leads to correlation rules of the form
  - A -> B [support, confidence. correlation].

correlation measures : Lift

- **The occurrence of itemset A is independent of the occurrence of itemset B, if P (A ∪ B) = P(A) P(B);**
- **otherwise, itemsets A and B are dependent and correlated.**
- **The lift between occurrence of A and B or correlation can be measured by:**

$$lift(A, B) = \frac{P(A \cup B)}{P(A)\,P(B)}$$

  - **lift(A,B) >1 means that A and B are positively correlated.**
  - **lift(A,B) < 1 means that the occurrence of A is negatively correlated.**
  - **lift(A,B) =1 means that A and B are independent and there is no correlation between them.**

correlation measures : Lift

- **The correlation formula can be re-written as:**
  - **lift(A,B) = P(B|A) / P(B) = Confidence(A-> B) / support(B)**
  - **Confidence(A-> B)= lift(A,B) P(B).**
- **Example: Association Rule: Tea -> Coffee**

	Coffee	$\overline{\text{Coffee}}$	
Tea	15	5	20
$\overline{\text{Tea}}$	75	5	80
	90	10	100

Confidence = P(Coffee|Tea) = 0.75
Support = P(Coffee) = 0.9
Then Lift = 0.75/0.9= 0.8333 (< 1v)
So this rule is negatively associated.

DIGITAL LEARNING CONTENT

Parul® University

www.paruluniversity.ac.in