UNIT-IV

Association Analysis:-

Topic-1

→ Basic concepts and algorithms

→ Definition

Topic-2

Frequent Item Generation:-

→ The Apriori principle

→ Frequent Itemset Generation in the Apriori algorithm.

→ Candidate Generation and pruning.

→ Support Counting

→ Computational Complexity

Topic-3

→ Compact Representation of Frequent Itemsets

→ Maximal Frequent Item sets

→ Closed Frequent Item sets.

Topic-4

Alternative Methods for Generating Frequent Item sets

→ FP-Growth Algorithm

→ FP-Tree Representation

→ Frequent Itemset Generation in FP-Growth Algorithm.

Topic-5

Evaluation of Association Patterns

→ Objective Measures of Interestingness

→ Measures beyond pairs of Binary Variables

→ Simpson's Paradox

→ Association Analysis:-

This is a methodology useful for discovering interesting relationships hidden in large data sets. The uncovered relationships can be represented in the form of association rules or sets of frequent items. For Eg, the following rule can be extracted from the data set, as shown below:

Table ~~Fig~~: 6.1

Consider Market basket data as Data set :-

TID Items

1.

Table 6.1. An example of market basket transactions.

FIG: 6.1

2.

TID	Items
1	{Bread, Milk}
2	{Bread, Diapers, Beer, Eggs}
3	{Milk, Diapers, Beer, Cola}
4	{Bread, Milk, Diapers, Beer}
5	{Bread, Milk, Diapers, Cola}

3.

The rule suggests that a strong relationship exists between the sale of diapers and beer because many customers who buy diapers also buy beer.

$$\{\text{Diapers}\} \rightarrow \{\text{Beer}\}$$

Retailers can use this type of rules to help them identify new opportunities for cross-selling their products to the customers.

Problem Definition :-

The basic terminology used in association analysis and presents a formal discription of the task as follows:

Binary Representation :-

Market basket data can be represented in a binary format as shown in Table 6.2. where each row corresponds to a transaction and each column corresponds to an item.

Table 6.2 A binary 0/1 representation of market basket data.

TID	Bread	Milk	Diapers	Beer	Eggs	Cola
1	1	1	0	0	0	0
2	1	0	1	1	1	0
3	0	1	1	1	0	1
4	1	1	1	1	0	0
5	1	1	1	0	0	1

An item can be treated as binary variable whose value is one if the item is present in a transaction and zero otherwise. Because the presence of an item in a transaction is often considered more important than its absence, an item is an asymmetric binary variable

Item Set and Support Count :-

Let $I = \{i_1, i_2, \ldots i_d\}$ be the set of all items in a market basket data and $T = \{t_1, t_2 \ldots t_N\}$ be the set of all transactions. Each transaction t_i contains a set of subset of items chosen from I. If an itemset

contains k items, it is called a k-itemset. For instance, {Bread, Milk, Diaper} is an example of 3-item set. The null (or empty) set is an itemset that does not contain any items.

Association Rule:-

An association rule is an implication expression of the form $X \rightarrow Y$, where X & Y are disjoint item sets.

i.e; $X \cap Y = \phi$. The strength of an association rule can be measured in terms of its support and confidence.

Support determines how often a rule is applicable to a given data set, while confidence determines how frequently items in Y appear in transactions that contain X. The formal definitions of these metrics are:

$$\text{Support}, s(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{N};$$

$$\text{Confidence}, c(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(X)}.$$

Eg:- Consider the rule {Milk, Diapers} → {Beer}. Since the support count for {Milk, Diapers, Beer} is 2 & the total no. of transactions is 5,

the rule's support is $2/5 = 0.4$.

The rule's confidence is obtained by dividing the support count for {Milk, Diapers, Beer} by the support count for {Milk, Diapers}. Since, there are 3 transactions that contain milk & diapers &

The confidence for this rule is $2/3 = 0.67$.

Formulation of Association Rule Mining Problem: The association rule mining problem can be formally stated as follows:

Definition

Association Rule Discovery:-

Given a set of transactions T, find all the rules having support ≥ minsup and confidence ≥ minconf, where minsup and minconf are the corresponding support and confidence thresholds.

A brute force approach for mining association rules is to compute the support and confidence for every possible rule. This approach is prohibitively expensive because there are many rules that can be extracted from a data set that contains d items is

$$R = 3^d - 2^{d+1} + 1$$

Eg: d = no. of items in the table 6.1

Topic-2

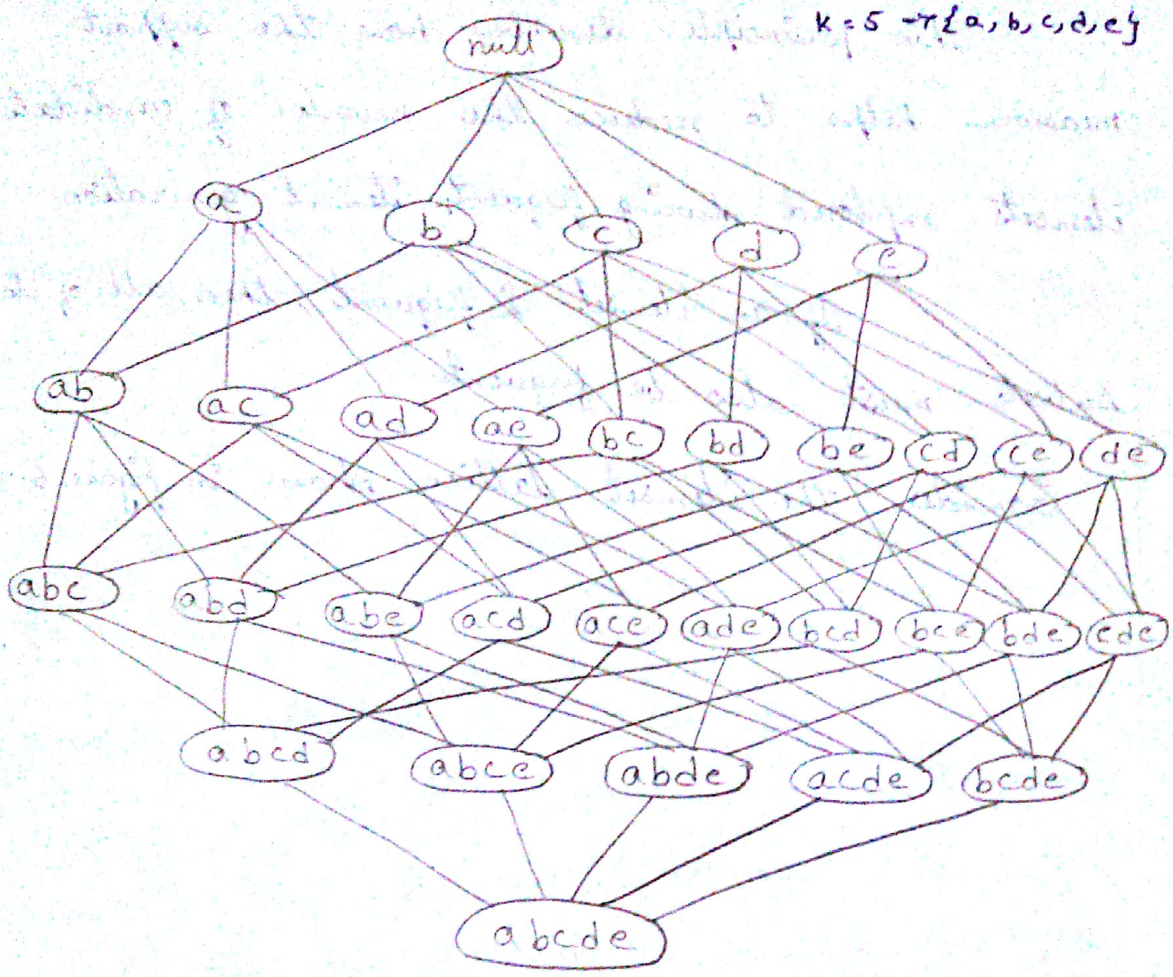
Frequent Itemset Generation:-

This is to find all the itemsets that satisfy the minsup threshold. These itemsets are called frequent itemsets. We can use lattice structure to enumerate the list of all possible itemsets.

Figure 6.1 shows an itemset lattice for $I = \{a, b, c, d, e\}$. In general, a data set that contains k items can generate upto $2^k - 1$ frequent itemsets, excluding the null set.

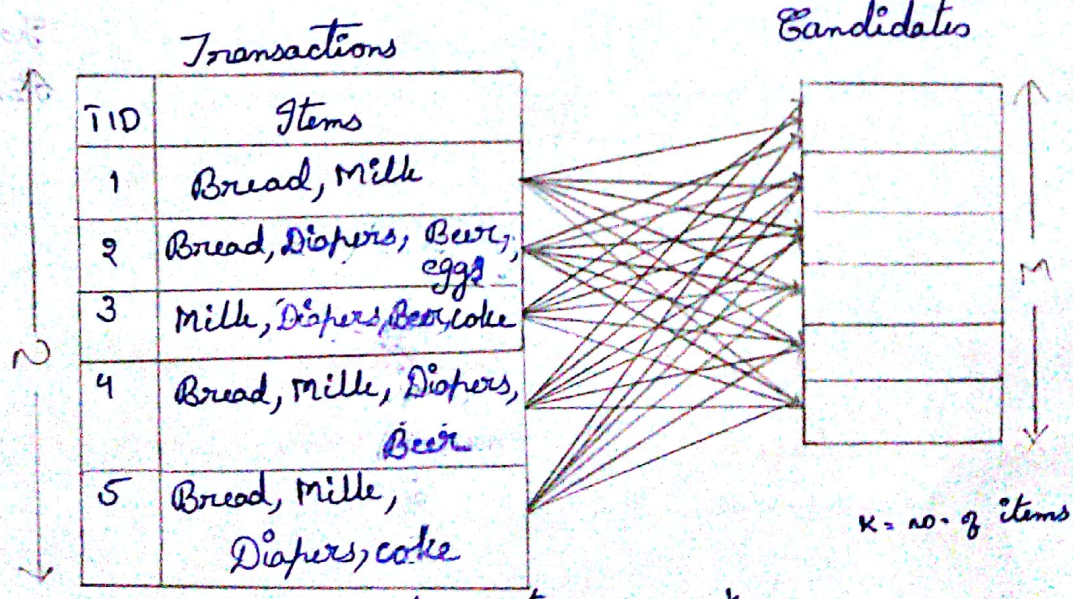
A brute force approach for finding frequent itemset is to determine the support count for every candidate itemset in the lattice structure. To do this, we

Figure 6.1 An itemset lattice



need to compare each candidate against every transaction as shown below: fig: 6.2.

fig: 6.2 Counting the support of candidate itemsets



N = no. of transactions, $M = 2^k - 1$

M = no. of candidate itemsets

→ The Apriori Principle :-

This principle describes how the support measure helps to reduce the number of candidate itemsets explored during frequent itemset generation.

Theorem 6.1 : If an itemset is frequent, then all of its subsets must also be frequent.

Consider the itemset lattice shown in figure 6.3.

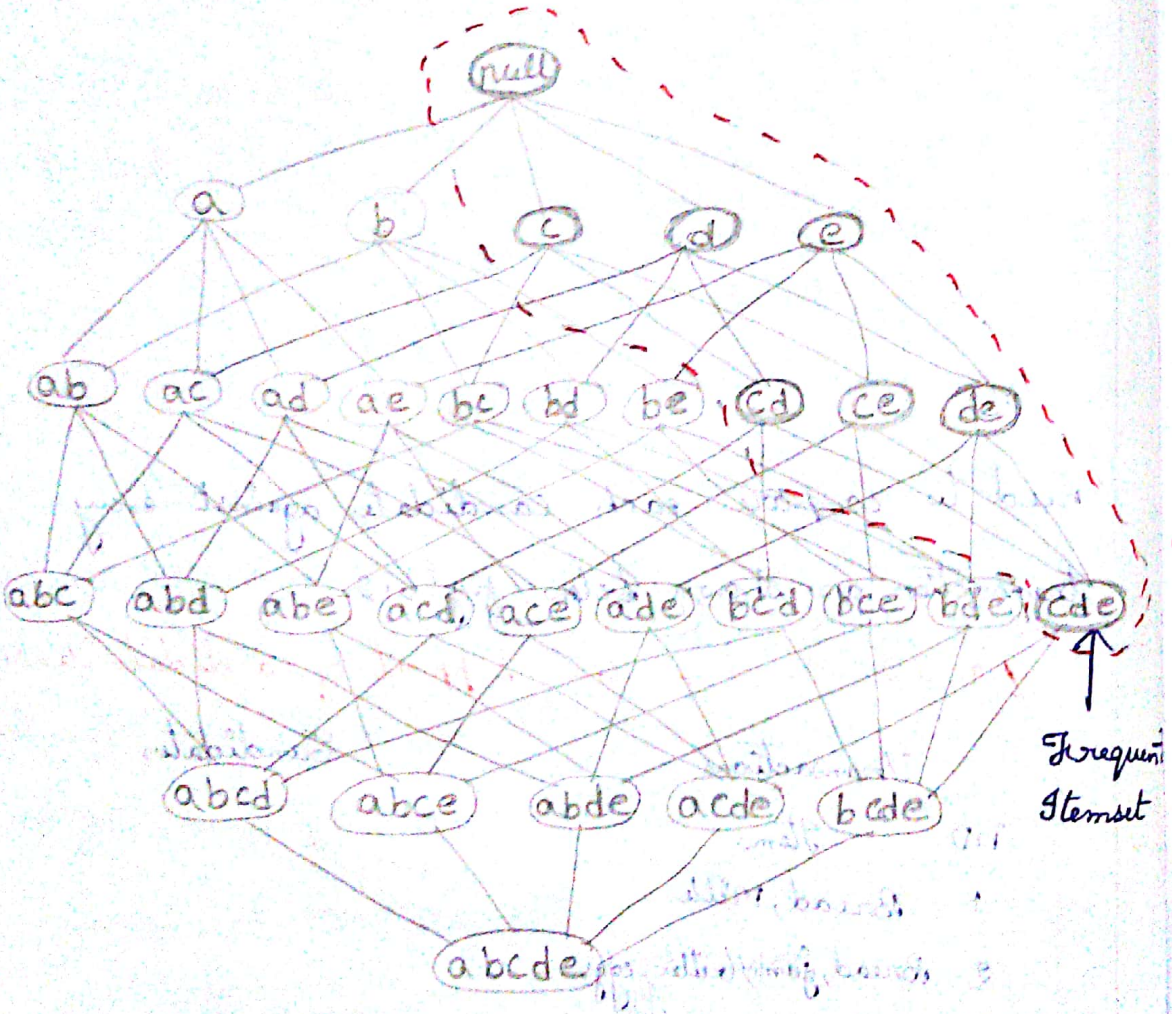


Fig 6.3 An Illustration of the Apriori principle. If {c, d, e} is frequent, then all subsets of this itemset are frequent.

Suppose {c, d, e} is a frequent itemset. Any transaction that contains {c, d, e} must also contain its subsets, {c, d}, {c, e}, {d, e}, {d} and {e}.

As a result if {c, d, e} is frequent, then all subsets of {c, d, e} (i.e; shaded itemsets in figure) must also be frequent.

Suppose, if an itemset {a, b} is infrequent then all of its supersets must be infrequent too. So, the exponential search space pruning or trimming, to be done for entire subgraph containing supersets of {a, b} is known as support-based pruning which is possible by a key property called anti-monotone property.

6.2.2 → Frequent Itemset Generation in the Apriori Algorithm:-

Apriori is the first association rule mining algorithm that pioneered the use of support based pruning to systematically control the exponential growth of candidate itemsets.

Consider the figure: 6.5 provides a high level illustration of the frequent itemset generation part of the Apriori algorithm for the transactions shown in Table 6.1 (Market Basket Transaction)

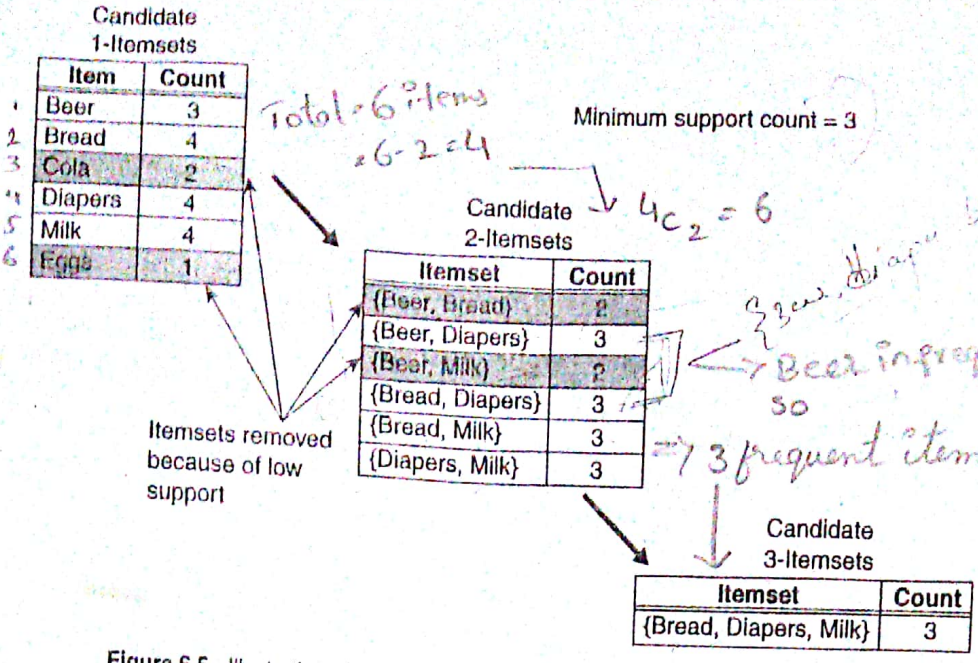


Figure 6.5. Illustration of frequent itemset generation using the *Apriori* algorithm.

The effectiveness of the Apriori pruning strategy can be shown by counting the number of candidate itemsets generated.

A brute-force strategy of enumerating all itemsets (upto size 3) as candidates will produce:

$$\binom{6}{1} + \binom{6}{2} + \binom{6}{3} = 6 + 15 + 20 = 41$$

candidates.

$\binom{6}{1} \Rightarrow {}^6C_1 = 6$

$\binom{6}{2} \Rightarrow {}^6C_2 = \frac{6 \times 5}{2}$

$\binom{6}{3} \Rightarrow {}^6C_3 = \frac{6 \times 6 \times 4}{3 \times 2}$

With the Apriori principle, this number decreases to

$$\binom{6}{1} + \binom{4}{2} + 1 = 6 + 6 + 1 = 13 \text{ candidates.}$$

The Apriori Algorithm for the frequent itemset generation is as follows:

Algorithm 6.1 Frequent itemset generation of the Apriori algo.

Algorithm 6.1 Frequent itemset generation of the *Apriori* algorithm.

1: $k = 1$.
2: $F_k = \{ i \mid i \in I \wedge \sigma(\{i\}) \geq N \times minsup \}$. {Find all frequent 1-itemsets}
3: **repeat**
4: $k = k + 1$.
5: $C_k = \text{apriori-gen}(F_{k-1})$. {Generate candidate itemsets}
6: **for** each transaction $t \in T$ **do**
7: $C_t = \text{subset}(C_k, t)$. {Identify all candidates that belong to t}
8: **for** each candidate itemset $c \in C_t$ **do**
9: $\sigma(c) = \sigma(c) + 1$. {Increment support count}
10: **end for**
11: **end for**
12: $F_k = \{ c \mid c \in C_k \wedge \sigma(c) \geq N \times minsup \}$. {Extract the frequent k-itemsets}
13: **until** $F_k = \emptyset$
14: Result $= \bigcup F_k$.

2 important characteristics in the frequent itemset generation part of the Apriori algorithm.

1) Level-wise algorithm LWA
2) Generate and Test strategy G&TS

LWA 1) It traverses the itemset lattice one level at a time, from frequent 1-itemsets

G&TS 2) This strategy is for finding frequent itemsets. At each iteration, new candidate itemsets are generated from the frequent itemsets found in the previous iteration.

→ **Candidate Generation and Pruning :-**

The apriori-gen function shown in Step 5 of Algorithm 6.1 generates candidate itemsets by performing the following two operations :

1) Candidate Generation :- This operation generates new candidate k-itemsets based on the frequent $(k-1)$ itemsets found in the previous iteration.

2) Candidate Pruning :- This operation eliminates some of the candidate k-itemsets using the support-based pruning strategy.

The following is a list of requirements for an effective candidate generation procedure:

→ 1. It should avoid generating too many unnecessary candidates. A candidate itemset is unnecenary if atleast one of its subsets is infrequent. According to anti monotone property of support such candidate is said to be infrequent

→ 2. It must ensure that the candidate set is complete. To ensure completeness, the set of candidate itemsets must subsume the set of all frequent itemsets i.e; $\forall k : F_k \subseteq C_k$

→ 3. It should not generate the same candidates itemsets more than once.

For Eg, the candidate itemset $\{a, b, c, d\}$ can be generated in many ways - by merging $\{a, b, c\}$ with $\{d\}$

{b,d} with {a,c}, {c} with {a,b,d} etc

The following are the candidate generation procedures including the one followed by the apriori-gen function:

→ Brute Force Method:-

The Brute-force method considers every k-itemset as a potential candidate and then applies the candidate pruning step to remove any unnecessary candidates as shown in fig 6.6

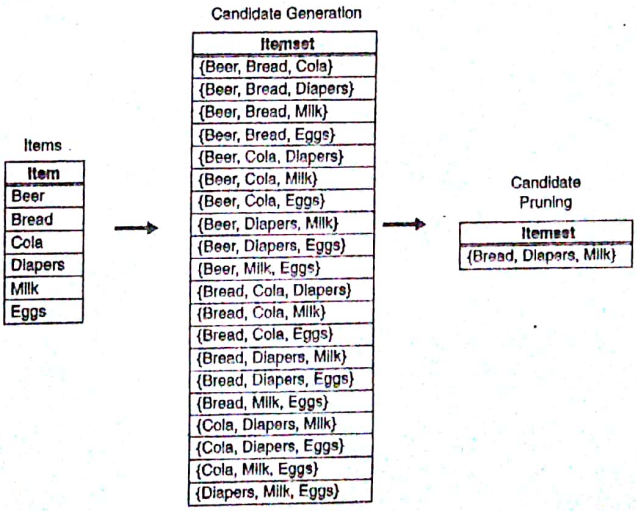


Figure 6.6. A brute-force method for generating candidate 3-itemsets.

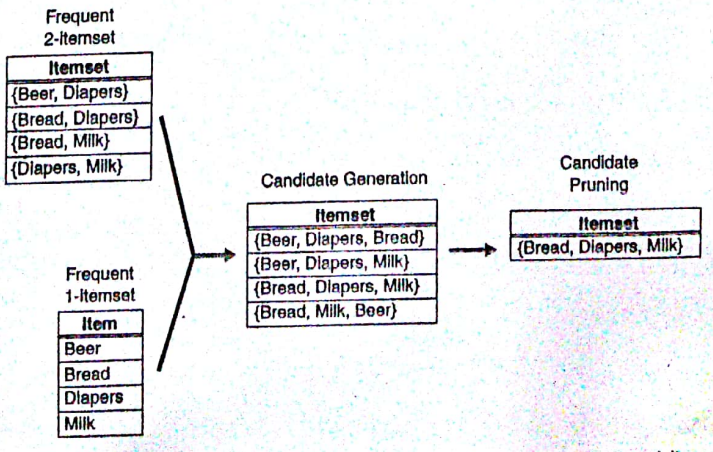


Figure 6.7. Generating and pruning candidate k-itemsets by merging a frequent $(k-1)$-itemset with a frequent item. Note that some of the candidates are unnecessary because their subsets are infrequent.

→ $F_{k-1} \times F_1$ Method :-

→ An Alternative method for candidate generation is to extend each frequent $(k-1)$ itemset with other frequent items as shown in fig 6.7

→ The procedure is complete because every frequent k-itemset is composed of a frequent $(k-1)$ itemset and a frequent 1-itemset.

→ This approach does not prevent the same candidate itemset from being generated more than once. To avoid duplicate candidates, the items in each frequent itemset are kept sorted in lexicographical order.

Eg: {Bread, Diapers, Milk} can be generated by merging {Bread, Diapers} with {Milk}, {Bread, Milk} with {Diapers} & {Diapers, Milk} with {Bread}.

→ $F_{k-1} \times F_{k-1}$ Method :-

The candidate generation procedure in the apriori-gen function merges a pair of frequent $(k-1)$ itemsets only if their first $k-2$ items are identical. Every when pruning step is included to avoid duplicate candidate itemsets as shown in fig 6.8

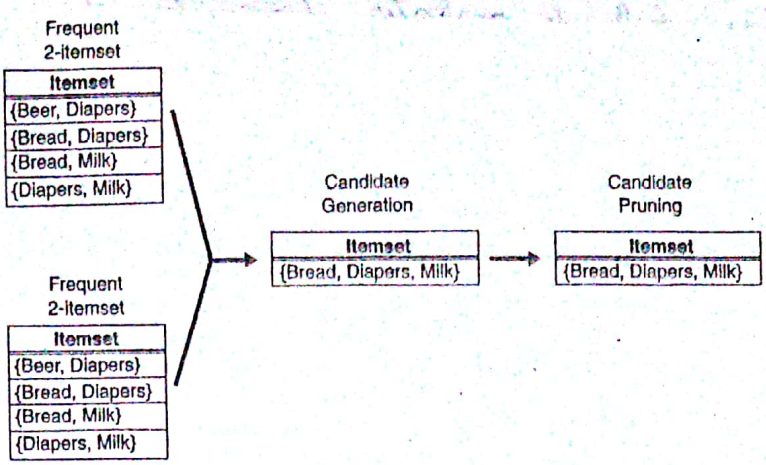


Figure 6.8. Generating and pruning candidate k-itemsets by merging pairs of frequent $(k-1)$-itemsets.

6.2.4 Support Counting :-

Support Counting is the process of determining the frequency of occurrence for every candidate itemset that survives the candidate pruning step of the apriori-gen function. Support Counting is implemented in step 6 through 11 of Algorithm 6.1.

→ One approach for doing this is to compare each transaction against every candidate itemset and to update the support counts of candidates contained in the transaction.

Consider an example,

Figure 6.9

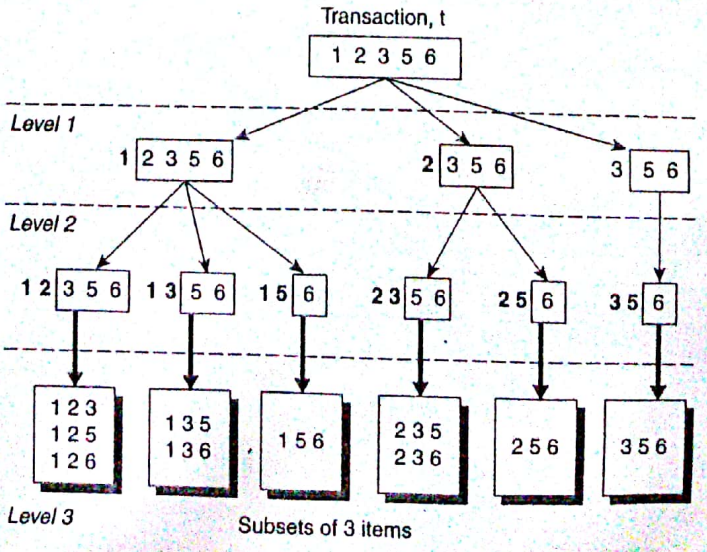


Figure 6.9. Enumerating subsets of three items from a transaction t.

Figure 6.9 demonstrates how itemsets contained in a transaction can be systematically enumerated.

⇒ i.e; by specifying their items one by one, from the leftmost item to the rightmost item. If it matches one of the candidates, then the support count of the corresponding candidate is incremented.

→ Matching Operation can be performed efficiently using a hash tree structure.

Figure 6.10 Counting the support of itemsets using hash structure

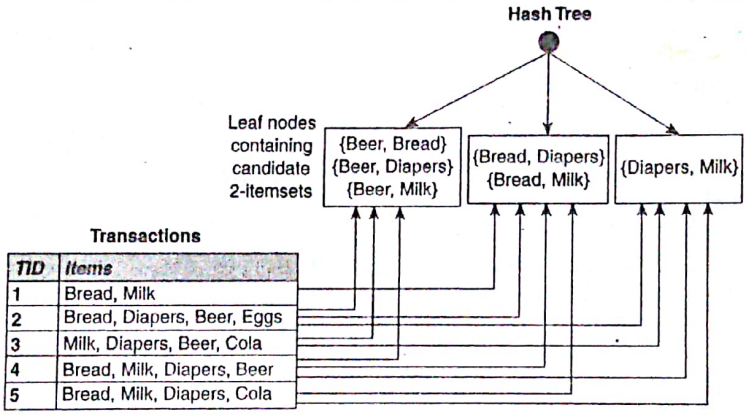


Figure 6.10. Counting the support of itemsets using hash structure.

6.2.5 Computational Complexity :-

The computational complexity of the Apriori algorithm can be affected by the following factors:

1) Support Threshold

2) Number of Items (Dimensionality)

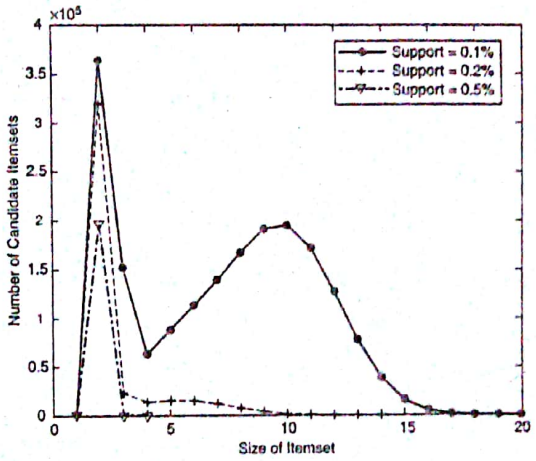
3) Number of Transactions

4) Average Transaction Width

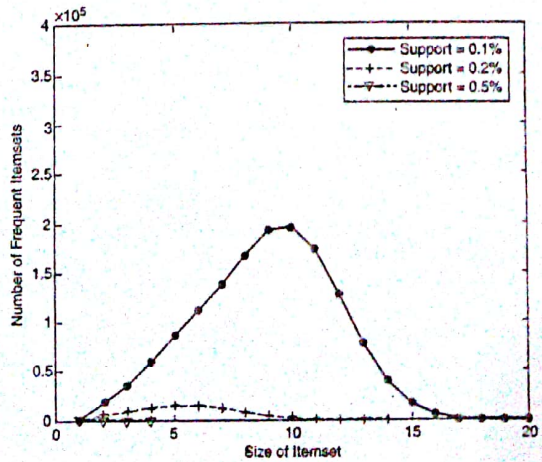
1) Support Threshold :-

Lowering the support threshold results in more itemsets being declared as frequent.

Before knowing computational complexity, let us consider a hash at different levels.



(a) Number of candidate itemsets.



(b) Number of frequent itemsets.

Figure 6.13. Effect of support threshold on the number of candidate and frequent itemsets.

This has an adverse effect on computational complexity of the algorithm because more candidate itemsets must be generated and counted. ie fig. 6.13

2) Number of Items (Dimensionality):-

As the no. of items increases more space will be needed to store support counts of no. of items. If the frequent items grows with dimensionality of data, the computation and I/O costs will increase.

3) Number of Transactions:-

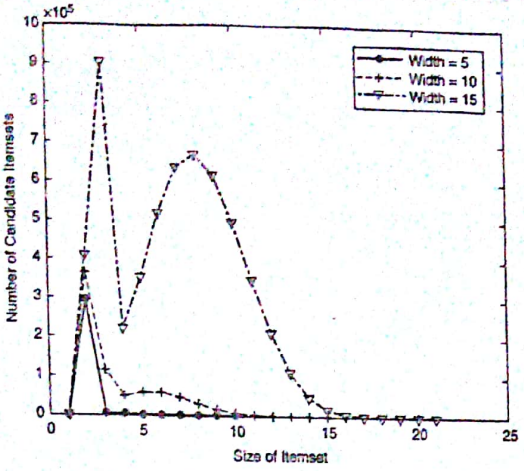
Since the Apriori algorithm makes repeated passes over the data set, its run time increases with larger number of transactions.

4) Average Transaction Width:-

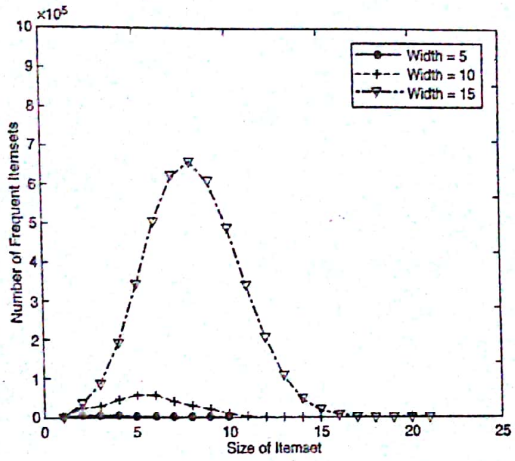
For dense data sets, the average transaction width can be very large. This affects the complexity of the Apriori algorithm in 2 ways.

→ 1) The maximum size of the frequent itemsets tends to increase as the average transaction width increases. As a result, more candidate itemsets must be examined during candidate generation & support counting. ie fig. 6.14

→ 2) As the transaction width increases, more



(a) Number of candidate itemsets.



(b) Number of Frequent Itemsets.

Figure 6.14. Effect of average transaction width on the number of candidate and frequent itemsets.

itemsets are contained in the transaction. This will increase the no. of hash tree traversals performed during support counting.

→ Maximal Frequent Itemsets :-

A Maximal Frequent Itemsets is defined as a frequent itemset for which none of its immediate supersets are frequent.

Shown below in fig 6.16

Consider the itemset lattice for this concept as the itemsets in the lattice are divided into two groups those that are :-

1) frequent
2) infrequent

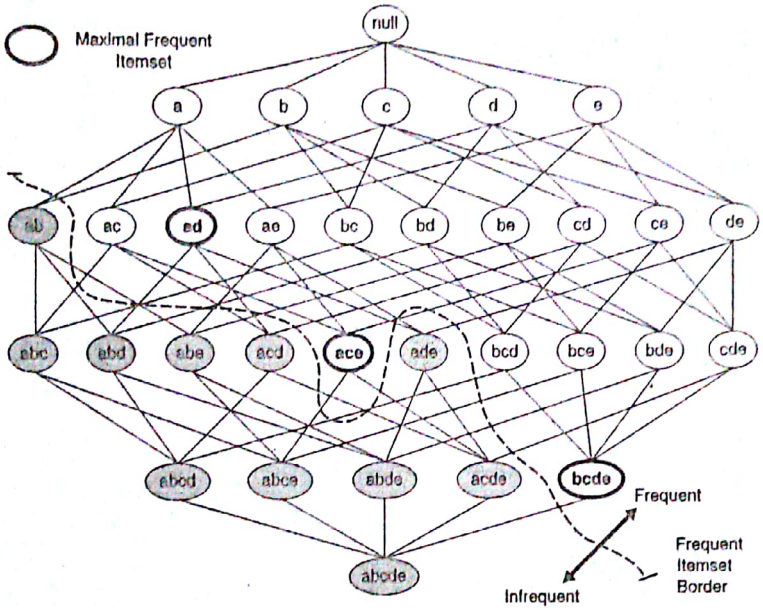


Figure 6.16. Maximal frequent itemset.

A frequent itemset border which is represented by a dashed line. Every itemset located above the border is frequent, while those located below the border are infrequent. Among the itemsets residing near the border, {a, d}, {a, c, e} and {b, c, d, e} are considered to be maximal frequent itemsets because their immediate supersets are infrequent.

This approach is practical only if an efficient algorithm exists to explicitly find the maximal frequent itemsets without having to enumerate all their subsets, which. It does not contain any support information.

6.4.2 Closed Frequent Itemsets

Closed itemsets provide a minimal representation of itemsets without losing their support information. A formal definition of a closed itemset is presented below:- **Definition 1:**

An itemset X is closed if none of its immediate supersets has exactly the same support count as X.

To better illustrate the support count of each itemset, we have associated each node (itemset) in the lattice with a list of its corresponding transaction ID's as shown in below fig. 6.17

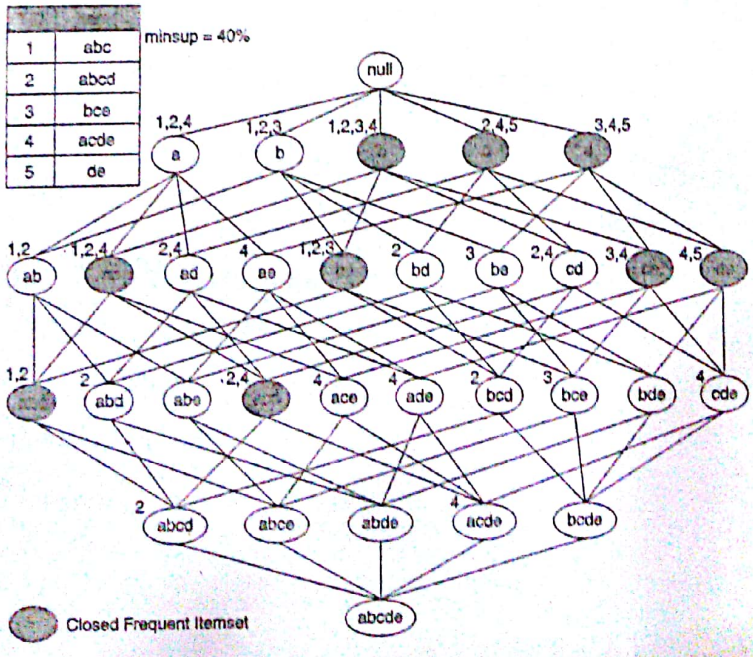


Figure 6.17. An example of the closed frequent itemsets (with minimum support count equal to 40%).

For eg, since the node {b,c} is associated with transaction ID's 1, 2 & 3 its support count is equal to three. Consequently, the support for {b} is identical to {b,c} and {b} is not a closed itemset.

Definition 2 :-

An itemset is a closed frequent itemset if it is closed and its support is greater than or equal to minsup.

Algorithm 6.4 to extract closed frequent itemsets from a given data set and to compute the support for the non-closed frequent itemsets. This is used to proceed from largest to smaller frequent item sets.

Algorithm 6.4 Support counting using closed frequent itemsets.

1: Let C denote the set of closed frequent itemsets
2: Let $k_{\max}$ denote the maximum size of closed frequent itemsets
3: $F_{k_{\max}} = \{f | f \in C, |f| = k_{\max}\}$ {Find all frequent itemsets of size $k_{\max}$.}
4: **for** $k = k_{\max} - 1$ downto 1 **do**
5: $F_k = \{f | f \subset F_{k+1}, |f| = k\}$ {Find all frequent itemsets of size k.}
6: **for** each $f \in F_k$ **do**
7: **if** $f \notin C$ **then**
8: $f.support = \max\{f'.support | f' \in F_{k+1}, f \subset f'\}$
9: **end if**
10: **end for**
11: **end for**

To illustrate the advantage of using closed frequent itemsets, consider the table 6.5 include data set. The total number of frequent itemsets is $3 \times (2^5 - 1) = 93$.

Table 6.5. A transaction data set for mining closed itemsets.

TID	a_1	a_2	a_3	a_4	a_5	b_1	b_2	b_3	b_4	b_5	c_1	c_2	c_3	c_4	c_5
1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
2	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
3	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
5	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
6	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
8	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
9	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
10	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1

Closed frequent itemsets are useful for removing some of the redundant association rules.

→ An association rule $X \rightarrow Y$ is redundant if there exists another rule $X' \rightarrow Y'$,

where X is a subset of X' and
Y is a " " Y'

such that the support and confidence for both rules are identical.

Finally, note that all maximal frequent itemsets are closed because none of the maximal freq. itemsets can have the same support count as their immediate supersets. The relationships among frequent, maximal and closed frequent itemsets are shown in fig 6.18

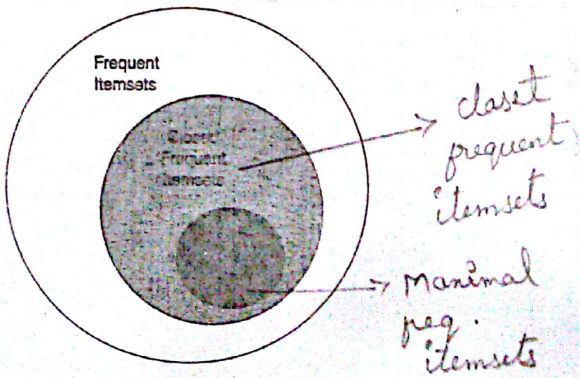


Figure 6.18. Relationships among frequent, maximal frequent, and closed frequent itemsets.

TOPIC-4

Alternative Methods for Generating Frequent Itemsets :-

Apriori is one of the earliest algorithms to have successfully addressed the combinational explosion of frequent itemset generation. It achieves this by applying the Apriori principle to prune the exponential search space.

Several alternative methods have been developed and improved upon the efficiency of the Apriori algorithm.

The following is a high level description of these methods.

Traversal of Itemset Lattice :-

→ A search for frequent itemsets can be conceptually viewed as a traversal on the itemset lattice as shown in fig 6.1.

→ The search strategy employed by an algorithm dictates how the lattice structure is traversed during the frequent itemset generation process.

→ Some search strategies are better than others, depending on the configuration of frequent itemsets in the lattice

→ An overview of these strategies are discussed below :-

→ General-to-Specific versus Specific-to-General :-

a → The Apriori algorithm uses a general-to-specific

search strategy, where pairs of frequent (k-1) itemsets are merged to obtain candidate k-itemsets. This strategy is effective and works best providing the maximum length of a frequent itemset is not too long. The configuration is shown in fig 6.19(a) where darker nodes represents infrequent item-sets.

b → The specific to general search strategy looks for more specific frequent itemsets first, before finding the more general frequent itemsets.

→ This strategy is useful to discover maximal frequent itemsets in dense transactions, where the frequent itemset border is located near the bottom of lattice as shown in fig 6.19(b)

c → The Apriori principle can be applied to prune all subsets of maximal frequent itemsets. Another approach is to combine both a & b steps. This bidirectional approach requires more space to store the candidate itemsets and helps in identifying the frequent itemset border fastly. as shown in fig 6.19(c)

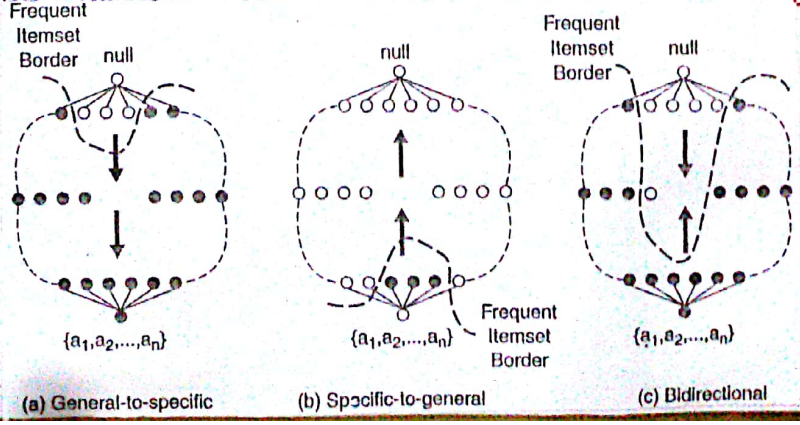


Fig: 6.19

a) General-Specific
b) Specific-General
c) Bidirectional Search

Fig: 6.22

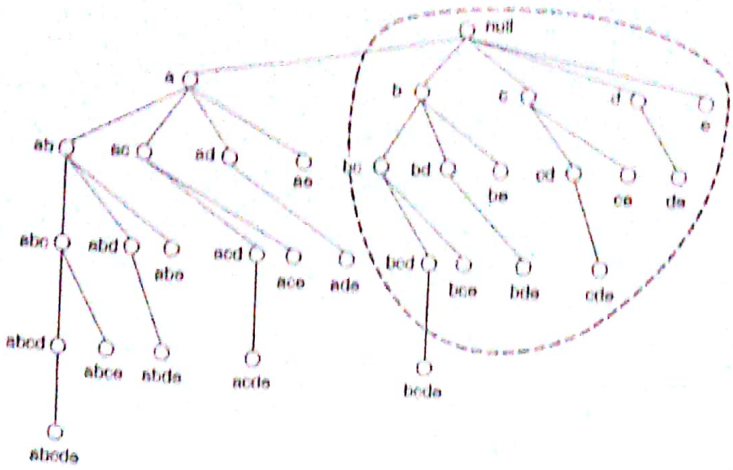


Figure 6.22. Generating candidate itemsets using the depth-first approach

→ Representation of Transaction Data set :-

There are many ways to represent a transaction data set. They are :-

1) Horizontal data layout } shown in
2) Vertical data layout. } Fig. 6.23

1) Horizontal data layout :-

This layout is adopted by many association rule mining algorithms, including Apriori.

Horizontal Data Layout

TID	Items
1	a,b,e
2	b,c,d
3	c,e
4	a,c,d
5	a,b,c,d
6	a,e
7	a,b
8	a,b,c
9	a,c,d
10	b

Vertical Data Layout

a	b	c	d	e
1	1	2	2	1
4	2	3	4	3
5	5	4	5	6
6	7	8	9	
7	8	9		
8	10			
9				

Figure 6.23. Horizontal and vertical data format.

The choice of representation can affect the I/O costs incurred when computing the support of candidate itemsets.

2) Vertical data Layout :-

Another approach is to store the list of transaction identifiers (TID-list) associated with each item. The support for each candidate itemset is obtained by intersecting the TID lists of its subset items.

6.7 ⟹ Topic : 5

Evaluation of Association Patterns :-

Association analysis algorithms have the potential to generate a large number of patterns. The patterns to identify the most interesting ones is not a trivial task because "one person's trash might be another person's treasure". It is therefore important to establish a set of well-accepted criteria for evaluating the quality of association patterns.

→ The first set of criteria can be established through statistical arguments. Patterns involving independent items and few uninteresting transactions which can be

eliminated by applying an Objective Interestingness measure that uses statistics derived from data to determine whether a pattern is interesting.

Eg's :- Support, Confidence & Correlation.

2 → The second set of criteria established through subjective arguements.

A pattern is considered subjectively uninteresting unless it reveals unexpected information about the data or provides useful knowledge that can lead to profitable actions.

Eg: rule {Butter} → {Bread} → may not be interesting
{Diaper} → {Beer} → is interesting

classified according to relationship

→ The following are the some of the approaches for incorporating subjective knowledge into the pattern discovery task

1) Visualization

2) Template - based approach

3) Subjective interestingness measure.

6.7 Objective Measures of Interestingness :-

An objective measure is a data-driven approach for evaluating the quality of association patterns.

→ computed based on the frequency counts tabulated in a contingency table.

Eg :-

Table 6.7. A 2-way contingency table for variables A and B.

	B	$\bar{B}$	
A	f_{11}	f_{10}	f_{1+}
$\bar{A}$	f_{01}	f_{00}	f_{0+}
	f_{+1}	f_{+0}	N

Table 6.7

Contingency table for a pair of binary variables A & B.

→ Notation $\bar{A}$ ($\bar{B}$) → indicate that A(B) is absent from a transaction

Each entry f_{ij} in this 2×2 table denotes a frequency count.

→ For Eg, f_{11} → No. of items A & B appear together in the same transaction.

f_{01} → No. of transactions that contain B but not A

row f_{i+} → Support count for A

column f_{+1} → Support count for B

Contingency tables are applicable not only for asymmetric binary variables but also for symmetric binary, nominal and ordinal variables.

Limitations of the Support-Confidence Framework:-

Existing association rule mining formulation relies on the support and confidence measures to eliminate uninteresting patterns.

The drawback of confidence is more subtle and is best demonstrated with the following example.

Eg:

Table 6.8. Beverage preferences among a group of 1000 people.

	$Coffee$	$\overline{Coffee}$	
Tea	150	50	200
$\overline{Tea}$	650	150	800
	800	200	1000

The information given in this table can be used to evaluate the association rule {Tea} → {Coffee}. Various measures as rules support 15% & confidence 75% values are reasonably high for people those who drink tea also tend to drink coffee. Still having limitations as $A\bar{B}$, $\bar{A}B$ etc, so use various objective measures to evaluate the quality of association patterns.

Interest Factor:-

The tea - coffee example shows that high - confidence rules can sometimes be misleading because the confidence measure ignores the support of the itemset appearing in the rule consequent. One way to address this problem is by applying

a metric known as Lift.

$$\text{Lift} = \frac{c(A \rightarrow B)}{s(B)} \qquad \text{equation 6.4}$$

which computes the ratio between the rule's confidence and the support of the itemset in the rule consequent.

For binary variables, Lift is equivalent to another objective measure called interest factor, is defined as follows:

$$I(A,B) = \frac{s(A,B)}{s(A) \times s(B)} = \frac{N f_{11}}{f_{1+} f_{+1}} \qquad \text{Eq. 6.5}$$

Interest factor compares the frequency of a pattern against a baseline frequency computed under the statistical independence assumption. The baseline frequency for a pair of mutually independent variables is

Eq. 6.6

$$\frac{f_{11}}{N} = \frac{f_{1+}}{N} \times \frac{f_{+1}}{N} \quad \text{or equivalently} \quad f_{11} = \frac{f_{1+} f_{+1}}{N}$$

This equation follows from the standard approach of using simple fractions as estimates for probabilities.

$\Rightarrow \frac{f_{11}}{N}$ → estimate for the joint probability $P(A,B)$

f_{1+}/N & f_{+1}/N → estimates for P(A) and P(B).

Eg:- for tea - coffee example shown in Table 6.8,

$$I = \frac{0.15}{0.2 \times 0.8} \quad \begin{matrix} f_{11} \\ f_{1+} \quad f_{+1} \end{matrix} = 0.9375$$

Limitations of Interest Factor:-

Consider an example, text mining domain ~~is~~ reasonable to assume that the association between a pair of words depends on the number of documents that contain both words.

Eg: Words

data & mining → stronger association than compiler & mining.

Table 6.9 shows the frequency of occurrences between two pair of words {p,q} and {r,s}.

Table 6.9. Contingency tables for the word pairs ({p,q} and {r,s}.

	p	$\bar{p}$	
q	880	50	930
$\bar{q}$	50	20	70
	930	70	1000

	r	$\bar{r}$	
s	20	50	70
$\bar{s}$	50	880	930
	70	930	1000

Using the formula given in equation 6.5, the Interest factor for {p,q} is 1.02 and {r,s} is 4.08.

Correlation Analysis:-

Correlation Analysis is a statistical - based technique for analyzing relationships between a pair of variables.

→ For continuous variables, correlation is defined using Pearson's correlation coefficient.

$$corr(x,y) = \frac{covariance(x,y)}{standard\text{-}deviation(x) * standard\text{-}deviation(y)} = \frac{s_{xy}}{s_x s_y}$$

$$Covariance(x,y) = s_{xy} = \frac{1}{n-1}\sum_{k=1}^{n}(x_k - \bar{x})(y_k - \bar{y})$$

$$standard\text{-}deviation(x) = s_x = \sqrt{\frac{1}{n-1}\sum_{k=1}^{n}(x_k - \bar{x})^2}$$

$$standard\text{-}deviation(y) = s_y = \sqrt{\frac{1}{n-1}\sum_{k=1}^{n}(y_k - \bar{y})^2}$$

$$\bar{x} = \frac{1}{n}\sum_{k=1}^{n} x_k \text{ is the mean of } x$$

$$\bar{y} = \frac{1}{n}\sum_{k=1}^{n} y_k \text{ is the mean of } y.$$

→ For binary variables, correlation can be measured using the ϕ-coefficient is defined as

$$\phi = \frac{f_{11}f_{00} - f_{01}f_{10}}{\sqrt{f_{1+}f_{+1}f_{0+}f_{+0}}}$$

The value of correlation ranges from -1 to +1.

If the variables are statistically independent, then $\phi = 0$.

Eg: The correlation between the tea and coffee drinkers in Table 6.8 is -0.0625.

Limitations of Correlation Analysis:-

The drawback of using correlation can be seen from the word association example given in Table 6.9. Although the words p & q appear together more often than r & s. their ϕ- coefficients are identical

i.e; $\phi(p,q) = \phi(r,s) = 0.232$.

This is because the ϕ- coefficients gives equal importance to both co-presence and co-absence of items in a transaction

IS Measure:-

IS is an alternative measure that has been proposed for handling asymmetric binary variables. The measure is defined as follows:

$$IS(A,B) = \sqrt{I(A,B) \times s(A,B)} = \frac{s(A,B)}{\sqrt{s(A)s(B)}}$$

NOTE: IS is large, when the interest factor and support of the pattern are large.

Eg: Value of IS for the word pairs $\{p,q\}$ and $\{r,s\}$ shown in Table 6.9 are 0.946 and 0.286 respectively. There is a possibility that IS is mathematically equivalent to the cosine measure for binary variables

$$\cos(x,y) = \frac{x \cdot y}{\|x\| \|y\|},$$

"Vector dot Product" → $x \cdot y = \sum_{k=1}^{n} x_k y_k$

Length of Vector → $\|x\| = \sqrt{\sum_{k=1}^{n} x_k^2} = \sqrt{x \cdot x}$

Consider A & B as a pair of bit vectors.

$$IS(A,B) = \frac{s(A,B)}{\sqrt{s(A) \times s(B)}} = \frac{A \cdot B}{|A| \times |B|} = \text{cosine}(A,B).$$

IS measure can also be expressed as the geometric mean between the confidence of association rules extracted from a pair of binary variables.

$$IS(A,B) = \sqrt{\frac{s(A,B)}{s(A)} \times \frac{s(A,B)}{s(B)}} = \sqrt{c(A \rightarrow B) \times c(B \rightarrow A)}$$

Limitations of IS Measure:-

The IS value for a pair of independent itemsets A and B is

$$IS_{indep}(A,B) = \frac{s(A,B)}{\sqrt{s(A) \times s(B)}} = \frac{s(A) \times s(B)}{\sqrt{s(A) \times s(B)}} = \sqrt{s(A) \times s(B)}$$

Eg: IS value between items p & q in Table 6.10 (0.889), it is still than the expected value when the items are statistically independant ($IS_{indep} = 0.9$).

→ Alternative Objective Interestingness Measures:-

Alternative Measures proposed for analyzing relationships between pairs of binary variables can be

divided into two categories.

1) Symmetric
2) Asymmetric

Properties of Objective Measures:-

1) Inversion Property
2) Scaling Property
3) Null Addition Property.

1) Inversion Property:-

The process of flipping a bit vector is called Inversion.

Eg: Consider Vectors A to F

A	B	C	D	E	F
1	0	0	1	0	0
0	0	1	1	1	0
0	0	1	1	1	0
0	0	1	1	1	0
0	1	1	0	1	1
0	0	1	1	1	0
0	0	1	1	1	0
0	0	1	1	1	0
1	0	0	1	0	0

The Vectors C & E are in fact related to the Vector A. Similarly D related to vectors B & F. by inverting their bits.

Their bits have been inverted from 0's (absence) to 1's (presence) and vice versa.

Null Addition Property :-

The process of adding unrelated data (documents) to a given set is known as Null addition Property Operation.

For applications such as document analysis or market basket analysis, the measure is expected invariant under the Null addition operation.

Scaling Property :-

→ An Evaluation measure is invariant under row / column each column and row can be multiplied by a constant without the measure to change its value.

The association between grade and gender is expected to remain unchanged despite changes in the sampling distribution as shown in below Example.

Table 6.16. The grade-gender example.

	Male	Female	
High	30	20	50
Low	40	10	50
	70	30	100

(a) Sample data from 1993.

	Male	Female	
High	60	60	120
Low	80	30	110
	140	90	230

(b) Sample data from 2004.

6.7 Measures beyond pairs of binary Variables :-

Measures include Interest Factor, Ps, Jaccard coefficient, etc. Consider an example of 3-dimensional contigency table for a, b & c as shown in below Table.

Table 6.18. Example of a three-dimensional contingency table.

c	b	$\bar{b}$	
a	f_{111}	f_{101}	f_{1+1}
$\bar{a}$	f_{011}	f_{001}	f_{0+1}
	f_{+11}	f_{+01}	f_{++1}

$\bar{c}$	b	$\bar{b}$	
a	f_{110}	f_{100}	f_{1+0}
$\bar{a}$	f_{010}	f_{000}	f_{0+0}
	f_{+10}	f_{+00}	f_{++0}

→ The measure considers only pairwise associations it may not capture all the underlying relationships within a pattern.

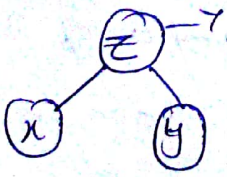
→ Analysis of multidimensional contingency tables is more complicated because of the presence of partial associations in the data.

→ For Eg, some associations may appear or disappear when conditioned upon the value of certain variables. This problem is known as "Simpson's Paradox".

6.7.3 → Simpson's Paradox :-

Simpson's Paradox is a phenomenon that the hidden variables may cause the observed relationship between a pair of variables to disappear or reverse its direction. Hidden variables nothing but a confounding factor

Eg :



→ Confounding factor

No relation between x & y but there's a relation through z.

Eg:

Table 6.19. A two-way contingency table between the sale of high-definition television and exercise machine.

Buy HDTV	Buy Exercise Machine		
	Yes	No	
Yes	99	81	180
No	54	66	120
	153	147	300

Table 6.20. Example of a three-way contingency table.

Customer Group	Buy HDTV	Buy Exercise Machine		Total
		Yes	No	
College Students	Yes	1	9	10
	No	4	30	34
Working Adult	Yes	98	72	170
	No	50	36	86

Rule: Table 6.19

{HDTV = yes} → {Exercise Machine = yes} ⇒ confidence = $99/180 = 55\%$.

{HDTV = No} → { " " = yes} ⇒ confidence = $\frac{54}{120} = 45\%$.

Table 6.20

Rule :-

for College students :

$c(\{HDTV = yes\} \rightarrow \{Exercise\ Machine = yes\}) = 1/10 = 10\%$.

$c(\{HDTV = No\} \rightarrow \{ " \quad " = yes\}) = 4/34 = 11.8\%$.

For Working Adults :-

$c(\{HDTV = yes\} \rightarrow \{Exercise\ Machine = yes\}) = 98/170 = 57.7\%$.

$c(\{HDTV = No\} \rightarrow \{Exercise\ Machine = yes\}) = 50/86 = 58.1\%$.

Consider, relationship between HDTV and exercise machine turns out to be stronger in the combined data than what it would have been if the data is stratified. (2 or more events cannot occur simultaneously)