
PES Institute of Technology
Bangalore South Campus
(1 K.M before Electronic City ,Bangalore 560100)

Solution Set -Test-II

Sub: Data Warehousing and Data mining (13MCA442)

Sem & Section:IV Sem,MCA
Name of the Faculty: R.Jayanthi

1.What is OLAP? Explain Characteristics of OLAP
i)FASMI ii) Codd's (10 Marks)

- Codd defines *On-line Analytical Processing or OLAP* as *the dynamic enterprise analysis required to create, manipulate, animate, and synthesize information from exegetical, contemplative, and formulaic data analysis models.*
- OLAP generally involves highly complex queries involving large amounts of data that use one or more aggregates. OLAP deals only with historical data accurate at a given point in time.
- OLAP is a software technology that enables
- analysts, managers and executives to gain insight
- into data through fast, consistent, interactive access
- to a wide variety of possible views of information that has been transformed from raw data to reflect the real dimensionality of the enterprise as understood by the user.
- OLAP is different from ODS and Data warehouse,
- It is primarily a software technology concerned with fast analysis of enterprise information.
- Often OLAP systems are data warehouse front end software tools to make aggregation data available efficiently, for advanced analysis, to an enterprise's measures.

Characteristics of OLAP systems

- Users : Decision makers
- Functions : Management critical
- Nature of queries : Mostly complex
- Nature of usage: Mostly adhoc
- Nature of Design: Subject oriented
- Number of users: Dozens
- Nature of Data: Historical, summarized, multidimensional
- Updates: Not allowed

- **FASMI Characteristics**
- Fast
- Analytic
- Shared
- Multidimensional
- Information

Codd's OLAP characteristics

The most important 10 rules are

1. Multidimensional conceptual view
2. Accessibility
3. Batch extraction vs interpretive
4. Multi user support
5. Strong OLAP results
6. Extraction of missing values
7. Treatment of missing values
8. Uniform reporting performance
9. Generic dimensionality
10. Unlimited dimensions and aggregation levels

2. Discuss on Data Cube operations with suitable examples(10 Marks)

Possible Solutions:

- Pre compute and store all (expensive) *compute all queries and stored.*
- Pre compute and store none (response time poor for large data cube) This means that the aggregates are computed on-the-fly using the raw data whenever a query is posed.
- Pre compute and store some (most frequent) This means that we pre-compute and store the most frequently queried aggregates and compute others as the need arises

Shows how the aggregates above are related and how an aggregate
at a higher level may be computed from the aggregates below.

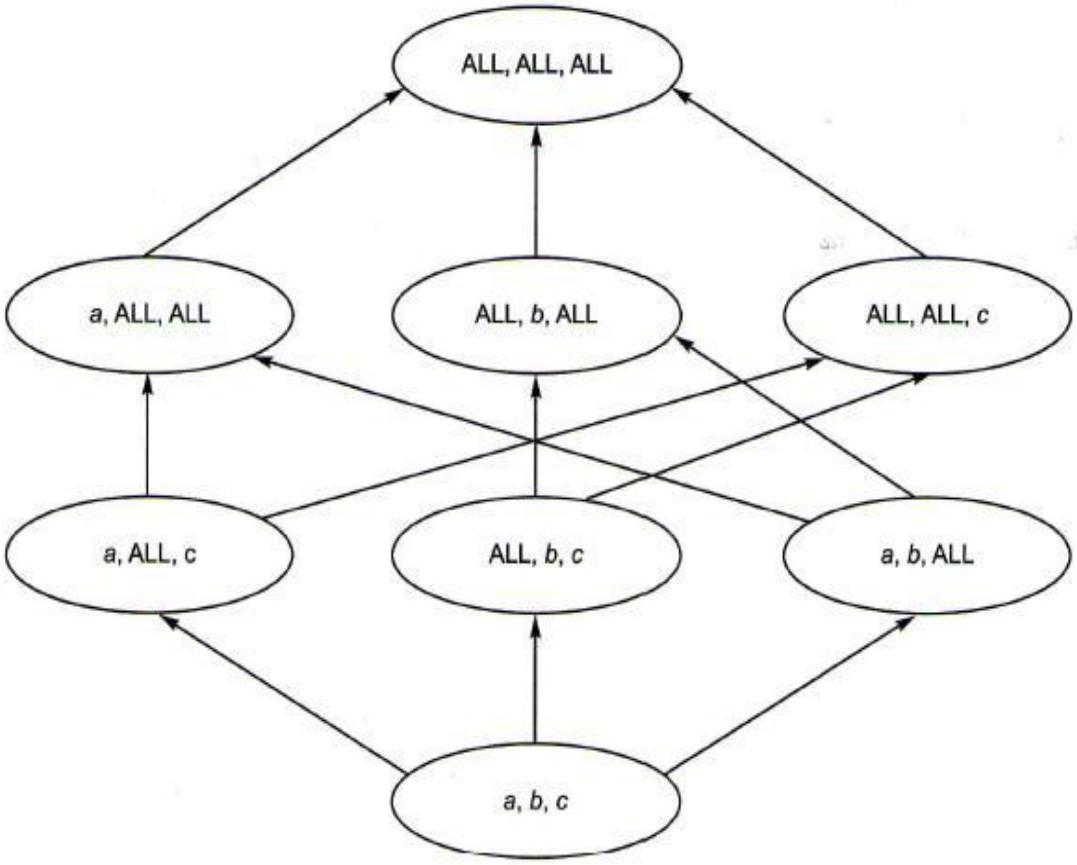


Figure 8.3 Relationships between aggregations of a three-dimensional cube.

Typical data cube Operations

- **Roll up (drill-up): summarize data**
  - *by climbing up hierarchy or by dimension reduction*
- **Drill down (roll down): reverse of roll-up**
  - *from higher level summary to lower level summary or detailed data, or introducing new dimensions*
- **Slice and dice:** *project and select*
- **Pivot (rotate):**
  - *reorient the cube, visualization, 3D to series of 2D planes*
- **Other operations**
  - *drill across: involving (across) more than one fact table*
    drill through: through the bottom level of the cube to its back-end relational tables (using SQL)

Roll-up

Roll-up is like zooming out on the data cube. It is required when the user needs further abstraction or less detail

Drill-down

Drill-down is like zooming in on the data and is therefore the reverse of roll-up. It is an appropriate operation when the user needs further detail. Drill-down adds more details to the data. Hierarchy defined on a dimension may be involved in drill-down

Slice and Dice – slice is performed on one dimenstion

Dice is performed on two or more dimensions Slice and dice are operations for browsing the data in the cube. The terms refer to the ability to look at information from different viewpoints A slice is a subset of the cube corresponding to a single value for one or more members of the dimensions Let the degree dimension be fixed as degree = BIT. The slice will not include any information about other degrees

Pivoted or Rotate:

The pivot operation is used when the user wishes to re-orient the view of the data cube. It may involve swapping the rows and columns, or moving one of the row dimensions into the column dimension

3. What is an Association rule? Discuss its importance.

Given a set of transactions, find rules that will predict the occurrence of an item based on the occurrences of other items in the transaction

TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Egs. Of Association rules

{Diaper} → {Beer},
{Milk, Bread} → {Eggs,Coke},
{Beer, Bread} → {Milk},

Association analysis is useful for discovering interesting relationships hidden in large amount of data.

From Table it is clear that the people who buy bread will also buy milk too. {Bread} à {Milk}

There are two key Issues that need to be addressed when applying association analysis to market basket data.

- First, discovering patterns from a large transaction data set can be computationally expensive.
- Second, some of the discovered patterns are potentially spurious (fake) because they may happen simply by chance.

Define i)Support ii)Confidence iii)Frequent item set(4+6 Marks)

Definition: Frequent Itemset

- **Itemset**
  - A collection of one or more items
    - Example: {Milk, Bread, Diaper}
  - k-itemset
    - An itemset that contains k items
- **Transaction Width:** is defined as the number of items present in a transaction.
- **Important Property of ItemSet is: "Support and Count"** which refers to the number of transactions that contain a particular itemset.

TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

- **Support count (σ)**
  - Frequency of occurrence of an itemset
  - E.g. σ({Milk, Bread,Diaper}) = 2
- **Support :** Support determines how often a rule is applicable to a given Data set
  - Fraction of transactions that contain an itemset
  - E.g. s({Milk, Bread, Diaper}) = 2/5
- **Frequent Itemset**
  - An itemset whose support is greater than or equal to a *minsup* threshold

Definition: Association Rule

- **Association Rule**
  - An implication expression of the form $X \rightarrow Y$, where X and Y are itemsets
  - Example: {Milk, Diaper} → {Beer}
- **Rule Evaluation Metrics**
  - Support (s)
    - Fraction of transactions that contain both X and Y
  - Confidence (c)
    - Measures how often items in Y appear in transactions that contain X

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Example:

$$\{\text{Milk}, \text{Diaper}\} \Rightarrow \text{Beer}$$

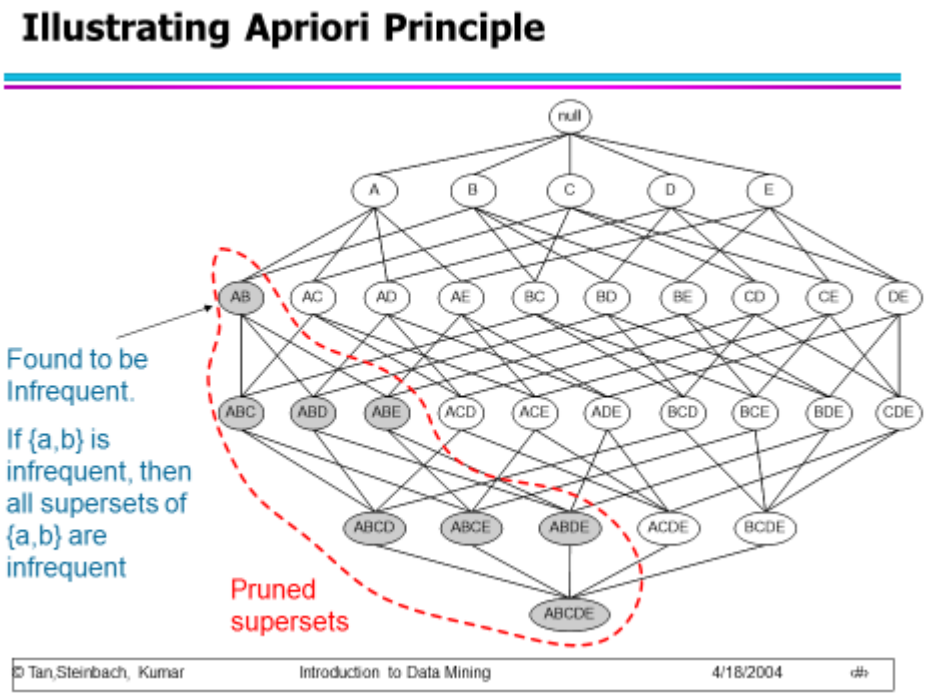
$$support = \frac{(X \cup Y).count}{n}$$

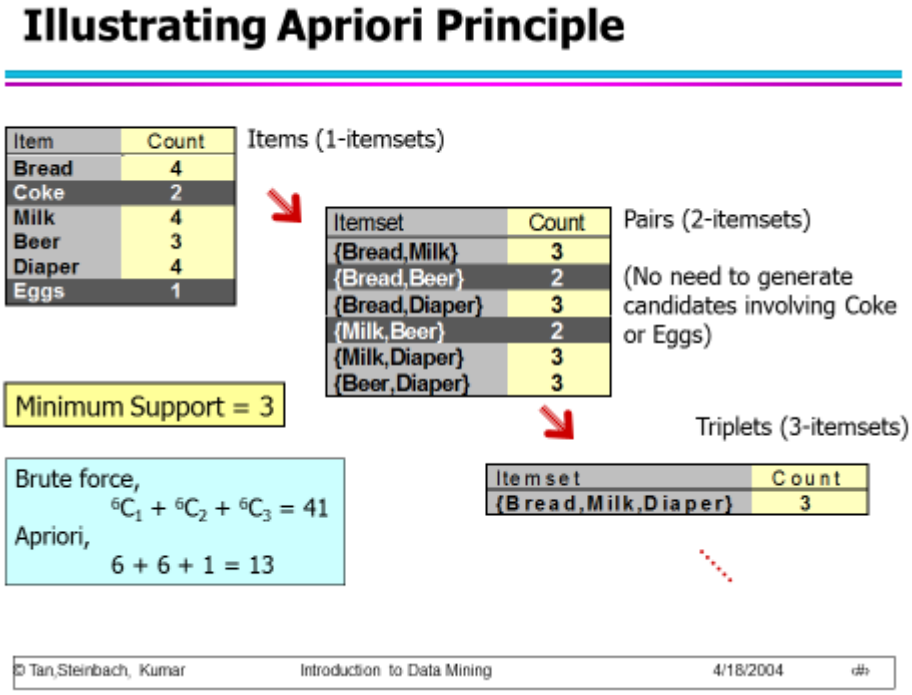
$$s = \frac{\sigma(\text{Milk}, \text{Diaper}, \text{Beer})}{|T|} = \frac{2}{5} = 0.4$$

$$confidence = \frac{(X \cup Y).count}{X.count}$$

$$c = \frac{\sigma(\text{Milk}, \text{Diaper}, \text{Beer})}{\sigma(\text{Milk}, \text{Diaper})} = \frac{2}{3} = 0.67$$

4. State Apriori principle. Explain how it is used for support-based pruning. Write a pseudo code for Apriori algorithm.(10 Marks)





- Initially, every item is considered as a candidate 1-itemset. After counting their supports, the candidate itemsets {Cola} and {Eggs} are discarded because they appear in fewer than three transactions.
- In the next iteration, candidate 2-itemsets are generated using only the frequent 1-itemsets because the Apriori principle ensures that all supersets of the infrequent 1-itemsets must be infrequent.
- Two of these six candidates, {Beer, Bread} and {Beer, Milk}, are subsequently found to be infrequent after computing their support values. The

remaining four candidates are frequent, and thus will be used to generate candidate 3-itemsets.

- With the Apriori principle, we only need to keep candidate 3-itemsets whose subsets are frequent. The only candidate that has this property is {Bread, Diapers, Milk}.
- Using Brute force strategy 6+15+20=41 (6 c1+6c2+6c3)
- Using Apriori 6+6+1=3 (6c1+4c2)

Apriori Algorithm

Method:

Let k=1

Generate frequent itemsets of length 1

Repeat until no new frequent itemsets are identified

i)Generate length (k+1) candidate itemsets from length k frequent itemsets

ii)Prune candidate itemsets containing subsets of length k that are infrequent

iii)Count the support of each candidate by scanning the DB

Eliminate candidates that are infrequent, leaving only those that are frequent

5. Draw F-P tree for the given set of Transactions and generate the frequent item set
T1{a,b}
T2{b,c,d}
T3{a,c,d,e}
T4{a,d,e}
T5{a,b,c}
T6{a,b,c,d}
T7{a}
T8{a,b,c}
T9{a,b,d}
T10{b,c,e} **(10 Marks)**

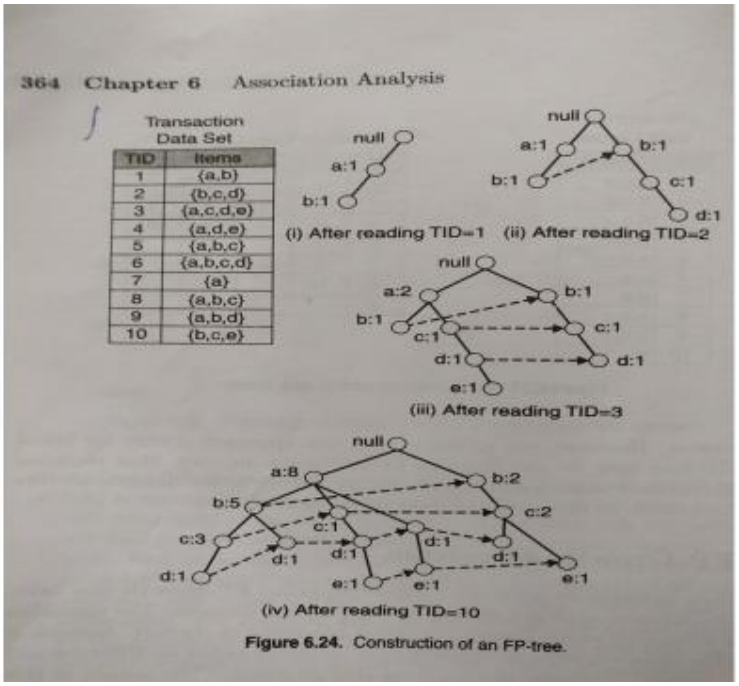


Figure 6.24. Construction of an FP-tree.

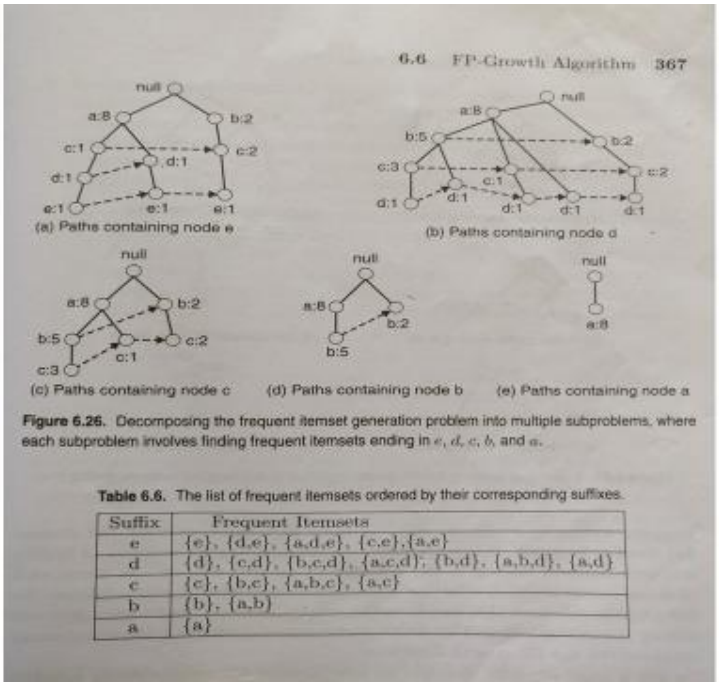
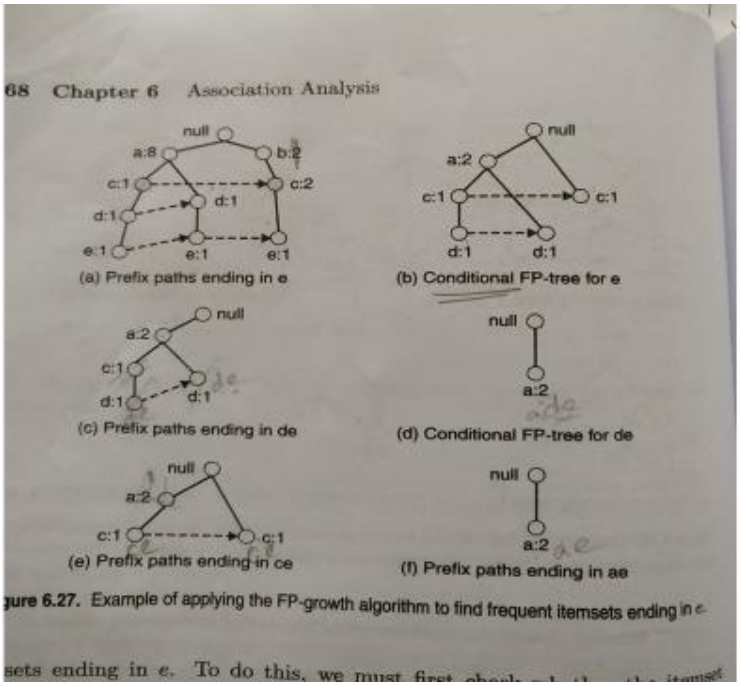


Figure 6.26. Decomposing the frequent itemset generation problem into multiple subproblems, where each subproblem involves finding frequent itemsets ending in e, d, c, b, and a.

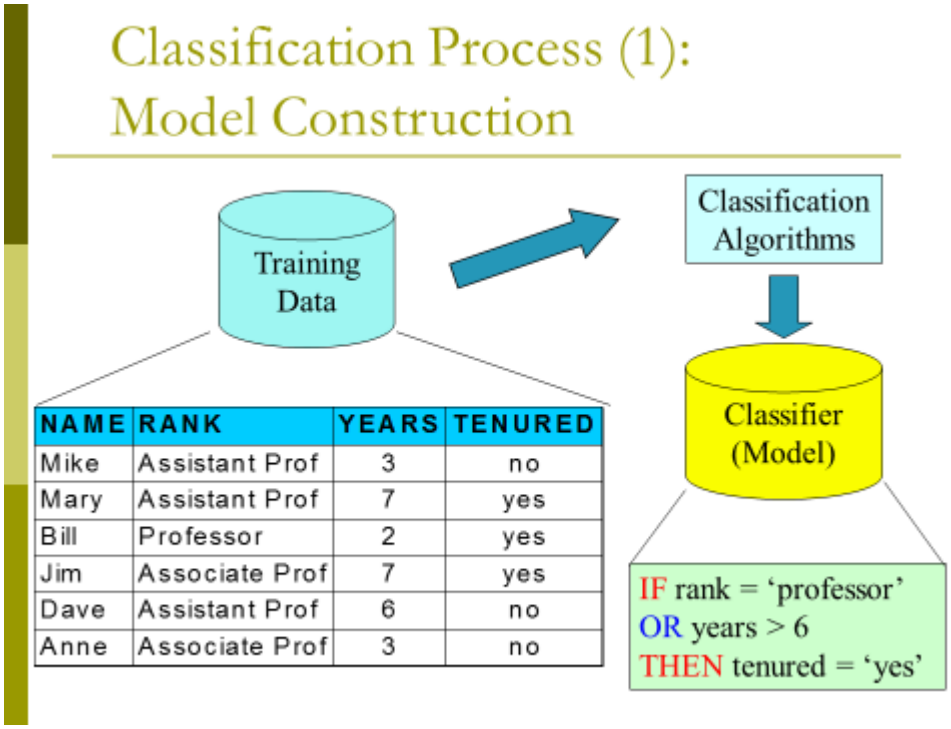
Table 6.6. The list of frequent itemsets ordered by their corresponding suffixes.

Suffix	Frequent Itemsets
e	{e}, {d,e}, {a,d,e}, {c,e},{a,e}
d	{d}, {c,d}, {b,c,d}, {a,c,d}, {b,d}, {a,b,d}, {a,d}
c	{c}, {b,c}, {a,b,c}, {a,c}
b	{b}, {a,b}
a	{a}



6. What is classification problem(4 Marks)

- **Classification:**
  - predicts categorical class labels
  - classifies data (constructs a model) based on the training set and the values (class labels) in a classifying attribute and uses it in classifying new data
    Typical Applications
  - credit approval
  - target marketing
  - medical diagnosis
  - treatment effectiveness analysis
- Credit approval
  - A bank wants to classify its customers based on whether they are expected to pay back their approved loans
  - The history of past customers is used to train the classifier
  - The classifier provides rules, which identify potentially reliable future customers
  - Classification rule:
    if age = "31...40" and income = high then credit_rating = excellent
  - Future customers
    Paul: age = 35, income = high ⇒ excellent credit rating
    John: age = 20, income = medium ⇒ fair credit rating



NAME	RANK	YEARS	TENURED
Mike	Assistant Prof	3	no
Mary	Assistant Prof	7	yes
Bill	Professor	2	yes
Jim	Associate Prof	7	yes
Dave	Assistant Prof	6	no
Anne	Associate Prof	3	no

Classification—A Two-Step Process

- Model construction: describing a set of predetermined classes
  - Each tuple/sample is assumed to belong to a predefined class, as determined by the class label attribute
  - The set of tuples used for model construction: training set
  - The model is represented as classification rules, decision trees, or mathematical formulae
- Model usage: for classifying future or unknown objects
  - Estimate accuracy of the model
    - The known label of test samples is compared with the classified result from the model
    - Accuracy rate is the percentage of test set samples that are correctly classified by the model
    - Test set is independent of training set, otherwise over-fitting will occur

Evaluating Classification Methods

- **Predictive accuracy**
- **Speed**
  - **time to construct the model**
  - **time to use the model**
- **Robustness**
  - **handling noise and missing values**
- **Scalability**
  - **efficiency in disk-resident databases**
- **Interpretability:**
  - **understanding and insight provided by the model**
- **Goodness of rules (quality)**
  - **decision tree size**
  - **compactness of classification rules**

b. Write and explain the Decision tree Induction algorithm(6 Marks)

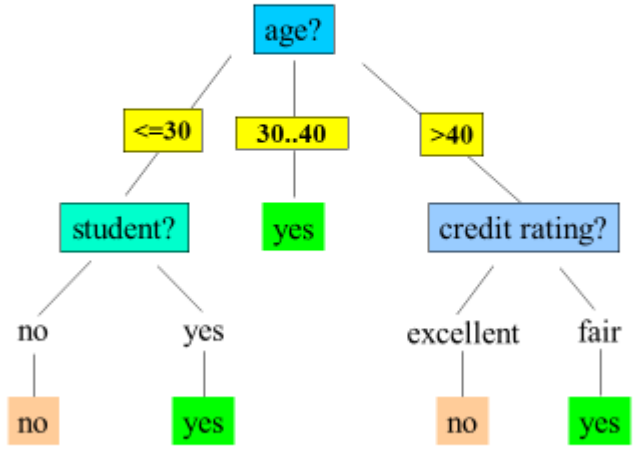
Classification by Decision Tree Induction

- **Decision tree**
  - A flow-chart-like tree structure
  - Internal node denotes a test on an attribute
  - Branch represents an outcome of the test
  - Leaf nodes represent class labels or class distribution
- **Decision tree generation consists of two phases**
  - **Tree construction**
    - At start, all the training examples are at the root
    - Partition examples recursively based on selected attributes
  - **Tree pruning**
    - Identify and remove branches that reflect noise or outliers
    - Use of decision tree: Classifying an unknown sample
    - Test the attribute values of the sample against the decision tree

Training Dataset

age	income	student	credit_rating	buys_comp
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

Output: A Decision Tree for "buys_computer"



Algorithm for Decision Tree Induction

- Basic algorithm (a greedy algorithm)
  - Tree is constructed in a top-down recursive divide-and-conquer manner
  - At start, all the training examples are at the root
  - Attributes are categorical (if continuous-valued, they are discretized in advance)
  - Samples are partitioned recursively based on selected attributes
  - Test attributes are selected on the basis of a heuristic or statistical measure (e.g., information gain)
- Conditions for stopping partitioning
  - All samples for a given node belong to the same class
  - There are no remaining attributes for further partitioning – majority voting is employed for classifying the leaf
  - There are no samples left

Algorithm for Decision Tree Induction (pseudocode)
Algorithm GenDecTree(Sample S, Attlist A)

1. create a node N
2. If all samples are of the same class C then label N with C; terminate;
3. If A is empty then label N with the most common class C in S (majority voting); terminate;
4. Select a∈A, with the highest information gain; Label N with a;
5. For each value v of a:
   a. Grow a branch from N with condition a=v;
   b. Let S_v be the subset of samples in S with a=v;
   c. If S_v is empty then attach a leaf labeled with the most common class in S;

Else attach the node generated by GenDecTree(S_v, A-a)

7. Explain the various measures for selecting the best split with an example ,for each attribute type(10 Marks)

Attribute Selection Measure: Information Gain (ID3/C4.5)

- Select the attribute with the highest information gain
- Let p_i be the probability that an arbitrary tuple in D belongs to class C_i, estimated by $|C_{i,D}|/|D|$
- Expected information (entropy) needed to classify a tuple in D:

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)$$

- **Information needed (after using A to split D into v partitions) to classify D:**

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times I(D_j)$$

- **Information gained by branching on attribute A**

$$Gain(A) = Info(D) - Info_A(D)$$

Attribute Selection: Information Gain

Class P: buys_computer = "yes"
Class N: buys_computer = "no"

$$Info(D) = I(9,5) = -\frac{9}{14}\log_2(\frac{9}{14}) - \frac{5}{14}\log_2(\frac{5}{14}) = 0.940$$

age	p_i	n_i	$I(p_i, n_i)$
<=30	2	3	0.971
31…40	4	0	0
>40	3	2	0.971

$$Info_{age}(D) = \frac{5}{14} I(2,3) + \frac{4}{14} I(4,0) + \frac{5}{14} I(3,2) = 0.694$$

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

$$Gain(age) = Info(D) - Info_{age}(D) = 0.246$$

$$Gain(income) = 0.029$$
$$Gain(student) = 0.151$$
$$Gain(credit\_rating) = 0.048$$

Comparing Attribute Selection Measures

- **The three measures, in general, return good results but**
  - **Information gain:**
    - biased towards multivalued attributes
  - **Gain ratio:**
    - tends to prefer unbalanced splits in which one partition is much smaller than the others
  - **Gini index:**
    - biased to multivalued attributes
    - has difficulty when # of classes is large

- tends to favor tests that result in equal-sized partitions and purity in both partitions

Gini index (CART, IBM IntelligentMiner)

- If a data set D contains examples from n classes, gini index, $gini(D)$ is defined as

$$gini(D) = 1 - \sum_{j=1}^{n} p_j^2$$

where p_j is the relative frequency of class j in D

- If a data set D is split on A into two subsets D_1 and D_2, the *gini* index $gini(D)$ is defined as

$$gini_A(D) = \frac{|D_1|}{|D|} gini(D_1) + \frac{|D_2|}{|D|} gini(D_2)$$

- Reduction in Impurity:

$$\Delta gini(A) = gini(D) - gini_A(D)$$

- The attribute provides the smallest $gini_{split}(D)$ (or the largest reduction in impurity) is chosen to split the node (*need to enumerate all the possible splitting points for each attribute*)

8.a.Write a note on Rule generation and pruning(6 Marks)
Rule generation

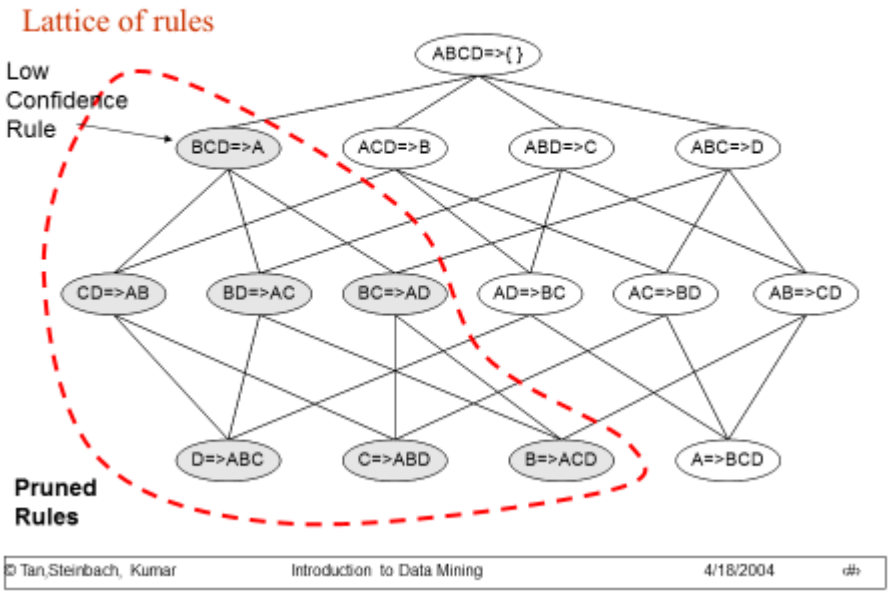
l How to efficiently generate rules from frequent itemsets?
- In general, confidence does not have an anti-monotone property

c(ABC →D) can be larger or smaller than c(AB →D)
- But confidence of rules generated from the same itemset has an anti-monotone property
- e.g., L = {A,B,C,D}:

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

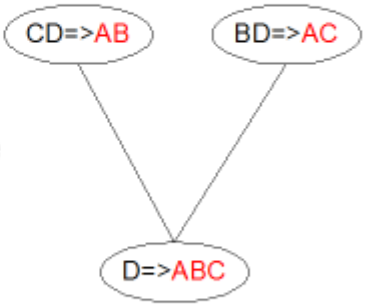
◆ Confidence is anti-monotone w.r.t. number of items on the RHS of the rule

Rule Generation for Apriori Algorithm



Rule Generation for Apriori Algorithm

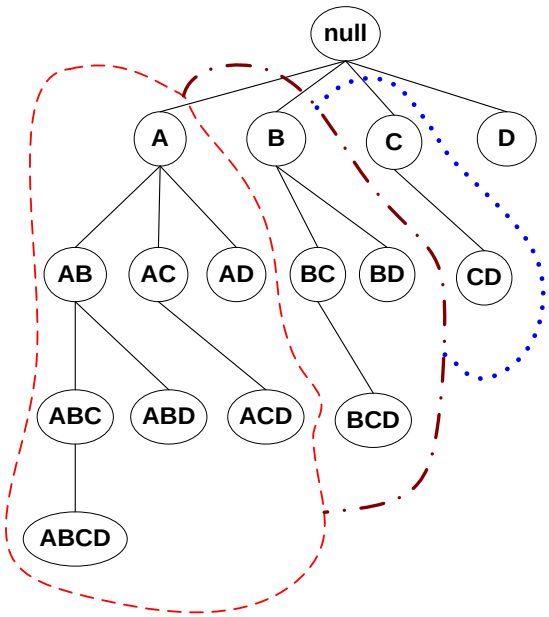
- Candidate rule is generated by merging two rules that share the same prefix in the rule consequent
- join(CD=>AB,BD=>AC) would produce the candidate rule D => ABC
- Prune rule D=>ABC if its subset AD=>BC does not have high confidence



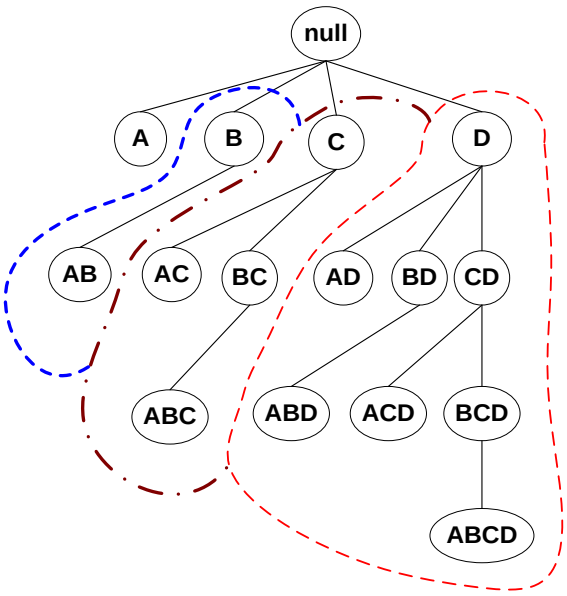
b. Short notes on Equivalence classes(4 Marks)

Alternative Methods for Frequent Itemset Generation

- **Traversal of Itemset Lattice**
  - **Equivalent Classes**
- Another way to envision the traversal is to first partition the lattice into disjoint groups of nodes (or equivalence classes). A frequent itemset generation algorithm searches for frequent itemsets within a particular equivalence class first before moving to another equivalence class



(a) Prefix tree



(b) Suffix tree